Denoising and Super Resolution

Choeun Kim, Kwanho Lee

June 5, 2026

IDEA, Department of Statistics, Seoul National University

Table of Contents

Denoising

Problem definition

- We formulate image denoising as the problem of estimating an unknown clean image from a noisy observation:

$$f_0(x, y) = f(x, y) + \epsilon(x, y), \qquad (x, y) \in \Omega, \quad \epsilon \sim N(0, \sigma^2)$$

- Here, $\Omega \subset \mathbb{R}^2$ denotes the image domain, $f_0(x, y), f(x, y)$ are observed noisy pixel value and unknown clean pixel value at (x, y), respectively.
- The goal is to estimate the clean image by solving

$$\hat{f} = \arg\min_f R(f) \quad \text{subject to} \quad \sum_{x,y} (f(x, y) - f_0(x, y))^2 = \sigma^2 \qquad (1)$$

- $R(f)$ encodes prior assumptions on clean images.

ROF

Nonlinear Total Variation Based Noise Removal Algorithms, Physica D 1992

ROF:Definition for Clean image

- Pixel values change rapidly both around noise and around object boundaries. If we penalize gradients too strongly, object edges are also smoothed out.

$$\int |\nabla f| dx dy$$

$$|\nabla f| := \sqrt{f_x^2 + f_y^2}$$

where u_x and u_y denotes horizontal and verical image gradient, respectively.

ROF Problem Setting

- Let $\Omega \subset \mathbb{R}^2$ be the image domain. The observed noisy image is modeled as

$$f_0(x, y) = f(x, y) + \epsilon(x, y), \qquad (x, y) \in \Omega,$$

where $f_0 : \Omega \to \mathbb{R}$ is the pixel value of observed noisy image, $f : \Omega \to \mathbb{R}$ is the pixel value of unknown clean image, and $n : \Omega \to \mathbb{R}$ is additive noise which mean zero and variance σ^2.

- The ROF model estimates u by solving the constrained total variation minimization problem

$$f^* \in \arg\min_f \int_\Omega |\nabla f(x, y)| \, dxdy$$

subject to

$$\int_\Omega f(x, y) \, dxdy = \int_\Omega f_0(x, y) \, dxdy,$$

and

$$\frac{1}{2} \int_\Omega \big(f(x, y) - f_0(x, y)\big)^2 \, dxdy = \sigma^2.$$

ROF Optimization: Lagrangian Formulation

- Define the total variance:

$$TV(f) = \int_\Omega |\nabla f| \, dxdy.$$

- Introduce the Lagrangian

$$\mathcal{L}(f, \lambda_1, \lambda_2) = \int_\Omega |\nabla f| \, dxdy + \lambda_1 \left(\int_\Omega f \, dxdy - \int_\Omega f_0 \, dxdy \right) + \lambda_2 \left(\frac{1}{2} \int_\Omega (f - f_0)^2 \, dxdy - \sigma^2 \right).$$

- Taking the first variation with respect to u gives

$$\frac{\partial TV(f)}{\partial f} = \frac{dTV(f + \epsilon\phi)}{d\epsilon} = -\operatorname{div}\left(\frac{\nabla f}{|\nabla f|} \right).$$

- Therefore, the Euler–Lagrange equation is

$$0 = \operatorname{div}\left(\frac{\nabla f}{|\nabla f|} \right) - \lambda_1 - \lambda_2 (f - f_0), \qquad \text{in } \Omega,$$

with the boundary condition

$$\frac{\Delta f}{\Delta \epsilon} = 0 \qquad \text{on } \partial\Omega.$$

ROF Optimization II: Time-dependent nonlinear PDE

- Instead of solving the Euler–Lagrange equation directly, ROF introduces an artificial time variable t.

$$f(x, y, 0) = f^{(0)}(x, y),$$
$$f(x, y, t) \longrightarrow \widehat{f}(x, y) \quad \text{as } t \to \infty.$$

  Here, $f^{(0)}$ is an initial estimate chosen to satisfy the noise constraints.
- This can be understood from gradient descent:

$$f^{n+1} = f^n - \eta \nabla_u L(f^n) \tag{2}$$
$$\frac{f^{n+1} - f^n}{\eta} = -\nabla_f L(f^n) \tag{3}$$
$$\frac{df}{dt} = -\nabla_f L(f^t) \tag{4}$$

