Hanoi University of Science and Technology
School of Informations and Comunication of Technology

Introduction to Deep Learning
COURSE: IT3320E

Capstone Project

RestoraVision: A Comprehensive System for Image Enhancement and Restoration

Group 24 - Members:

Member 1:	**Đàm Quang Đức**	**20225483**
Member 2:	**Trần Hữu Đạo**	**20220061**
Member 3:	**Vũ Hữu An**	**20225467**
Member 4:	**Nguyễn Đức Anh**	**20225468**
Member 5:	**Nguyễn Trọng Tâm**	**20225527**

December, 2024

Contribution

Name	Model	Contribution
Đàm Quang Đức	FSRCNN	20%
Trần Hữu Đạo	DiffIR	20%
Vũ Hữu An	SRGAN	20%
Nguyễn Đức Anh	NAFNet	20%
Nguyễn Trọng Tâm	SwinIR	20%

Contents

Abstract

Image restoration and enhancement play a critical role in diverse applications, from medical imaging to satellite analysis. This report introduces *RestoraVision*, a unified system designed to tackle two primary tasks: image super-resolution and image deblurring. By leveraging state-of-the-art models such as FSRCNN, SwinIR, SRGAN for super-resolution, and NAFNet for deblurring, RestoraVision delivers high-quality outputs tailored for various use cases. Additionally, the DiffIR framework is integrated to unify and enhance both tasks, offering significant improvements in efficiency and restoration quality. Experimental results demonstrate RestoraVision's superior performance in quantitative metrics and visual fidelity, establishing it as a comprehensive solution for image restoration.

1 Introduction

Image restoration and enhancement have become pivotal in addressing challenges posed by degraded visual data in domains such as healthcare, remote sensing, and digital photography. High-quality restoration is essential for extracting meaningful insights and improving user experience. However, achieving this quality is often hindered by factors like low resolution and motion blur, which demand robust solutions.

To address these challenges, we present *RestoraVision*, a comprehensive system that integrates cutting-edge models for image super-resolution and deblurring. RestoraVision bridges the gap between traditional restoration techniques and modern deep learning approaches by employing specialized models for each task. Additionally, the DiffIR framework is introduced as a unifying component that enhances the system's adaptability and performance across both tasks. This report details our methodology, experimental findings, and the significant impact of RestoraVision on advancing the field of image restoration.

2 Model Selection and Framework Design

Image restoration tasks such as super-resolution and deblurring have been extensively studied in the literature. Traditional approaches, including interpolation techniques and deconvolution algorithms, often struggle with artifacts and lack the capacity to generalize across varying degradation levels.

With the advent of deep learning, models like SRCNN and EDSR have significantly advanced super-resolution tasks by employing convolutional neural networks (CNNs) to learn hierarchical features. Transformer-based methods, such as SwinIR, have further improved performance by capturing long-range dependencies. Generative adversarial networks (GANs), exemplified by SRGAN, have demonstrated the ability to produce perceptually realistic high-resolution images. For image deblurring, models like DeblurGAN and DMPHN have shown promise in removing motion artifacts. Simplified architectures

such as NAFNet have gained attention for their efficiency and effectiveness in handling complex blur patterns.

The choice of models for RestoraVision was driven by a careful consideration of their strengths and compatibility with the system's goals. For image super-resolution, FSRCNN was selected for its efficiency and speed, making it ideal for real-time applications. SwinIR was chosen for its ability to capture long-range dependencies and preserve intricate details, while SRGAN was included for its emphasis on perceptual quality, delivering visually appealing results. In the context of image deblurring, NAFNet emerged as the optimal choice due to its streamlined architecture and robust performance in addressing various blur patterns. The integration of these models ensures that RestoraVision achieves a balance between computational efficiency and restoration quality, making it adaptable to diverse scenarios.

Central to the framework is the DiffIR component, which was selected for its capacity to unify the tasks of super-resolution and deblurring. DiffIR's innovative approach to shared training and feature enhancement plays a crucial role in the system's overall efficiency and performance. Its dual-stage training process, comprising pretraining and direct training of the diffusion model, ensures efficient convergence and high-quality outputs. The integration of DiffIR not only simplifies the overall architecture but also improves adaptability to varying restoration scenarios.

This section highlights the rationale behind our model selection and framework design, emphasizing the synergy between these components in achieving the objectives of RestoraVision.

3 Methodology

3.1 Image Super-Resolution

3.1.1 Swin IR

Introduction.

Image restoration tasks focus on reconstructing high-quality images from their degraded counterparts. Convolutional neural networks (CNNs) have emerged as the dominant approach for image restoration, thanks to significant advancements in architecture design, such as residual learning and dense connections. These CNN-based models have consistently demonstrated improved performance compared to traditional model-based methods.

Transformers, on the other hand, leverage self-attention mechanisms to capture global context and have achieved remarkable success in various vision tasks. However, vision Transformers for image restoration typically divide the input image into fixed-size patches, processing each independently. This approach presents two challenges: border pixels cannot leverage neighboring pixels outside the patch, and border artifacts may appear around patch boundaries. While overlapping patches can mitigate these issues, they come at the cost of increased computational demands.

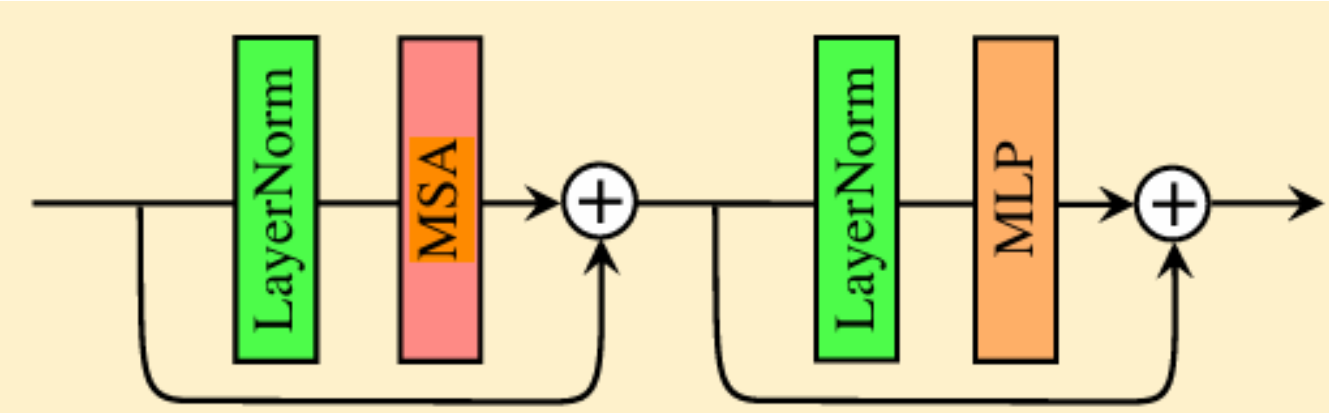

Figure 1: Swin Transformer layer(STL)

The Swin Transformer offers a promising solution by combining the strengths of CNNs and Transformers. It adopts a local attention mechanism, enabling the processing of large images like CNNs, while also modeling long-range dependencies through a shifted window scheme.

Building on this, there is SwinIR, an image restoration model based on the Swin Transformer architecture. SwinIR comprises three main components: shallow feature extraction, deep feature extraction, and high-quality image reconstruction. The shallow feature extraction module uses a convolutional layer to capture low-frequency information, which is directly passed to the reconstruction module. The deep feature extraction module utilizes residual Swin Transformer blocks (RSTBs), incorporating Swin Transformer layers for local attention and cross-window interactions. A convolutional layer at the end of each block enhances features, while residual connections enable effective feature aggregation. Finally, the reconstruction module fuses shallow and deep features to produce high-quality images.

Compared to traditional CNN-based models, SwinIR offers several advantages: (1) content-based interactions between image content and attention weights, functioning similarly to spatially varying convolutions; (2) the ability to model long-range dependencies through the shifted window mechanism; and (3) superior performance with fewer parameters. This combination of benefits allows SwinIR to outperform existing methods for image super-resolution while maintaining efficiency.

Residual Swin Transformer Block.

The Swin Transformer Layer (STL) builds upon the conventional multi-head self-attention mechanism from the original Transformer layer. The main distinctions lie in the localized attention approach and the shifted window scheme. As shown Fig.1, the Swin Transformer first reshapes the input of size $H \times W \times C$ by segmenting it into non-overlapping $M \times M$ windows, where $H_{W \times H}$ is the total number of windows. It then calculates the self-attention representation for each window (i.e., localized attention). For a local window feature $X \in \mathbb{R}^{M \times M \times C}$, the query, key, and value matrices Q, K, V are computed as follows:

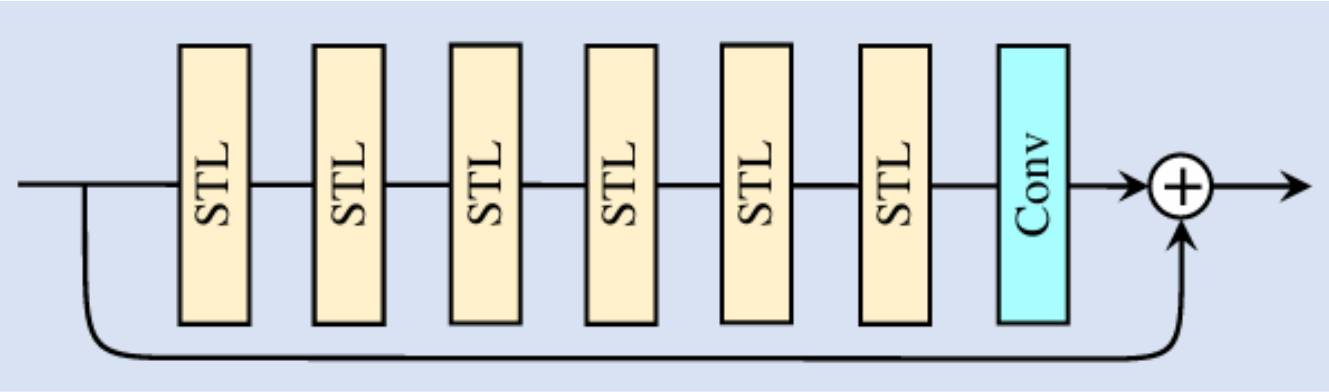

Figure 2: Residual Swin Transformer Block(RSTB)

$$Q = XP_Q, \quad K = XP_K, \quad V = XP_V, \tag{1}$$

where P_Q, P_K, P_V are projection matrices that are consistent across different windows. In general, we have $Q, K, V \in \mathbb{R}^{M^2 \times d}$. The attention matrix is then derived using the self-attention mechanism within a local window:

$$\text{Attention}(Q, K, V) = \text{SoftMax}\left(\frac{QK^\top}{\sqrt{d}} + B\right), \tag{2}$$

where B is the learnable relative positional encoding. In practice, the attention function for h is executed in parallel, with concatenation of results for multi-headed self-attention (MSA). Subsequently, a multi-layer perceptron (MLP) that consists of two fully connected layers with GELU non-linearity is employed for further transformation. A LayerNorm (LN) layer is applied prior to both MSA and MLP modules. The overall process is defined as follows:

$$X = \text{MSA}(\text{LN}(X)) + X \tag{3}$$

$$X = \text{MLP}(\text{LN}(X)) + X \tag{4}$$

As shown in Fig.2, the residual Swin Transformer block (RSTB) serves as a residual component that integrates Swin Transformer layers (STL) with convolutional layers. For the input feature $F_{i,0}$ of the i-th RSTB, we first extract a sequence of intermediate features $F_{i,1}, F_{i,2}, \ldots, F_{i,L}$ using L Swin Transformer layers.