- In ROF, the gradient descent direction becomes

$$\frac{du}{dt} = \operatorname{div}\left(\frac{\nabla u^t}{|\nabla u^t|}\right) - \lambda^t (u^t - u_0)$$

- To find the stable state satisfying $\frac{df}{dt} = 0$, we update f iteratively:

$$f^{t+1} = f^t + \Delta t \left[\operatorname{div}\left(\frac{\nabla f^t}{|\nabla f^t|}\right) - \lambda^t (f^t - f_0)\right]$$

BM3D

Image Denoising by Sparse 3D Transform-Domain Collaborative Filtering, IEEE TIP 2007

BM3D: Patch-Based Estimation

- Image denoising can be formulated as

$$f_0(x, y) = f(x, y) + \epsilon(x, y)$$

- BM3D does not estimate function on Ω. But it estimates patch independently:

$$R_{x,y}(f_0) = R_{x,y}(f) + R_{x,y}(\epsilon)$$

  where $R_{x,y}$ extracts an $N \times N$ patch centered at location (x, y).

BM3D: Transform f

- Instead of estimating the clean patch directly in the pixel domain, we represent it using a fixed transform basis.

$$R_{x,y}(f) = \sum_{k=1}^{N^2} \theta_{x,y,k}\phi_k, \quad R_{x,y}(f_0) = \sum_{k=1}^{N^2} \theta_{0,x,y,k}\phi_k$$

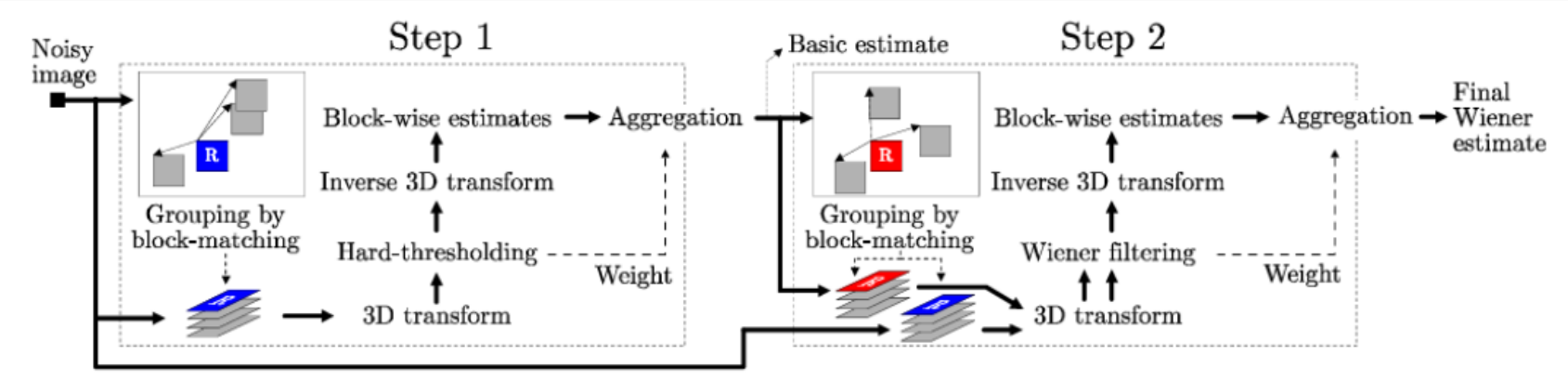
  where $\{\phi_k\}_{k=1}^{N^2}$ is a fixed basis such as low and high frequency basis function.

- Therefore, denoising can be viewed as estimating the clean transform coefficients $\boldsymbol{\theta}_{x,y,k}$ from the $\boldsymbol{\theta}_{0,x,y,k}$:

$$\arg\min_{\boldsymbol{\theta}} |\boldsymbol{\theta}_0 - \boldsymbol{\theta}|_2^2 + \lambda|\boldsymbol{\theta}|_0$$

- And it is well known that clean image patches are often sparse or concentrated in a few large coefficients.

BM3D



BM3D

- **Grouping & 3D tranform**: to solve problem that how can we get representative coefficient for certain patch.
- Why we need **Step2**

BM3D: A Toy Example of 3D Transform

1. Group two similar patches

$$P_1 = \begin{pmatrix} 11 & 9 \\ 1 & -1 \end{pmatrix}, \qquad P_2 = \begin{pmatrix} 9 & 11 \\ -1 & 1 \end{pmatrix}$$

Use the following 2×2 spatial basis patterns:

$$\phi_1 = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, \qquad \phi_2 = \frac{1}{2}\begin{pmatrix} 1 & -1 \\ 1 & -1 \end{pmatrix}$$

$$\phi_3 = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ -1 & -1 \end{pmatrix}, \qquad \phi_4 = \frac{1}{2}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

Each patch can be written as a linear combination:

$$P_1 = 10\phi_1 + 2\phi_2 + 10\phi_3 + 0\phi_4$$

$$P_2 = 10\phi_1 - 2\phi_2 + 10\phi_3 + 0\phi_4$$

$$\hat{\theta}_{P_1} = \begin{pmatrix} 10 & 2 \\ 10 & 0 \end{pmatrix}, \qquad \hat{\theta}_{P_2} = \begin{pmatrix} 10 & -2 \\ 10 & 0 \end{pmatrix}$$

2. Transform coefficients along the group direction

Use two basis vectors across the two patches:

$$\psi_1 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ 1 \end{pmatrix}, \psi_2 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

For each spatial basis, collect coefficients across patches:

$$\begin{pmatrix} 10 \\ 10 \end{pmatrix} = 10\sqrt{2}\,\psi_1 + 0\psi_2$$

$$\begin{pmatrix} 2 \\ -2 \end{pmatrix} = 0\,\psi_1 + 2\sqrt{2}\,\psi_2$$

$$\begin{pmatrix} 10 \\ 10 \end{pmatrix} = 10\sqrt{2}\,\psi_1 + 0\psi_2$$

$$\hat{\theta}_{\text{common}} = \begin{pmatrix} 10\sqrt{2} & 0 \\ 10\sqrt{2} & 0 \end{pmatrix}, \hat{\theta}_{\text{diff}} = \begin{pmatrix} 0 & 2\sqrt{2} \\ 0 & 0 \end{pmatrix}$$

where θ_{common} represents spatial patterns shared by both patches, while θ_{diff} represents spatial-pattern differences between the two patches.

$$[\hat{\theta}_{\text{common}}, \hat{\theta}_{\text{diff}}] = T_{3D}(P_1)$$

BM3D: Details for Step 2

- From the same matched patch group, compute 3D-transform coefficients:

$$\boldsymbol{\theta}^0_{(x,y)} = T_{3D}(R_{(x,y)}(f_0)) \qquad \boldsymbol{\theta}^1_{(x,y)} = T_{3D}(R_{(x,y)}(f_1))$$

- Since $f_0 = f + \epsilon$, each 3D-transform coefficient satisfies

$$\theta_{0,(x,y),j} = \theta_{(x,y),j} + \epsilon_{(x,y),j}, \qquad \epsilon_{(x,y),j} \sim \mathcal{N}(0, \sigma^2),$$

  where

$$\theta^1_{(x,y),j} \approx \theta_{(x,y),j}$$

- We want to shrink each noisy coefficient:

$$\widehat{\theta}^{(2)}_{(x,y),j} = \beta_{(x,y),j}\theta_{0,(x,y),j}$$

- Minimize the MSE only over the noise:

$$\beta^*_{(x,y),j} = \arg\min_{\beta} \mathbb{E}_\epsilon \left[\left(\theta_{(x,y),j} - \beta\theta_{0,(x,y),j} \right)^2 \right] = \frac{\left|\theta_{(x,y),j}\right|^2}{\left|\theta_{(x,y),j}\right|^2 + \sigma^2}.$$

- Since $\theta_{(x,y),j}$ is unknown, use the Step 1 coefficient:

$$\widehat{\theta}^{(2)}_{(x,y),j} = \frac{\left|\theta^{(1)}_{(x,y),j}\right|^2}{\left|\theta^{(1)}_{(x,y),j}\right|^2 + \sigma^2}\theta_{0,(x,y),j}$$