$$F_{i,j} = H_{\text{STL}_{i,j}}(F_{i,j-1}), \quad j = 1, 2, \ldots, L, \tag{5}$$

where $H_{\text{STL},i,j}(\cdot)$ is the j-th Swin Transformer layer in the i-th RSTB. Then, we add a convolutional layer before the residual connection. The output of the RSTB is formulated as

$$F_{i,\text{out}} = H_{\text{CONV}_i}(F_{i,L}) + F_{i,0} \tag{6}$$

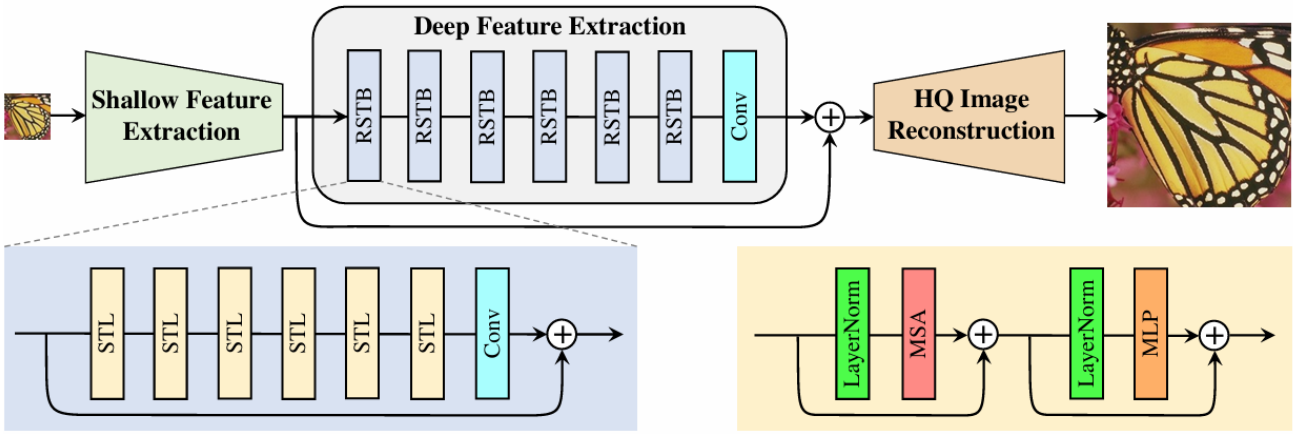

Figure 3: The architecture of SwinIR for super resolution

where $H_{\text{CONV}_i}(\cdot)$ is the convolutional layer in the i-th RSTB. This design offers two advantages: first, while Transformers can be seen as a form of spatially varying convolution, convolutional layers with invariant filters improve translational equivalence. Second, the residual connection facilitates identity-based aggregation from different blocks to the reconstruction module, enabling the combination of various feature levels.

SwinIR model.

As shown in Fig.3, SwinIR consists of three modules: shallow feature extraction, deep feature extraction and high quality (HQ) image reconstruction modules. We employ the same feature extraction modules for all restoration tasks, but use different reconstruction modules for different tasks.

The low-quality (LQ) input image $I_{LQ} \in \mathbb{R}^{H\times W\times C_{in}}$ (H, W, C_{in} are height, width, input channel number). To extract the shallow feature, we use the convolution layer with the shape of 3×3 $H_{SF}(.)$:

$$F_0 = H_{SF}(I_{LQ}), \tag{7}$$

With $F_0 \in R^{H\times W\times C}$, where C is the number of feature channel number. This is the method the model uses to change the input to be in the higher dimension.

Given $H_{DF}(.)$ is the module containing K residual Swin Transfomer blocks (RSTB) and a 3×3 convolutional layer. We have:

$$F_{DF} = H_{DF}(F_0) \tag{8}$$

Using image SR as an example, we use shallow and deep characteristics to rebuild the high-quality picture I_{HRQ}:

$$I_{HRQ} = H_{REC}(F_0 + F_{DF}) \tag{9}$$

Where $H_{REC}(.)$ is the function of reconstruction module. By the skip connection, SwinIR can provide low-frequency data straight to the reconstruction module, assisting the deep feature extraction module in stabilizing training and concentrating on high-frequency data.

For image SR, the loss function is L_1 pixel loss:

$$\mathcal{L} = \|I_{RHQ} - I_{HQ}\|_1, \tag{10}$$

where I_{RHQ} is obtained by taking I_{LQ} as the input of SwinIR, and I_{HQ} is the corresponding ground-truth HQ image.

3.1.2 Fast Super Resolution by Convolution Neural Network (FSRCNN)

Introduction to FSRCNN FSRCNN (Fast Super-Resolution Convolutional Neural Network) is a CNN-based approach for the Super Resolution task. It is a single neural network that takes a low-resolution (LR) image as input, "cleverly" upscales it, and returns a high-resolution (HR) image that is N times larger. N is the upscaling factor defined by the user; in this project, $N = 4$. This model represents an advancement from the previous SRCNN (Super-Resolution Convolutional Neural Network), featuring numerous improvements in performance and processing speed.

FSRCNN performs the super-resolution process without the need for a preliminary interpolation step, which helps save time and computational resources. Instead, this model operates directly on the low-resolution image, referred to as Y_s, to extract the necessary features.

The feature extraction in FSRCNN is the first and crucial step, where the network employs filters to generate high-dimensional feature vectors from batches of images. This enables the model to better understand the essential elements required to reconstruct high-quality images.

The components of FSRCNN include:

- Feature extraction
- A shrinking layer
- Non-linear mapping
- An expanding layer
- A deconvolution layer

These layers interact to produce higher-resolution images from the initial input data. The design of FSRCNN has proven effective, particularly when it comes to conserving energy and minimizing computational complexity.

Through these enhancements, FSRCNN not only improves image quality but also optimizes processing speed, making it a compelling choice for applications demanding super-resolution.

Preliminaries: SRCNN

This report focuses on FSRCNN, a refined iteration of the SRCNN (Super-Resolution Convolutional Neural Network) model. To provide context, we will first present an overview of SRCNN, outlining its core architecture and operational principles.

SRCNN is designed to learn an end-to-end mapping function f that relates the bicubic-interpolated low-resolution (LR) image Y to the corresponding high-resolution (HR) image X. Notably, the network architecture comprises only convolutional layers, ensuring that the output size is identical to the input image size. In SRCNN, the steps are as follows:

1. **Patch Extraction and Representation:** This initial layer is responsible for extracting patches from the input image and representing each patch as a high-dimensional feature vector.

2. **Non-Linear Mapping:** The middle layer performs a non-linear mapping of the feature vectors to another set of feature vectors, effectively capturing the high-resolution features.

3. **Reconstruction:** The final layer aggregates these high-dimensional features to produce the output image.

The computational complexity of the SRCNN model can be expressed as follows:

$$O\left(f_1^2 n_1 + n_1 f_2^2 n_2 + n_2 f_3^2 \cdot S_{HR}\right) \tag{11}$$

where it is linearly proportional to the size of HR image, S_{HR}. The larger the HR image, the higher the complexity.

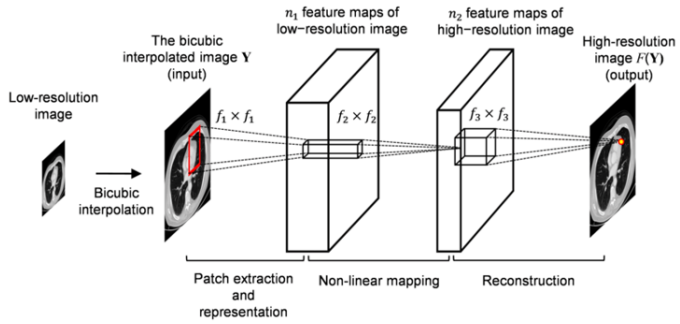

Figure 4: Network Architecture of SRCNN.

General Network Architecture of FSRCNN

The FSRCNN model is structured with five key components: feature extraction, shrinking, non-linear mapping, expanding, and deconvolution. The first four components are comprised of convolutional layers, whereas the last component functions as a deconvolution layer. For the sake of clarity, we represent a convolutional layer as `Conv(fi, ni, ci)` and a deconvolution layer as `DeConv(fi, ni, ci)`. In this notation:

- f_i: refers to the filter size,
- n_i: denotes the number of filters,
- c_i: represents the number of channels for each layer.

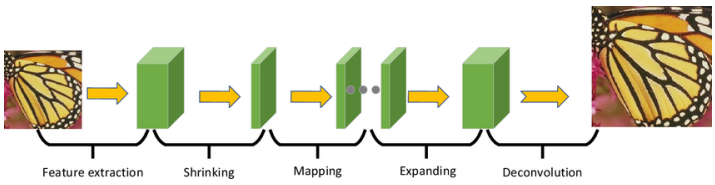

Figure 5: Network Architecture of SRCNN.

Feature Extraction.

The feature extraction stage in FSRCNN mirrors the approach used in SRCNN but applies directly to the original low-resolution (LR) input image rather than an interpolated version. FSRCNN processes the small input of the LR, indicated as Y_s, to extract features. By performing convolutions with an initial set of filters, each input patch (with a 1-pixel overlap) is converted into a high-dimensional feature vector.

In contrast to SRCNN, where the first layer's filters are applied to the upscaled image Y, FSRCNN optimizes this process by directly operating on Y_s. Since most pixels in Y are interpolated from Y_s, a 5×5 patch in Y_s encapsulates nearly all the information found in a 9×9 patch in Y. Therefore, FSRCNN reduces the filter size to $f_1 = 5$, minimizing information loss while maintaining efficiency. For the number of channels, the model retains the value $c_1 = 1$ from SRCNN.

The final step involves determining the filter count n_1, which defines the dimensionality of the LR feature space. This parameter, referred to as d, represents a key design variable in the feature extraction process. Overall, the configuration of the first layer in FSRCNN can be expressed as Conv$(5 \times d \times 1)$, reflecting a balance between computational efficiency and feature representation capacity.

Shrinking.

In this model, instead of directly mapping the high-dimensional low-resolution (LR) features to the high-resolution (HR) feature space after feature extraction, an intermediate shrinking step is introduced. This addresses the high computational complexity associated with mapping when the LR feature dimension, d, is large.

To mitigate this, a shrinking layer is added immediately following the feature extraction layer, reducing the LR feature dimension d to a smaller value s. The filter size is fixed at $f_2 = 1$, allowing the filters to act as a linear combination of the LR features. By selecting a smaller filter count, $n_2 = s \ll d$, the LR feature dimension is significantly reduced from d to s. The shrinking layer is represented as

Conv$(1 \times s \times d)$. This approach not only lowers computational complexity but also reduces the number of parameters, improving the efficiency of the model.

Non-Linear Mapping.

The non-linear mapping step plays a pivotal role in determining the performance of super-resolution (SR). Two key factors influence this step: the width (i.e., the number of filters in each layer) and the depth (i.e., the number of layers in the mapping subsection).

Based on experimental observations, using larger filters like 5×5 improves performance compared to smaller filters like 1×1. However, very deep networks have not been extensively explored in previous works. To strike a balance between performance and network complexity, FSRCNN adopts a moderate filter size of $f_3 = 3$. Instead of employing a single wide mapping layer, FSRCNN replaces it with multiple 3×3 layers.

The number of mapping layers is a critical variable, denoted as m, which directly affects both the mapping accuracy and the computational complexity. To ensure consistency, all mapping layers use the same number of filters, $n_3 = s$. Thus, the non-linear mapping step can be described as m Conv$(3 \times s \times s)$.

Expanding.

The expanding layer acts as the reverse of the shrinking layer. While the shrinking layer reduces the dimensionality of the LR features to improve computational efficiency, directly generating the high-resolution (HR) image from these low-dimensional features would degrade the reconstruction quality. Therefore, FSRCNN includes an expanding layer after the mapping step to increase the feature dimensionality back to its original size.

This layer uses 1×1 filters, consistent with the shrinking layer, and the number of filters matches the number used in the LR feature extraction layer. Unlike the shrinking layer, represented as Conv$(1 \times s \times d)$, the expanding layer is represented as Conv$(1 \times d \times s)$. Experiments show that excluding the expanding layer reduces performance by up to 0.3 dB on the Set5 dataset.

Deconvolution.

The final step is the deconvolution layer, which performs upsampling and combines the extracted features using a set of deconvolution filters. Deconvolution essentially works as the inverse of convolution. In convolution, the filter is applied to the input with a stride k, producing an output that is $\frac{1}{k}$ times the input size. Conversely, deconvolution generates an output that is k times the input size by reversing the roles of the input and output.

FSRCNN leverages this property by setting the stride $k = n$, where n is the desired upscaling factor. This allows the deconvolution layer to directly output the reconstructed HR image. The filter size of the deconvolution layer, f_5, is chosen to be 9×9, consistent with the first layer of SRCNN, as the deconvolution process is analogous to extracting features from the HR image in reverse. The

deconvolution layer is represented as DeConv$(9 \times 1 \times d)$.

Unlike traditional approaches that use fixed interpolation kernels (e.g., bicubic or bilinear) or unpooling followed by convolution, the deconvolution layer in FSRCNN learns an upsampling kernel tailored to the input features. These learned kernels are diverse and optimized for the task, significantly improving performance. Forcing these kernels to be identical would lead to inefficient parameter utilization and a drop in performance by at least 0.9 dB on the Set5 dataset.

Activation Function: PreLU.

The model uses the Parametric Rectified Linear Unit (PReLU) activation function instead of the commonly used ReLU. PReLU introduces a learnable parameter a_i for the negative part of the activation, defined as:

$$f(x_i) = \max(x_i, 0) + a_i \cdot \min(0, x_i),$$

where x_i is the input signal and a_i is specific to the i-th channel. Unlike ReLU, which sets $a_i = 0$, PReLU learns the value of a_i, addressing the issue of "dead features" caused by zero gradients in ReLU.

Cost Function.

The mean square error (MSE) is adopted as the cost function for training, similar to SRCNN. The optimization objective is formulated as:

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \|F(Y_s^i; \theta) - X^i\|_2^2,$$

where Y_s^i and X^i represent the i-th LR and HR sub-image pair, $F(Y_s^i; \theta)$ is the network output, and θ denotes the network parameters. Training is conducted using stochastic gradient descent with backpropagation.

Overall Structure and Complex Computation.

The FSRCNN model is structured into several components that work together to form a complete architecture. These include feature extraction, shrinking, non-linear mapping, expanding, and deconvolution layers. The overall network can be represented as:

$$\text{Conv}(5, d, 1) \rightarrow \text{PReLU} \rightarrow \text{Conv}(1, s, d) \rightarrow \text{PReLU} \rightarrow m \times \text{Conv}(3, s, s) \rightarrow \text{DeConv}(9, 1, d)$$

The model includes three key variables that influence both the performance and speed of the network:

1. The LR feature dimension d,
2. The number of shrinking filters s,

3. The mapping depth m.

For simplicity, the FSRCNN model is represented as `FSRCNN(d, s, m)`. The computational complexity of the model, considering these variables, is expressed as:

$$O\left(\{25d + sd + 9ms^2 + ds + 81d\}S_{LR}\right) = O\left(\{9ms^2 + 2sd + 106d\}S_{LR}\right)$$

In this equation, S_{LR} represents the size of the low-resolution image. The complexity increases with the number of filters and the feature dimension d, and the number of mapping layers m directly influences the number of operations required. This complexity analysis highlights how the performance of the FSRCNN network is highly dependent on the design choices of these parameters, making it a trade-off between accuracy and computational efficiency.

Result.

Evaluation Procedure

- **Preprocessing:** High-resolution (HR) images are downscaled using bicubic interpolation with the specified scaling factor (e.g., $\times 4$) to generate low-resolution (LR) inputs.
- **Model Inference:** The FSRCNN model processes the LR inputs to reconstruct HR images directly, bypassing the need for preliminary interpolation.
- **Metric Calculation:** Quantitative metrics such as Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) are utilized to objectively assess the visual quality of the reconstructed images. These metrics are computed on the luminance (Y-channel) of the YCbCr color space to reflect human visual perception more accurately. Additionally, the model's computational efficiency is evaluated by measuring the inference time. This metric is crucial for determining the model's suitability for practical applications, particularly in real-time or resource-constrained environments.

Test Results The FSRCNN model's performance was assessed using three benchmark datasets: Set5, Set14, and Set12. These datasets represent a mix of simple and complex image scenarios, providing a comprehensive evaluation of the model's super-resolution capabilities. To ensure consistency, the evaluation focused on the luminance (Y-channel) of the YCbCr color space, as it closely aligns with human visual perception. The model's reconstruction quality was quantified using Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM), both widely recognized metrics in image quality assessment. Additionally, the average inference time per image was recorded to evaluate computational efficiency, a critical factor for real-time and resource-constrained applications.

Below is a summary of the results across the three datasets:

Dataset	Average PSNR (dB)	Average SSIM	Avg Run Time (s)
Set5	30.50	0.8429	0.2
Set12	27.44	0.7760	0.23
Set14	27.30	0.7310	0.21

Table 1: Performance Evaluation of FSRCNN on Test Datasets.

This table summarizes the average PSNR, SSIM, and average runtime for the FSRCNN model on three test datasets (Set5, Set12, and Set14). The metrics provide insight into the model's image reconstruction quality and computational efficiency.

The results demonstrate that the FSRCNN model performs exceptionally well on Set5, a simpler dataset, achieving high PSNR and SSIM values. On more challenging datasets like Set14 and Set12, the model maintains reasonable reconstruction quality, with a slight decrease in PSNR and SSIM. Overall, these findings confirm the FSRCNN model's capability to deliver high-quality image reconstruction while maintaining a balance between performance and computational efficiency. Future optimization efforts could focus on reducing inference time to enhance the model's real-world applicability.

3.1.3 SRGAN

Introduction.

Deep generative models have traditionally faced significant computational challenges in approximating probabilistic distributions and effectively leveraging neural network architectures. Generative Adversarial Networks (GANs) emerged as an innovative solution to these fundamental limitations. The core principle of GANs lies in a unique adversarial learning framework that fundamentally transforms generative modeling.

In the GAN framework, two neural networks engage in a competitive learning process:

- A generative model (G) that aims to capture the underlying data distribution
- A discriminative model (D) designed to distinguish between authentic training data and generated samples

This adversarial mechanism creates a dynamic training environment where both networks continuously refine their capabilities. The generator strives to produce increasingly convincing synthetic data, while the discriminator becomes progressively more sophisticated in detecting generated content.

Image super-resolution (SR) represents a critical image processing challenge that addresses the reconstruction of high-resolution images from low-resolution inputs. Traditional interpolation methods often produce suboptimal results, characterized by blurriness, artifacts, and loss of fine details. The field requires advanced techniques that can intelligently enhance image resolution while preserving perceptual quality and textural information.

The primary challenges in image super-resolution include:

- Maintaining perceptual realism
- Preserving fine-grained textural details
- Minimizing artifacts and distortions
- Generating visually compelling high-resolution outputs

The Super-Resolution Generative Adversarial Network (SRGAN) represents a groundbreaking approach to addressing image super-resolution limitations. Unlike traditional methods that rely on pixel-level mean squared error (MSE) optimization, SRGAN introduces a novel perceptual loss function that prioritizes visual quality and perceptual realism.

Key innovations of SRGAN include:

- A deep residual network architecture
- An adversarial training strategy
- A perceptual loss function combining adversarial and content losses
- Ability to generate visually sophisticated high-resolution images

This research explores the implementation and performance of SRGAN, investigating its architectural innovations, training methodology, and potential applications in advanced image reconstruction tasks.

Architecture.

The SRGAN architecture comprises two primary neural networks: a deep residual network generator and a discriminator network, each designed to address specific challenges in high-quality image super-resolution.

A discriminator network D_{θ_D} is optimized with G_{θ_G} in an alternating manner to solve the adversarial min-max problem:

$$\min_{\theta_G} \max_{\theta_D} \mathbb{E}_{I^{HR} \sim p_{train}(I^{HR})}[\log D_{\theta_D}(I^{HR})] + \mathbb{E}_{I^{LR} \sim P_G(I^{LR})}[\log(1 - D_{\theta_D}(G_{\theta_G}(I^{LR})))].$$

The generator network employs a deep residual architecture optimized for image upscaling. Key characteristics include:

- Utilization of residual blocks with parametric ReLU activation
- Convolutional layers with stride and kernel configurations designed for efficient feature extraction
- Pixel shuffle layers for upsampling, replacing traditional transposed convolution
- Network depth optimized to capture multi-scale image features

The generator's primary objective is to transform low-resolution input images into high-resolution outputs that are perceptually indistinguishable from ground truth images.