DnCNN

Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising, IEEE TIP 2017

DnCNN: Learning the Residual Mapping

- Classical denoising methods rely on hand-crafted image priors:

$$\hat{f} = \arg\min_{f} \{D(f - f_0) + \lambda R(f)\}$$

  Here, $R(f)$ encodes prior assumptions on clean images.
- DnCNN does not explicitly specify a regularizer $R(f)$. Instead, it learns a residual mapping from noisy-clean image pairs.
- Under the noise model

$$f_0 = f + \epsilon$$

- DnCNN trains a CNN G_θ to estimate this residual:

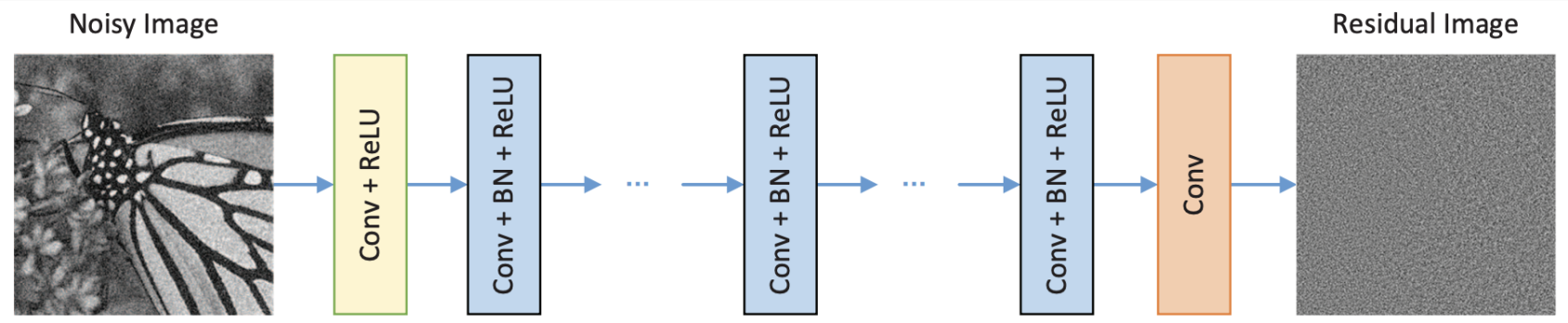
$$G_\theta(f_0) \approx f_0 - f$$

- The training objective is

$$\theta^\star = \arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \|G_\theta(f_{0,i}) - (f_{0,i} - f_i)\|_F^2$$

  where $\{(f_{0,i}, f_i)\}$ represents N noisy-clean training image pairs.

DnCNN: Residual Learning for Image Denoising



Architecture of DnCNN

- **Residual Learning**: DnCNN can be used to learn the full mapping

$$Q_\theta(f_{0,i}) \approx f_i$$

$$Q_\theta(f_{0,i}) \approx f_{0,i} - \text{small correction from noise}$$

  $Q_\theta(f_{0,i})$ would be close to identity mapping.

- **Effective for unknown noise levels:** By training on multiple noise levels, DnCNN learns a noise-level-adaptive residual estimator rather than a denoiser specialized to a fixed σ.

Connection between G_θ and $R(f)$

- Classical prior-based denoising can be written as

$$E(f; f_0) = \frac{1}{2}\|f - f_0\|_2^2 + \lambda R(f)$$

- A gradient-descent update from the current estimate f^t is

$$f^{t+1} = f^t - \eta \nabla_f E(f; f_0)\big|_{f=f^t}.$$

  Since

$$\nabla_f E(f; f_0) = (f - f_0) + \lambda \nabla R(f),$$

  we have

$$f^{t+1} = f^t - \eta \left[(f^t - f_0) + \lambda \nabla R(f^t)\right].$$

- If we initialize with the noisy image $f^0 = f_0$, then

$$f^1 = f_0 - \eta\lambda \nabla R(f_0),$$

- In this sense, residual learning can be viewed as a learned generalization of a one-step prior-driven denoising update:

$$G_\theta(f_0) \approx f_0 - f, \qquad \hat{f} = f_0 - G_\theta(f_0).$$

DAE and Score Matching

A Connection Between Score Matching and Denoising Autoencoders, Neural Computation 2011

AE and DAE

- Autoencoder learns a reconstruction map:

$$r_\theta(x) = g_\theta(f_\theta(x)), \qquad \min_\theta \mathbb{E}_{x \sim p_{\text{data}}} \|r_\theta(x) - x\|^2.$$

- Denoising autoencoder corrupts the input:

$$\tilde{x} = x + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2 I).$$

- DAE learns to recover clean data from corrupted data:

$$\min_\theta \mathbb{E}_{x,\tilde{x}} \|r_\theta(\tilde{x}) - x\|^2$$

$$r_\theta(\tilde{x}) := W^T \operatorname{sigmoid}(W\tilde{x} + b) + c$$

Score Matching and Denoising Score Matching

- The score of a density is
$$\nabla_x \log p(x)$$
- Score matching learns a vector field close to the true score:
$$\min_\theta \mathbb{E}_{x\sim p_{\text{data}}} \|\psi_\theta(x) - \nabla_x \log p_{\text{data}}(x)\|^2$$
- Denoising score matching uses corrupted samples:
$$\tilde{x} = x + \epsilon, \qquad q_\sigma(\tilde{x} \mid x) = \mathcal{N}(\tilde{x}; x, \sigma^2 I)$$
- Since
$$\nabla_{\tilde{x}} \log q_\sigma(\tilde{x} \mid x) = \frac{x - \tilde{x}}{\sigma^2},$$
DSM trains
$$\min_\theta \mathbb{E}_{x,\tilde{x}} \left\| \psi_\theta(\tilde{x}) - \frac{x - \tilde{x}}{\sigma^2} \right\|^2$$

DAE ≈ DSM

- DAE objective:

$$J_{DAE}(\theta) := \min_{\theta} \mathbb{E}_{x,\tilde{x}} \|r_\theta(\tilde{x}) - x\|^2$$

- Rewrite the DSM objective:

$$J_{DSM}(\theta) := \left[\frac{1}{2}\|\psi(\tilde{x};\theta) - \frac{\partial \log q_\sigma(\tilde{x}|x)}{\partial \tilde{x}}\|^2\right]$$

put $\psi(\tilde{x};\theta) = r_\theta(\tilde{x}) - \tilde{x}$

$$= \left[\frac{1}{2}\|\frac{1}{\sigma^2}(W^T \operatorname{sigmoid}(W\tilde{x} + b) + c - \tilde{x}) - \frac{\partial \log q_\sigma(\tilde{x}|x)}{\partial \tilde{x}}\|^2\right]$$

$$= \frac{1}{2\sigma^4}\left[\|(W^T \operatorname{sigmoid}(W\tilde{x} + b) + c) - x\|^2\right]$$

$$= \frac{1}{2\sigma^4} J_{DAE}(\theta)$$

From DAE to Residual Learning

- We've get :

$$r_\theta(\tilde{x}) - \tilde{x} \propto \nabla_{\tilde{x}} \log p(\tilde{x})$$

- Define the residual predictor:

$$G_\theta(\tilde{x}) \approx \tilde{x} - r_\theta(\tilde{x}) \propto -\nabla_{\tilde{x}} \log q_\sigma(\tilde{x})$$

- Therefore, DnCNN does not move the image in the negative-score direction. Instead, it predicts a **residual proportional to the negative score**, which represents the noise component to be removed.

Super-Resolution

Introduction

- Let $\mathbf{Y} \in \mathbb{R}^{c \times h \times w}$ be a low-resolution image and $\mathbf{X} \in \mathbb{R}^{c \times H \times W}$ be a corresponding high-resolution image. $(h < H, w < W)$
- That is,

$$\mathbf{Y} = (k \otimes \mathbf{X})_{\downarrow s} + n,$$

  where k is a blurry kernel, $\downarrow_s$ is a downsampling operator and n is the independent noise.
- The goal is to find $\mathbf{X}$ from the observed image $\mathbf{Y}$.
- This is an ill-posed problem, since there can be more than one HR images corresponding to a LR image.