The discriminator network functions as a binary classifier with the following architectural principles:

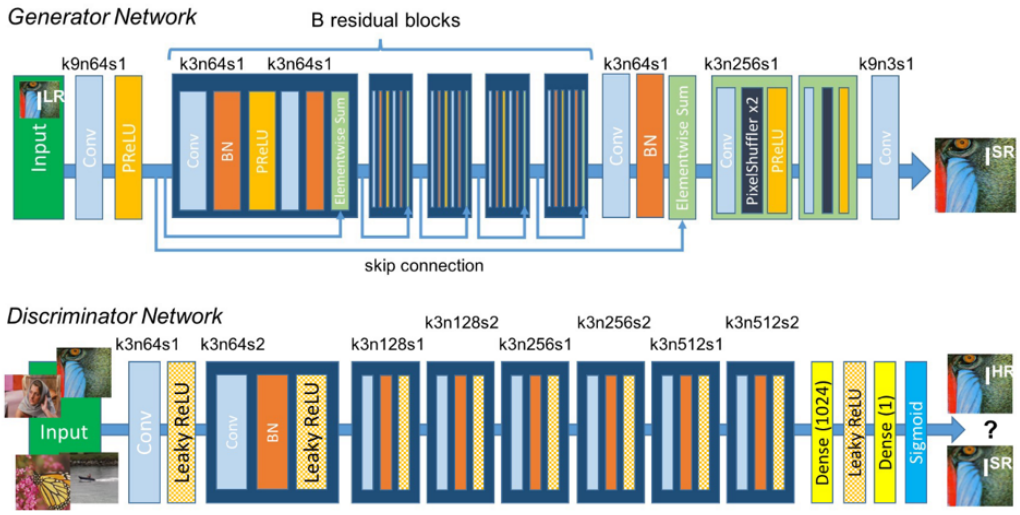

Figure 6: Architecture of Generator and Discriminator Network with corresponding kernel size (k), number of feature maps (n) and stride (s) indicated for each convolutional layer.

- Convolutional layers progressively increasing in depth
- Batch normalization to stabilize feature learning
- Leaky ReLU activation functions
- Fully connected layers for final classification

The discriminator's role is to distinguish between authentic high-resolution images and generated super-resolved images, providing critical feedback to the generator during adversarial training.

SRGAN introduces a novel perceptual loss function that fundamentally differs from traditional pixel-wise loss metrics. The loss function is formulated as:

$$l^{SR} = \underbrace{\underbrace{l_X^{SR}}_{\text{content loss}} + \underbrace{10^{-3} \cdot l_{Gen}^{SR}}_{\text{adversarial loss}} + \underbrace{2.10^{-8} \cdot l_{TV}^{SR}}_{\text{total variation loss}}}_{\text{perceptual loss (for VGG based content losses)}}$$

The content loss comprises two primary components:

- **Pixel-wise Mean Squared Error (MSE):**

$$l_{MSE}^{SR} = \frac{1}{r^2 WH} \sum_{x=1}^{rW} \sum_{y=1}^{rH} \left(I_{x,y}^{HR} - G_{\theta_G}(I^{LR})_{x,y} \right)^2 .$$

- **Perceptual loss derived from pre-trained VGG16:**

$$l_{VGG/i,j}^{SR} = \frac{1}{W_{i,j} H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} \left(\phi_{i,j}(I^{HR})_{x,y} - \phi_{i,j}(G_{\theta_G}(I^{LR})_{x,y}) \right)^2 .$$

where $\phi_{i,j}$ represents the feature map of the j^{th} convolution (after activation) before the i^{th} max-pool layer.

The perceptual loss leverages feature representations from intermediate convolutional layers, capturing high-level image characteristics beyond pixel-level differences.

The adversarial loss term, derived from the generator-discriminator interaction, encourages the generation of photorealistic images. It is formulated to maximize the probability of the generated image being classified as authentic by the discriminator, while minimizing the generative loss, defined as:

$$l_{Gen}^{SR} = \sum_{n=1}^{N} -\log D_{\theta_D}(G_{\theta_G}(I^{LR})),$$

where $D_{\theta_D}(G_{\theta_G}(I^{LR}))$ is the probability that the reconstructed image $G_{\theta_G}(I^{LR})$ is a ground truth high-resolution (HR) image. For better computation of the gradient, the term $-\log D_{\theta_D}(G_{\theta_G}(I^{LR}))$ is minimized instead of $\log[1 - D_{\theta_D}(G_{\theta_G}(I^{LR}))]$.

Total Variation Loss is a regularization technique that encourages spatial smoothness in an image by minimizing the differences between neighboring pixel intensities. It penalizes abrupt changes in pixel values, reducing noise while preserving edge structures. It is formulated as:

$$l_{TV}^{SR} = \frac{1}{HW} \sum_{i=0}^{H} \sum_{j=0}^{W} \sqrt{(x_{i+1,j} - x_{i,j})^2 + (x_{i,j+1} - x_{i,j})^2}.$$

Experiments and Evaluation.

The super-resolution model was trained with the following configuration parameters:

Parameter	**Value**	**Note**
Learning Rate	1×10^{-4}	
Training Epochs	400	Main training phase using VGG loss (enable discriminator)
Warmup Epochs	10	Initial training phase using MSE loss (disable discriminator)
Batch Size	16	
Number of Workers	4	
Upscaling Scale	4x	
HR Size	96×96	
LR Size	24×24	

Table 2: Training Configuration

3.2 Image Deblurring

3.2.1 NAFNet

Introduction.

- With the development of deep learning, the performance of image restoration methods improved significantly. Deep learning-based methods have achieved tremendous success.
- Despite their good performance, these methods suffer from high system complexity. So a nonlinear activation free network, namely NAFNet, was introduced, which matches or even surpasses the performance of the baseline despite its simplified structure. By revealing the connections between GELU, Channel Attention to Gated Linear Unit (GLU), it further simplifies the baseline by removing or replacing the nonlinear activation functions (e.g. Sigmoid, ReLU, and GELU).
- State-of-the-art (SOTA) results are achieved on various challenging benchmarks, e.g., 33.69 dB PSNR on GoPro (for image deblurring), exceeding the previous SOTA by 0.38 dB with only 8.4% of its computational costs; 40.30 dB PSNR on SIDD (for image denoising), exceeding the previous SOTA by 0.28 dB with less than half of its computational costs.

Baseline.

- Its approach focuses on simplicity, avoiding the inclusion of unnecessary components unless validated by empirical evaluations. Experiments were primarily conducted using models with a size of approximately 16 GMACs, following the HINet Simple framework, with MACs estimated for inputs of 256 $\times$ 256 resolution. Results from models with varying capacities are detailed in the experimental section. Performance validation was carried out on widely-used datasets: GoPro for deblurring as this task is foundational in low-level vision.

Architecture To reduce the inter-block complexity, the model adopts the classic single-stage U-shaped architecture with skip connections (Figure 7c). The architecture is believed not to be a barrier to performance.

A Plain Block Neural networks are built by stacking blocks. While the stacking approach has been decided (using a UNet architecture), designing the internal structure of each block remains a challenge. To address this, we begin with a plain block comprising basic components such as convolution, ReLU, and shortcut connections arranged as illustrated in Figure 8b. This design is referred to as PlainNet for simplicity.

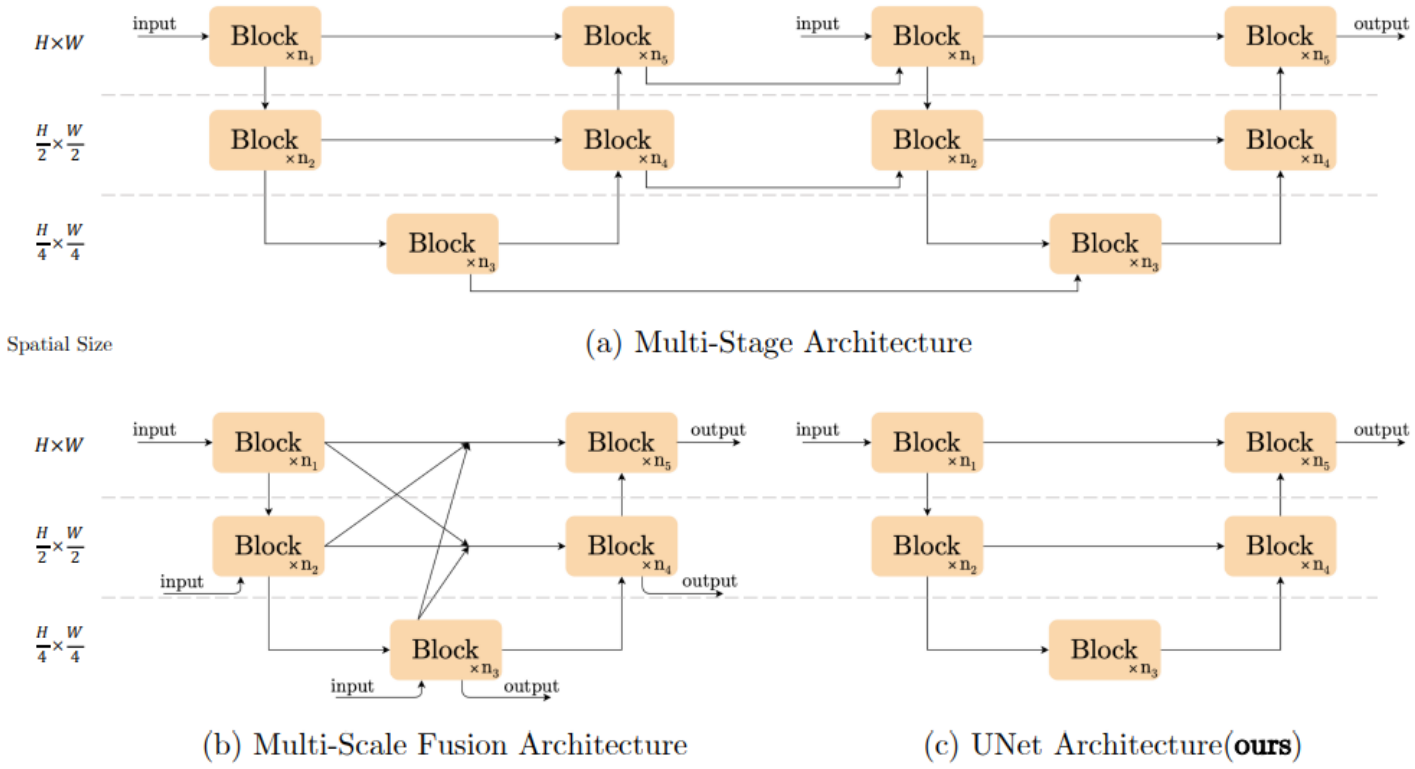

Figure 7: A comparison of the architectures used in image restoration models, with dashes used to highlight features of varying sizes. (a) The multi-stage architecture sequentially stacks UNet architectures. (b) The multi-scale fusion architecture combines features at different scales. (c) The UNet architecture is the one adopted in our approach. For simplicity, certain details are intentionally omitted, such as downsampling/upsampling layers, feature fusion modules, and input/output shortcuts.