Image Super-Resolution via Sparse Representation

- Assume the following:

$$\mathbf{Y} = SH\mathbf{X}, \tag{5}$$

  where H represents a blurring filter and S the down-sampling operator.

- Basic Idea : Consider the image patch $\mathbf{x} \in \mathbb{R}^{c \times H' \times W'}$ and the patch dictionary $\mathbf{D} \in \mathbb{R}^{c \times H' \times W' \times K}$. Assume that the image patch $\mathbf{x} \approx \mathbf{D}\boldsymbol{\alpha}$, where $\boldsymbol{\alpha} \in \mathbb{R}^{K}$.
- Now consider two dictionaries – $\mathbf{D}_h$ for HR images and $\mathbf{D}_l$ for LR images.

- Note that as the dictionary $\mathbf{D}_h$ becomes overcomplete, there may exist many $\boldsymbol{\alpha}$ satisfying $\mathbf{x} = \mathbf{D}_h\boldsymbol{\alpha}$.
- Thus, the authors aim to find the sparsest solution.

$$\mathbf{x} \approx \mathbf{D}_h\boldsymbol{\alpha} \text{ for some } \boldsymbol{\alpha} \in \mathbb{R}^K \text{ with } \|\boldsymbol{\alpha}\|_0 \ll K \tag{6}$$

Image Super-Resolution via Sparse Representation

Local model from sparse representation

- Two dictionaries ($\mathbf{D}_l$, $\mathbf{D}_h$) are trained to have the same sparse representations for each HR and LR image patch pair.

$$\mathbf{x} = \mathbf{D}_h \boldsymbol{\alpha}_h, \mathbf{y} = \mathbf{D}_l \boldsymbol{\alpha}_l, \boldsymbol{\alpha}_h \approx \boldsymbol{\alpha}_l$$

- Inference Process
  - For each input LR patch $\mathbf{y}$, find a sparse representation w.r.t. $\mathbf{D}_l$.
  - The corresponding HR patch $\mathbf{x}$ is generated using the coefficients and $\mathbf{D}_h$.

Image Super-Resolution via Sparse Representation

- Finding the sparsest representation of $\mathbf{y}$

$$\min \|\boldsymbol{\alpha}\|_0 \text{ s.t. } \|F\mathbf{D}_l\boldsymbol{\alpha} - F\mathbf{y}\|_2^2 \leq \epsilon \tag{7}$$

  where F is a (linear) feature extraction operator.

$$\min \|\boldsymbol{\alpha}\|_1 \text{ s.t. } \|F\mathbf{D}_l\boldsymbol{\alpha} - F\mathbf{y}\|_2^2 \leq \epsilon \tag{8}$$

  Solving (8) for each local patch does not gaurantee the **compatibility between adjacent patches.**

- The resulting optimization problem is

$$\begin{aligned} \min \|\boldsymbol{\alpha}\|_1 \text{ s.t. } & \|F\mathbf{D}_l\boldsymbol{\alpha} - F\mathbf{y}\|_2^2 \leq \epsilon, \\ & \|P\mathbf{D}_h\boldsymbol{\alpha} - \boldsymbol{\omega}\|_2^2 \leq \epsilon, \end{aligned} \tag{9}$$

  where the matrix P extracts the region of overlap between the current target patch and previously reconstructed HR image, and $\boldsymbol{\omega}$ contains the values of the previously reconstructed HR image on the overlap.