Normalization Normalization is a common technique in high-level computer vision tasks and is increasingly used in low-level vision as well. While Batch Normalization was discarded due to instability caused by small batch sizes, Instance Normalization was reintroduced to address this issue. However, it was found that instance normalization does not consistently improve performance and often requires manual adjustments.

In contrast, the growing popularity of transformers has led to the widespread adoption of Layer Normalization in various methods, including state-of-the-art (SOTA) approaches. This observation suggests that Layer Normalization may play a crucial role in achieving SOTA performance for image restoration tasks. In summary, Layer Normalization was added to the plain block to enhance training stability and overall performance.

Activation The activation function in the plain block, Rectified Linear Unit (ReLU), is extensively used in computer vision. However, there is a tendency to replace ReLU with GELU in SOTA methods. This model follows that trend. In short, replacing ReLU with GELU in the plain block brings non-trivial

gains in image deblurring.

Non-linear Activation Free Network.

- **Gated Linear Units:** The gated linear units can be formulated as:

$$\text{Gate}(\mathbf{X}, f, g, \sigma) = f(\mathbf{X}) \odot \sigma(g(\mathbf{X})),$$

where $\mathbf{X}$ represents the feature map, f and g are linear transformers, and σ is a non-linear activation function.

- The activation function in the baseline GELU:

$$\text{GELU}(x) = x\Phi(x),$$

where Φ indicates the cumulative distribution function of the standard normal distribution. GELU can also be approximated and implemented as:

$$0.5x\left(1 + \tanh\left(\sqrt{\frac{2}{\pi}}\left(x + 0.044715x^3\right)\right)\right).$$

From these two above equations, it can be observed that GELU is a specific instance of GLU, where f and g are identity functions and σ is replaced by Φ. Drawing on this similarity, we hypothesize that GLU can be viewed as a generalized form of activation functions, potentially serving as a replacement for traditional nonlinear activation functions. Additionally, GLU inherently incorporates nonlinearity and does not strictly rely on σ: even if σ is removed, the Gate function $\text{Gate}(\mathbf{X}) = f(\mathbf{X}) \odot g(\mathbf{X})$ still introduces nonlinearity.

Building on these insights, we propose a simplified variant of GLU: we split the feature map into two parts along the channel dimension and perform element-wise multiplication, as illustrated in Figure 4c. This approach, referred to as SimpleGate, is significantly more straightforward compared to the complex implementation of GELU in Equation 3, as it only requires element-wise multiplication.

$$\textit{SimpleGate}(\mathbf{X}, \mathbf{Y}) = \mathbf{X} \odot \mathbf{Y},$$

where $\mathbf{X}$ and $\mathbf{Y}$ are feature maps of the same size.

By replacing GELU in the baseline model with the proposed SimpleGate, the performance improved for image deblurring on the GoPro dataset, achieving gains of 0.41 dB (from 32.35 dB to 32.76 dB). These results indicate that GELU can effectively be replaced by SimpleGate. At this stage, only a few nonlinear activation functions remain in the network, namely Sigmoid and ReLU, which are used in the channel attention module. The simplification of these components will be discussed next.

Simplified Channel Attention In Baseine Section, we adopt the channel attention mechanism within our block as it efficiently captures global information while remaining computationally lightweight. As shown in Figure 9a, the spatial information is first squeezed into the channel dimension, after which a multilayer perceptron (MLP) computes the channel attention. The resulting attention values are then used to weight the feature map. This process can be mathematically expressed as:

$$CA(\mathbf{X}) = \mathbf{X} * \sigma\left(W_2 \max(0, W_1 \mathsf{pool}(\mathbf{X}))\right),$$

where $\mathbf{X}$ represents the feature map, pool indicates the global average pooling operation which aggregates the spatial information into channels, and σ is a nonlinear activation function (Sigmoid). W_1, W_2 are fully-connected layers, and ReLU is adopted between the two fully-connected layers. Finally, $*$ denotes a channelwise product operation. If we regard the channel-attention calculation as a function, noted as Ψ with input $\mathbf{X}$, Equation 5 can be rewritten as:

$$CA(\mathbf{X}) = \mathbf{X} * \Psi(\mathbf{X}).$$

It can be noticed that Equation 6 is very similar to Equation 1. This inspires us to consider channel attention as a special case of GLU, which can be simplified like GLU in the previous subsection. By retaining the two most important roles of channel attention, that is, aggregating global information and channel information interaction, we propose the Simplified Channel Attention:

$$SCA(\mathbf{X}) = \mathbf{X} * W_{\mathsf{pool}}(\mathbf{X}).$$

The notations follow Equation 5. Apparently, Simplified Channel Attention (Equation 7) is simpler than the original one (Equation 5), as shown in Figures 9a and 9b. Although it is simpler, there is no loss of performance: $+$0.09 dB (32.76 dB to 32.85 dB) on GoPro.

Summary Building on the baseline introduced in Baseline Section, we simplify it further by replacing GELU with SimpleGate and Channel Attention with Simplified Channel Attention, while maintaining performance. Notably, after these modifications, the network contains no nonlinear activation functions (such as ReLU, GELU, or Sigmoid). Therefore, we refer to this streamlined version as the Nonlinear Activation Free Network (NAFNet). Despite the absence of nonlinear activations, NAFNet matches or even outperforms the original baseline. This allows us to affirmatively answer the questions posed at the start of this section, highlighting the simplicity and effectiveness of NAFNet.

3.3 DiffIR Framework

3.3.1 Introduction to DiffIR: An Efficient Diffusion Model for Image Restoration

While diffusion models (DMs) have demonstrated state-of-the-art performance in image synthesis by sequentially applying denoising networks to generating images from scratch, applying this paradigm

directly to image restoration (IR) is computationally inefficient. In IR, most pixels and structural information are already provided, making it redundant to process entire images or feature maps, which not only increases computational overhead but also introduces potential artifacts. Although DMs possess robust data estimation capabilities, their traditional synthesis approach is unsuitable for IR tasks. To overcome this limitation, DiffIR is introduced as an efficient DM-based framework for IR. By estimating a compact IR prior representation (IPR) to guide the restoration process, DiffIR significantly reduces model size and iteration requirements, achieving more precise results than conventional methods.

As shown in Figure 10, DiffIR integrates a compact IR prior extraction network (CPEN), a dynamic IR transformer (DIRformer), and a denoising network. DiffIR employs a two-stage training process: pretraining and direct DM training. During the pretraining phase, ground-truth images are input into CPENS1 to extract a compact IR prior representation (IPR), which then guides the DIRformer. In the second stage, the DM is trained to directly estimate the same IPR as CPENS1, using only low-quality (LQ) images. It is observed that, since the IPR is a compact vector, DiffIR requires fewer iterations compared to traditional DMs, resulting in more accurate estimations and stable, realistic outputs. Due to the reduced number of iterations, DiffIR enables joint optimization of CPENS2, DIRformer, and the denoising network, which further minimizes the impact of estimation errors. Extensive experiments across various IR tasks demonstrate that DiffIR achieves state-of-the-art performance with significantly lower computational costs.

3.3.2 Preliminaries: Diffusion Models

Diffusion models (DMs) are utilized in this study to generate accurate image restoration priors (IPRs). The training phase firstly incorporates a diffusion mechanism that transforms an input image x_0 into Gaussian noise $x_T \sim \mathcal{N}(0, 1)$ over T iterations. Each step in this forward process is modeled as:

$$q(x_t \mid x_{t-1}) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I\right), \tag{12}$$

where x_t denotes the noisy image at time step t, β_t is a predefined scale factor, and $\mathcal{N}$ represents a Gaussian distribution. Furthermore, the distribution of x_T can be directly conditioned on the input image x_0 as:

$$q(x_t \mid x_0) = \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I\right), \tag{13}$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=0}^{t} \alpha_i$.

During the reverse process, DMs start with the last sample x_T from the forward process and gradually denoise it to reconstruct a high-quality image x_0. The reverse process follows:

$$p(x_{t-1} \mid x_t, x_0) = \mathcal{N}\left(x_{t-1}; \mu_t(x_t, x_0), \sigma_t^2 I\right), \tag{14}$$

where the mean and variance are defined as:

$$\mu_t(x_t, x_0) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right), \quad \sigma_t^2 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t.$$

Here, ϵ represents the noise within x_t, which serves as the primary source of uncertainty in the reverse process. A denoising network $\epsilon_\theta(x_t, t)$ is employed to approximate ϵ. To train this network, clean images x_0 are perturbed using noise $\epsilon \sim \mathcal{N}(0, I)$ at a uniformly selected time step t, generating noisy images x_t based on the forward process. The model parameters θ are optimized by minimizing:

$$\mathbb{E}_{x_0,t,\epsilon}\left[\left\|\epsilon - \epsilon_\theta\left(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t\right)\right\|_2^2\right]. \tag{15}$$

In the inference stage, the DMs sample a Gaussian random noise x_T and then gradually denoise x_T T times until it reaches an output x_0 using using Eq. (14).

3.3.3 Architecture

Figure 10 illustrates the architecture of DiffIR, which consists of three primary components: a compact IR prior extraction network (CPEN), a dynamic IR transformer (DIRformer), and a denoising network. The training process of DiffIR is divided into two phases: pretraining and direct training of the diffusion model (DM), described in 3.3.3 and 3.3.3, respectively.

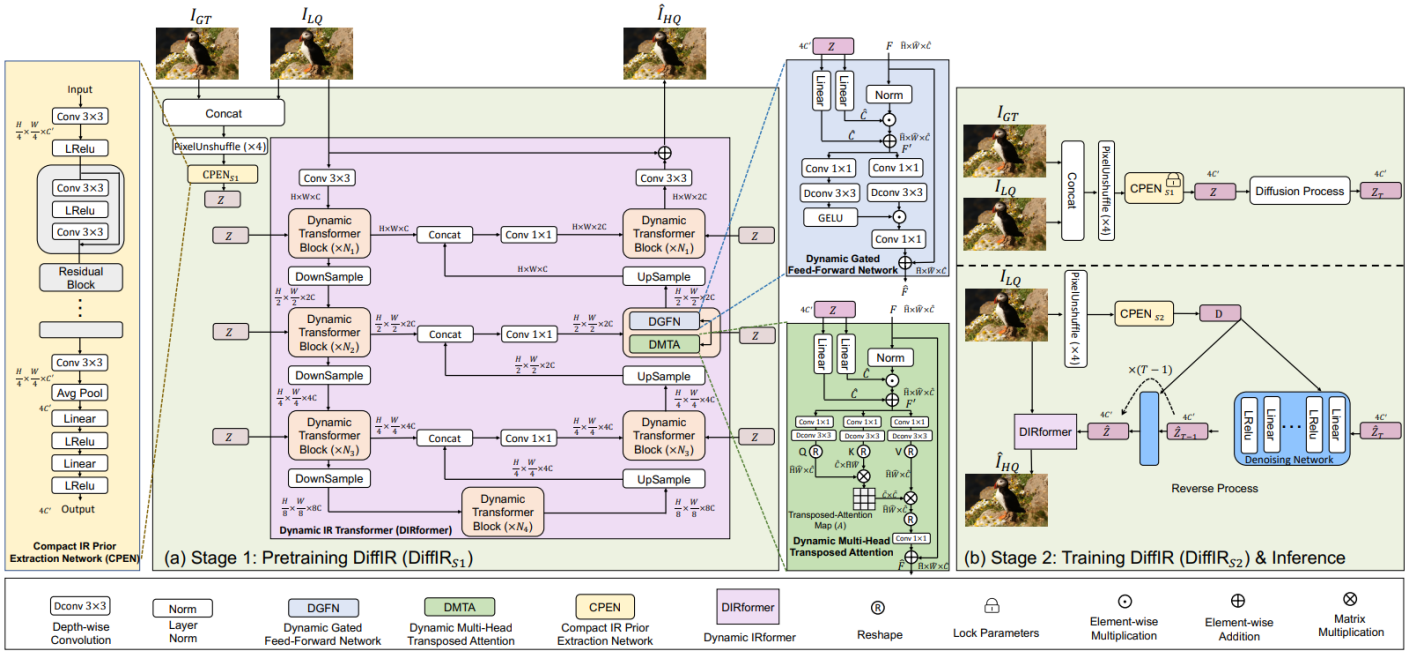

Figure 10: Overview of the proposed DiffIR system, comprising DIRformer, CPEN, and the denoising network. DiffIR operates in two training stages: (a) In the first stage, CPENS1 takes the ground-truth image as input and generates an IPR Z, which guides DIRformer in restoring images. CPENS1 is jointly optimized with DiffIRS1 to ensure the full utilization of the extracted IPR. (b) In the second stage, the Diffusion Model's (DM) strong data estimation capabilities are leveraged to estimate the IPR extracted by the pretrained CPENS1. Importantly, during this stage, the ground-truth image is not input into CPENS2 or the denoising network. During inference, only the reverse process of the DM is used.

Pretraining DiffIR. Before delving into the pretraining process, there are two critical networks in this stage: the compact IR prior extraction network (CPEN) and the dynamic IR transformer (DIRformer). The CPEN structure, highlighted in the yellow box of Figure 10, is composed of residual blocks and linear layers, which facilitate the extraction of compact IR prior representations (IPR). These representations are subsequently used by DIRformer to reconstruct the input low-quality (LQ) images. Leveraging the transformer's ability to model long-range pixel dependencies, DIRformer is designed with transformer blocks as its fundamental units. These blocks are organized in a U-Net architecture, enabling multi-level feature extraction and aggregation.

The dynamic transformer blocks of DIRformer incorporate two key components: dynamic multi-head transposed attention (DMTA, Figure 10 green box) and a dynamic gated feed-forward network (DGFN, Figure 10 blue box). These elements utilize the IPR as dynamic modulation parameters, thereby enhancing feature maps with restoration details. During the pretraining phase (Figure 10 (a)), CPENS1 and DIRformer are trained jointly. Specifically, ground-truth and LQ images are concatenated and downsampled using the PixelUnshuffle operation to generate inputs for CPENS1. The extracted IPR

$Z \in \mathbb{R}^{4C'}$ is computed as:

$$Z = \text{CPENS1}(\text{PixelUnshuffle}(\text{Concat}(I_{\text{GT}}, I_{\text{LQ}}))). \tag{16}$$

The IPR Z is passed to the dynamic transformer blocks of DIRformer as dynamic modulation parameters to guide the restoration process:

$$F' = W_1^l Z \odot \text{Norm}(F) + W_2^l Z, \tag{17}$$

where $\odot$ denotes element-wise multiplication, Norm represents layer normalization, W_l are linear layers, and F and $F' \in \mathbb{R}^{\hat{H} \times \hat{W} \times \hat{C}}$ are the input and output feature maps, respectively.

DMTA aggregates global spatial information by projecting F' into query Q, key K, and value V using depth-wise and point-wise convolutions. Then the query, the key and the value were reshaped to $\hat{Q} \in \mathbb{R}^{\hat{H}\hat{W} \times \hat{C}}$, $\hat{K} \in \mathbb{R}^{\hat{C} \times \hat{H}\hat{W}}$ and $\hat{V} \in \mathbb{R}^{\hat{H}\hat{W} \times \hat{C}}$, respectively. After reshaping, the dot product between $\hat{Q}$ and $\hat{K}$ was computed to generate a transposed attention map A of reduced size $\mathbb{R}^{\hat{C} \times \hat{C}}$, significantly improving computational efficiency compared to the regular attention map, which has the size $\mathbb{R}^{\hat{H}\hat{W} \times \hat{H}\hat{W}}$:

$$\hat{F} = W_c \hat{V} \cdot \text{Softmax}\left(\frac{\hat{K} \cdot \hat{Q}}{\gamma}\right) + F,$$

where γ is a learnable scaling parameter.

DGFN focuses on local feature aggregation, combining 1×1 Conv to collect information from different channels and 3×3 depth-wise Conv to aggregate information from neighboring pixels with a gating mechanism to enhance feature encoding:

$$\hat{F} = \text{GELU}(W_1^d W_1^c F') \odot W_2^d W_2^c F' + F. \tag{18}$$

Jointly training CPENS1 and DIRformer ensures optimal utilization of IPR for restoration. The reconstruction loss is defined as:

$$\mathcal{L}_{\text{rec}} = \|I_{\text{GT}} - \hat{I}_{\text{HQ}}\|_1, \tag{19}$$

where I_{GT} and $\hat{I}_{\text{HQ}}$ denote the ground-truth and restored high-quality images, respectively. Additional perceptual and adversarial losses may be incorporated for tasks emphasizing visual quality.

Diffusion Models for Image Restoration. In the second phase (Figure 10 (b)), the diffusion model (DM) leverages its powerful data estimation capabilities to estimate the IPR. Utilizing the pretrained CPENS1, the IPR $Z \in \mathbb{R}^{4C'}$ is extracted, followed by a forward diffusion process to sample Z_T:

$$q(Z_T|Z) = \mathcal{N}(Z_T; \sqrt{\bar{\alpha}_T} Z, (1 - \bar{\alpha}_T) I), \tag{20}$$

where $\bar{\alpha}_T$ and α_T are defined in Eqs. (12) and (13).

Traditional diffusion models (DMs) incur significant computational overhead due to the iterative nature of their processes. To address this, conventional DMs randomly sample a time-step $t \in [1, T]$ and optimize the denoising network solely for that specific time step, as outlined in Eqs. (12), (13), (14), and (15). However, this approach lacks joint training between the denoising network and the decoder (e.g., DIRformer), resulting that estimation inaccuracies from the denoising network can prevent the DIRformer from achieving its full potential.

In contrast, DiffIR begins at the T-th time step and executes all denoising iterations to generate $\hat{Z}$, which is subsequently fed into the DIRformer for joint optimization:

$$\hat{Z}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\hat{Z}_t - \epsilon \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \right), \tag{21}$$

where ϵ represents the predicted noise, and the noise is modeled by CPENS2 and the denoising network as described in Eq. (14). Unlike traditional DMs, DiffIR eliminates variance estimation, avoiding the insertion of noise, thereby enhancing the accuracy of the estimated intermediate prior representation (IPR) and improving performance.

During the reverse diffusion process, CPENS2 first extracts a conditional vector $D \in \mathbb{R}^{4C'}$ from low-quality (LQ) images:

$$D = \text{CPENS2}(\text{PixelUnshuffle}(I_{\text{LQ}})), \tag{22}$$

where CPENS2 shares the same architecture as CPENS1, except that the input dimension of its initial convolutional layer is 48, which is half of that of CPENS1. The denoising network ϵ_θ then estimates noise at each time step t as $\epsilon_\theta(\text{Concat}(\hat{Z}_t, t, D))$. The estimated noise is substituted back into Eq. (21) to compute $\hat{Z}_{t-1}$ for the subsequent iteration.

After T iterations, the final estimated IPR $\hat{Z} \in \mathbb{R}^{4C'}$ is obtained. The training process jointly optimizes CPENS2, the denoising network, and DIRformer using the loss function L_{all}:

$$L_{\text{diff}} = \frac{1}{4C'} \sum_{i=1}^{4C'} \left\| \hat{Z}^{(i)} - Z^{(i)} \right\|, \quad L_{\text{all}} = L_{\text{rec}} + L_{\text{diff}}, \tag{23}$$

where additional perceptual and adversarial losses can be incorporated into L_{all} to enhance visual quality.

During inference, only the reverse diffusion process (depicted in the lower section of Fig. 10 (b)) is utilized. CPENS2 extracts the conditional vector D from LQ images, and a Gaussian noise $\hat{Z}_T$ is randomly sampled. The denoising network employs $\hat{Z}_T$ and D to iteratively estimate $\hat{Z}$ over T iterations. Finally, DIRformer uses the estimated $\hat{Z}$ to restore the LQ images.

3.3.4 Training Setting

DiffIR was evaluated in two image restoration tasks: image super-resolution (SR) and single-image motion deblurring. The model adopts a 4-level encoder-decoder structure, with the attention heads in

DMTA configured as [1, 2, 4, 8], and channel dimensions set to [48, 96, 192, 384] across levels 1 to 4. The number of dynamic transformer blocks was adjusted according to the task, set to [13, 1, 1, 1] for SR and [3, 5, 6, 6] for deblurring. The CPEN channel dimension C' was set to 64.

The diffusion model was trained with $T = 4$ total timesteps, with β_t increasing linearly from 0.1 to 0.99. Using Adam optimizer ($\beta_1 = 0.9$, $\beta_2 = 0.99$), the joint optimization approach outperformed traditional DM optimization by iterating through all denoising steps, allowing better integration with DIRformer. Performance comparisons indicated that DiffIR achieves convergence with only 3–4 iterations, significantly faster than traditional DMs, which require over 200 iterations.

4 Experiments

4.1 Experimental Setup

The experimental training and validation of RestoraVision is conducted using diverse datasets tailored to the respective tasks. **Image Super-Resolution.**

To train and validate the performance of the image super-resolution models, we utilized two widely used datasets: **DIV2K** and **Flickr2K**. These datasets provide high-quality ground truth images at multiple resolutions, ensuring a robust benchmark for super-resolution tasks.

DIV2K Dataset The DIVerse 2K (DIV2K) dataset is a standard benchmark dataset designed for image super-resolution and enhancement tasks. It contains **800 high-resolution images**, spanning a variety of scenes, objects, and textures. The images are provided at a native resolution of **2K** (2048 pixels on the longer side), which ensures a high level of detail. The dataset is divided into three subsets: **800 training images**, **100 validation images**, and **100 test images**.