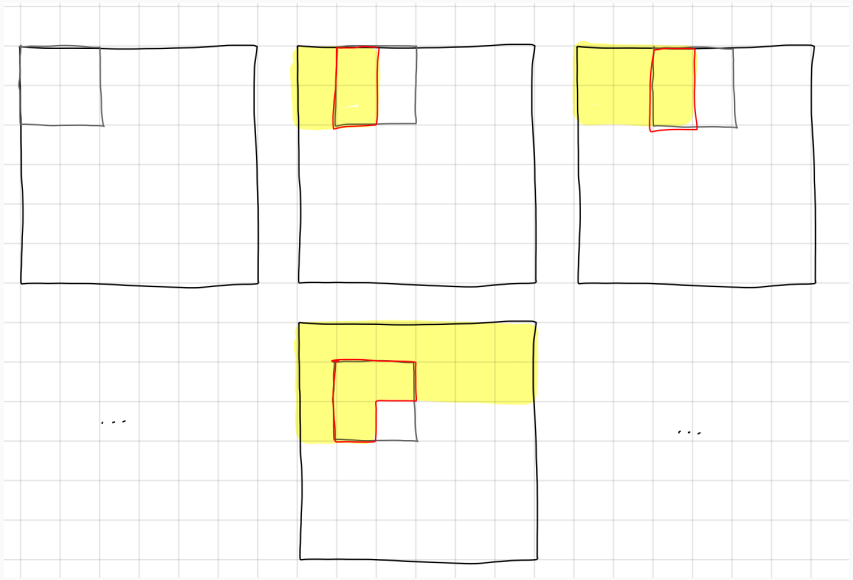


Illustration of the patch overlap constraint described in Eq. (9)

- The constrained optimization (9) can be similarly reformulated as:

$$\min_{\boldsymbol{\alpha}} \|\tilde{\mathbf{D}}\boldsymbol{\alpha} - \tilde{\mathbf{y}}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1, \tag{10}$$

where $\tilde{\mathbf{D}} = \begin{bmatrix} F\mathbf{D}_l \\ P\mathbf{D}_h \end{bmatrix}$ and $\tilde{\mathbf{y}} = \begin{bmatrix} F\mathbf{y} \\ \boldsymbol{\omega} \end{bmatrix}$.

Note that the HR image $\mathbf{X}_0$ produced by the sparse representation approach may not satisfy the reconstruction constraint (5) exactly.

- Eliminate this discrepancy by projecting $\mathbf{X}_0$ onto the solution space of $SH\mathbf{X} = \mathbf{Y}$ by computing

$$\mathbf{X}^* = arg\min_{\mathbf{X}} \|SH\mathbf{X} - \mathbf{Y}\|_2^2 + c\|\mathbf{X} - \mathbf{X}_0\|_2^2 \tag{11}$$

Image Super-Resolution via Sparse Representation

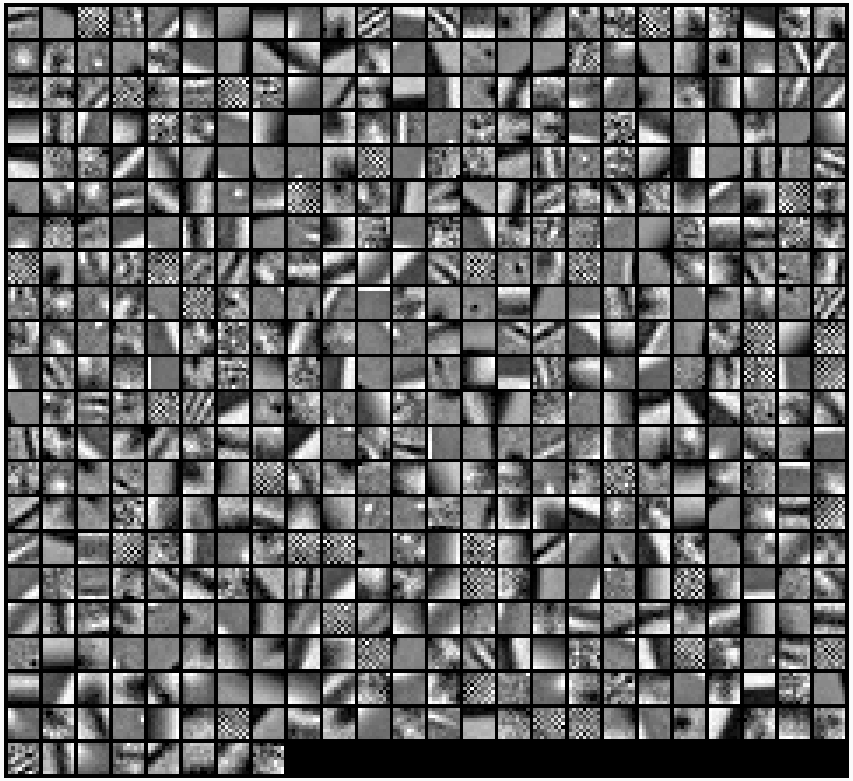
- Joint Dictionary Training

  Let $X^h = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ be the vector combined with a set of sampled HR image patches and $Y^l = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ be that of a set of sampled LR image patches.