To facilitate super-resolution model training and validation, the DIV2K dataset is commonly downsampled to create **two low-resolution versions** corresponding to 2× and 4× scaling factors. These low-resolution images are used as inputs to the super-resolution models, while the original high-resolution images serve as the ground truth. The dataset is widely appreciated for its diversity in terms of content and texture complexity, making it a critical benchmark in the field of super-resolution.

Flickr2K Dataset The Flickr2K dataset complements DIV2K by providing a larger number of high-resolution images, specifically **2650 high-resolution images** sourced from Flickr. Like DIV2K, the images in Flickr2K are varied in content, featuring natural landscapes, urban scenes, and intricate textures. The dataset supports multiple scaling factors (e.g., 2× and 4×), enabling the creation of low-resolution inputs for model training.

One of the key advantages of the Flickr2K dataset is its size, which allows for better generalization during training by introducing more diverse textures and scenes. When used alongside DIV2K, it enriches the training process by exposing models to a broader range of visual features and complex patterns,

thus improving performance on unseen data.

Both datasets provide a reliable foundation for training and validating super-resolution models, ensuring that the trained networks are capable of producing visually appealing and high-quality images across different resolutions and scenarios.

To benchmark the performance of our models, including SwinIR, FSRCNN, SRGAN, DiffIR, and DiffIRS2, we utilized two well-known datasets, Set14 and Urban100.

Set14: The Set14 dataset is a widely used benchmark dataset in image super-resolution research. It contains 14 diverse images that cover a variety of content types, such as urban scenes, natural landscapes, and textual content. The images in Set14 have high-quality ground truths and are commonly used to evaluate the generalization ability of SR models across different scenarios. The dataset's diversity ensures that models are tested on images with varying textures, frequencies, and structures, making it a robust measure of SR performance. Metrics such as Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) are typically used to quantify the performance of SR models on this dataset.

Urban100: The Urban100 dataset is specifically designed for evaluating image super-resolution models in urban environments. It consists of 100 high-resolution images that capture a wide range of architectural and structural details commonly found in urban settings. The dataset emphasizes challenges such as repetitive patterns, fine-grained textures, and sharp edges, which are critical in assessing the ability of SR models to handle high-frequency details. Urban100 is particularly valuable for benchmarking as it highlights the performance gaps between different SR models in scenarios requiring precise reconstruction of complex structures. The dataset has become a standard for testing advanced SR algorithms due to its unique focus on intricate urban details.

By employing these two datasets, we ensure a comprehensive evaluation of our super-resolution models across diverse scenarios, ranging from general-purpose content in Set14 to highly detailed urban environments in Urban100. This dual-dataset approach provides a balanced assessment of both generalization and specificity in SR performance.

Image Deblurring. For the image deblurring task, we utilized the **GoPro** dataset, a widely recognized benchmark designed to train and evaluate the performance of deblurring models under realistic motion blur scenarios.

GoPro Dataset The GoPro dataset consists of **3214 pairs of blurry and sharp images** captured in real-world settings using a GoPro Hero4 camera. It is specifically curated to simulate motion blur resulting from camera shake and object motion, making it an ideal dataset for training and evaluating deblurring algorithms. The dataset includes diverse scenes such as indoor environments, outdoor landscapes, and dynamic objects, ensuring a comprehensive representation of motion blur challenges.

Each image pair in the dataset contains a blurry image and its corresponding sharp ground truth. The blurry images are generated by averaging consecutive frames from high-frame-rate video sequences

(240 fps), while the sharp images represent a single high-quality frame. The dataset is divided into **2103 image pairs for training** and **1111 image pairs for testing**, enabling a standardized traning and evaluation protocol.

The GoPro dataset has become a cornerstone for training and benchmarking image deblurring models due to its realism and diversity. By capturing motion blur under natural conditions, it allows models to generalize effectively to real-world deblurring tasks. In our experiments, it served as a robust foundation for training and evaluating the deblurring performance of NAFNet and DiffIR.

Training Protocols: To enhance the generalization capabilities of our models, we employ several data augmentation techniques during the training phase. These include random cropping, horizontal and vertical flipping, and random rotations. Random cropping extracts smaller patches from high-resolution images, helping models learn localized features and improving robustness to spatial variability. Flip and rotation augmentations introduce invariance to geometric transformations, enabling the models to generalize better to unseen data. These augmentations are applied probabilistically to diversify the training data and mitigate overfitting.

Evaluation Metrics: For evaluating the quality of restored images, we utilize multiple quantitative metrics tailored to specific tasks:

- **Peak Signal-to-Noise Ratio (PSNR):** PSNR measures the pixel-wise fidelity between the restored image $I_{restored}$ and the ground truth I_{ground_truth}. It is computed as:

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{MAX^2}{\text{MSE}} \right), \tag{24}$$

  where MAX is the maximum possible pixel value (e.g., 255 for 8-bit images) and MSE is the mean squared error between $I_{restored}$ and I_{ground_truth}. Higher PSNR values indicate better restoration fidelity.

- **Structural Similarity Index Measure (SSIM):** SSIM evaluates the structural similarity between $I_{restored}$ and I_{ground_truth}, considering luminance, contrast, and structure. It is defined as:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \tag{25}$$

  where μ_x and μ_y are the mean intensities, σ_x^2 and σ_y^2 are the variances, and σ_{xy} is the covariance of x and y. Constants C_1 and C_2 stabilize the division. SSIM ranges from 0 to 1, with higher values indicating better structural similarity.

- **Learned Perceptual Image Patch Similarity (LPIPS):** LPIPS assesses perceptual quality by comparing deep feature activations from a pretrained neural network (e.g., VGG). Unlike PSNR and SSIM, LPIPS focuses on perceptual similarity rather than pixel-wise accuracy. Lower LPIPS scores indicate higher perceptual quality. The metric is computed as:

$$\text{LPIPS}(x, y) = \sum_l \frac{1}{H_l W_l} \sum_{h,w} ||\phi_l(x)_{h,w} - \phi_l(y)_{h,w}||_2^2, \tag{26}$$

where ϕ_l represents the feature map at layer l of the pretrained network, and H_l, W_l denote the spatial dimensions of the feature map.

For the task of deblurring, PSNR and SSIM are used to quantify restoration fidelity and structural similarity, respectively. For single image super-resolution, we employ PSNR and LPIPS to measure both pixel-wise accuracy and perceptual realism, ensuring a comprehensive evaluation of the models' performance.

4.2 Results on Image Super-Resolution

Table 3: Quantitative comparison for **Single Image Super-Resolution** on Set14 and Urban100 datasets. Best and second-best performances are marked in bold and underlined, respectively.

Method	**Set14**		**Urban100**	
	PSNR ↑	LPIPS ↓	PSNR ↑	LPIPS ↓
SRGAN	26.84	0.1327	24.41	0.1439
DiffIRS2	**27.73**	<u>0.1117</u>	<u>26.05</u>	**0.1007**
SwinIR	<u>27.62</u>	**0.1103**	**27.56**	<u>0.1052</u>
FSRCNN	25.98	0.1397	24.92	0.1409

Table 3 presents the quantitative results of different super-resolution methods on the Set14 and Urban100 benchmark datasets. The evaluation metrics include PSNR for pixel-wise fidelity and LPIPS for perceptual quality.

On the Set14 dataset, **DiffIRS2** achieves the highest PSNR of **27.73**, demonstrating its strong ability to recover fine details. However, SwinIR achieves the lowest LPIPS value of **0.1103**, indicating superior perceptual quality. This suggests that while DiffIRS2 excels at reconstructing pixel-wise accuracy, SwinIR provides images that are more visually appealing.

For the Urban100 dataset, **SwinIR** outperforms all other methods with the highest PSNR of **27.56** and the second-best LPIPS value of <u>0.1052</u>. This highlights SwinIR's effectiveness in handling complex urban textures and high-frequency details. DiffIRS2 also performs competitively with a PSNR of **26.05** and LPIPS of **0.1007**, further proving its robustness in challenging scenarios.

In comparison, SRGAN and FSRCNN achieve relatively lower performance across both datasets. FSRCNN's lower PSNR and higher LPIPS values suggest limitations in reconstructing fine textures, while SRGAN, though better perceptually, still falls short in terms of pixel-wise fidelity.

Overall, SwinIR demonstrates the best balance between fidelity and perceptual quality, particularly on the Urban100 dataset, making it a highly effective solution for real-world applications where both accuracy and visual appeal are essential. Figure 12 shows the results of our models.

4.3 Results on Image Deblurring

Table 4: Quantitative comparison for **Image Deblurring** using PSNR and SSIM metrics.

Method	PSNR ↑	SSIM ↑
DiffIRS2	**33.69**	33.20
NAFNet	0.967	**0.963**

Table 4 presents the quantitative results for the image deblurring task, evaluated using PSNR and SSIM metrics.

DiffIRS2 achieves the highest PSNR value of **33.69**, demonstrating its superior ability to recover sharp and high-quality details from blurred images. However, NAFNet achieves the highest SSIM score of **0.963**, reflecting its strong performance in preserving structural and perceptual information within the deblurred images.

While DiffIRS2 excels in terms of pixel-wise fidelity (PSNR), NAFNet's higher SSIM indicates its effectiveness in maintaining overall visual coherence and fine structures. The close results between the two methods highlight their complementary strengths: DiffIRS2 for sharpness and NAFNet for perceptual quality.

Overall, both methods deliver strong performance on the deblurring task, with DiffIRS2 leading in fidelity and NAFNet excelling in structural similarity, making them suitable for different real-world applications depending on the desired restoration priorities. Figure 13 shows the results of models.

5 Conclusion and Future Work

RestoraVision represents a significant advancement in the field of image restoration, combining specialized models with the innovative DiffIR framework. By integrating state-of-the-art methods such as SwinIR, DiffIRS2, and NAFNet, RestoraVision achieves superior results across both super-resolution and deblurring tasks. The experimental results on benchmark datasets, including Set14, Urban100, and others, demonstrate the system's ability to deliver high-fidelity and perceptually appealing image restorations. SwinIR emerges as particularly effective for super-resolution tasks on complex urban textures, while DiffIRS2 and NAFNet complement each other in delivering sharpness and structural similarity for deblurring.

Looking ahead, there are several promising directions to expand RestoraVision's capabilities. First, we aim to extend the system to tackle video restoration tasks, addressing temporal consistency challenges in addition to spatial fidelity. Second, we plan to explore lightweight architectures that optimize computational efficiency for real-time applications, making RestoraVision more accessible for deployment

on edge devices and mobile platforms. Lastly, incorporating unsupervised and self-supervised learning paradigms could further enhance the system's robustness to diverse and unseen data.

By addressing these challenges, RestoraVision is poised to establish itself as a comprehensive and scalable solution for a wide range of image and video restoration applications, bridging the gap between research advancements and practical, real-world deployment.

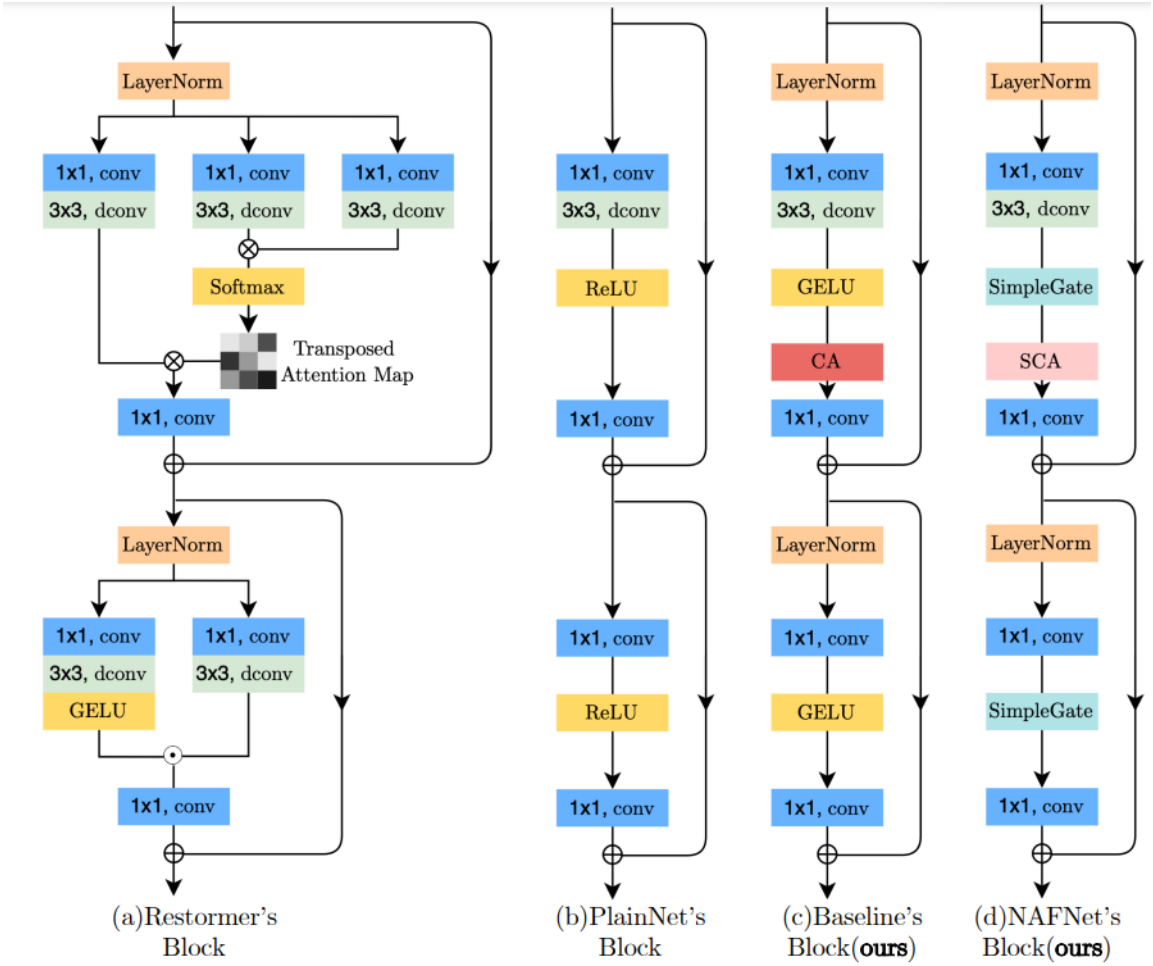

Figure 8: **Intra-block structure comparison.** $\otimes$: matrix multiplication, $\odot/\oplus$: elementwise multiplication/addition. dconv: Depthwise convolution. Nonlinear activation functions are represented by yellow boxes. (a) The block in Restormer, where some details are omitted for simplicity, such as reshaping the feature maps. (b) The block in PlainNet, which includes the most commonly used components. (c) Our proposed baseline, which, compared to (b), incorporates Channel Attention (CA) and LayerNorm, and replaces ReLU with GELU. (d) The block in our proposed Nonlinear Activation Free Network, where CA and GELU are replaced with Simplified Channel Attention (SCA) and SimpleGate, respectively.

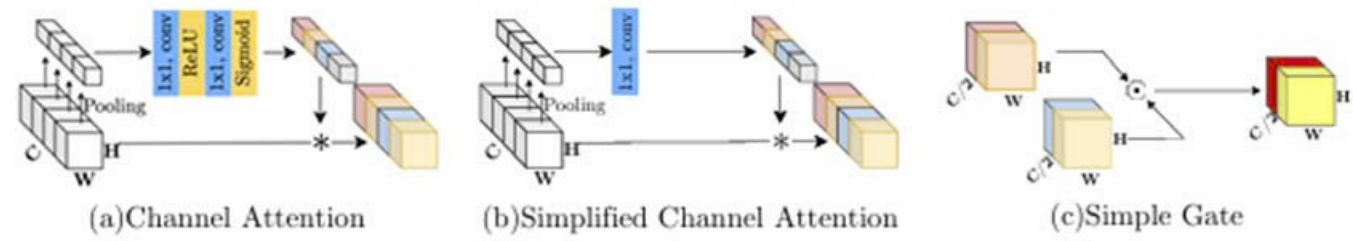

Figure 9: **Illustration of** $\otimes$: (a) Channel Attention[16] (CA), (b) Simplified Channel Attention (SCA), and (c) Simple Gate (SG).

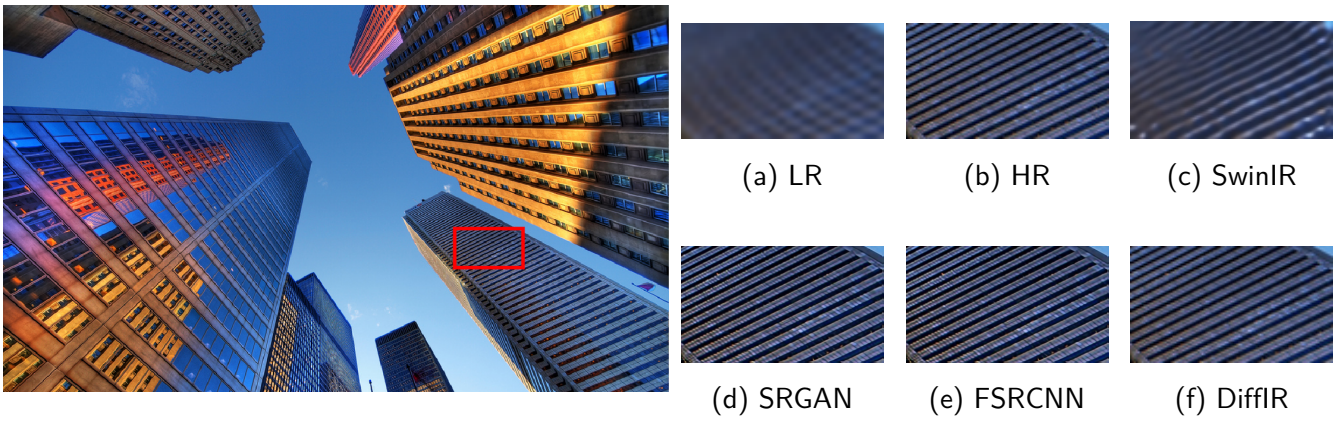

Figure 11: Visualizing the SR models.

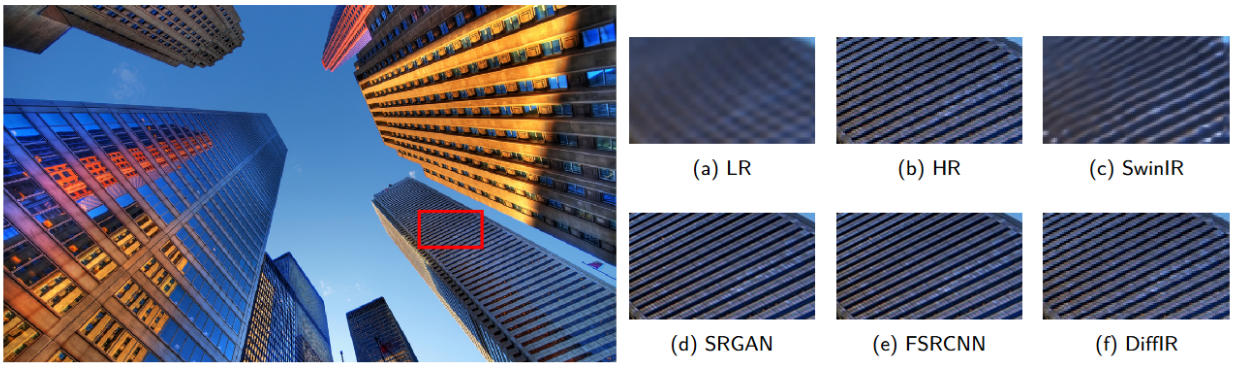

Figure 12: Visualizing the super-resolution models

(a) Low Quality (b) DiffIR (c) NAFNet

Figure 13: Visualizing the deblurring model

References

[1] B. Xia, Y. Zhang, S. Wang, Y. Wang, X. Wu, Y. Tian, W. Yang, and L. V. Gool, "Diffir: Efficient diffusion model for image restoration," 2023.

[2] C. Dong, C. C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," 2016.

[3] L. Chen, X. Chu, X. Zhang, and J. Sun, "Simple baselines for image restoration," 2022.

[4] J. Liang, J. Cao, G. Sun, K. Zhang, L. V. Gool, and R. Timofte, "Swinir: Image restoration using swin transformer," 2021.

[5] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," 2017.

[6] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, "Image super-resolution using very deep residual channel attention networks," 2018.

[7] X. Wang, L. Xie, C. Dong, and Y. Shan, "Real-esrgan: Training real-world blind super-resolution with pure synthetic data," 2021.

[8] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning a single convolutional super-resolution network for multiple degradations," 2018.

[9] S. Nah, T. H. Kim, and K. M. Lee, "Deep multi-scale convolutional neural network for dynamic scene deblurring," 2017.

[10] O. Kupyn, T. Martyniuk, J. Wu, and Z. Wang, "Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better," 2019.

[11] A. Zamir, A. Arora, S. Khan, F. S. Khan, and M.-H. Yang, "Restormer: Efficient transformer for high-resolution image restoration," 2023.

[12] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. C. Loy, "Esrgan: Enhanced super-resolution generative adversarial networks," 2018.

[13] S.-J. Cho, J. Lee, Y. Jo, S.-W. Kim, J. Oh, and M. Kim, "Rethinking data augmentation for image super-resolution: A comprehensive analysis and a new strategy," 2021.

[14] K. Zhang, W. Zuo, and L. Zhang, "Deep plug-and-play super-resolution for arbitrary blur kernels," 2019.

[15] J. M. Park, J. W. Lee, S. Cho, and S. Lee, "Multi-resolution fusion networks for image restoration," 2020.

[16] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," 2014.

[17] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," 2017.