$$\min_{\{\mathbf{D}_h, \mathbf{D}_l, Z\}} \frac{1}{N}\|X^h - \mathbf{D}_h Z\|_2^2 + \frac{1}{M}\|Y^l - \mathbf{D}_l Z\|_2^2 + \lambda \left(\frac{1}{N} + \frac{1}{M}\right)\|Z\|_1,$$

  where $\mathbf{D}_h \in \mathbb{R}^{c \times H' \times W' \times K}, \mathbf{D}_l \in \mathbb{R}^{c \times h' \times w' \times K}, N = cH'W', M = ch'w',$ and $Z \in \mathbb{R}^{K \times n}$.

Image Super-Resolution via Sparse Representation



The HR image patch dictionary

SRGAN

- Generator : generate a super-resolved image given input LR image, $\mathbf{Y}$
- Discriminator : discriminate whether a given input image is HR or SR
- Objective Function
  - Generator

$$L_G = L_{VGG} + 10^{-3} L_{adv},$$
$$L_{VGG} = \|\phi(\mathbf{X}) - \phi(G(\mathbf{Y}))\|_2^2, \;\; L_{adv} = -\mathbb{E}_{\mathbf{Y}}[\log D(G(\mathbf{Y}))] \tag{12}$$

    where ϕ is VGG19 network (image encoder)

  - Discriminator

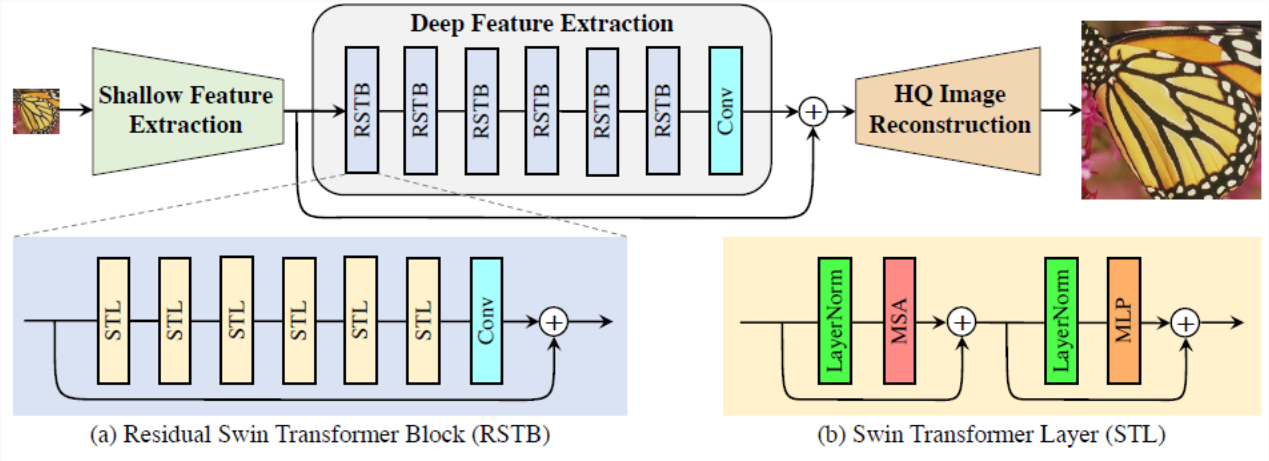
$$L_D = -\mathbb{E}_{\mathbf{X}}[\log D(\mathbf{X})] - \mathbb{E}_{\mathbf{Y}}[\log(1 - D(G(\mathbf{Y}))] \tag{13}$$

SwinIR

SwinIR: Image Restoration using Swin Transformer, ICCVW 2021

- Most CNN methods are not effective for long-range dependency modelling.
- As an alternative to CNN, Transformer designs a self-attention mechanism to capture global interactions. However, ViT usually divides the input image into small patches with fixed size and processes each patch independently. Using the strategy, the reconstruction result may introduce border artifacts.
- Network Architecture
  - Shallow feature extraction
  - Deep feature extraction
  - High-quality image reconstruction

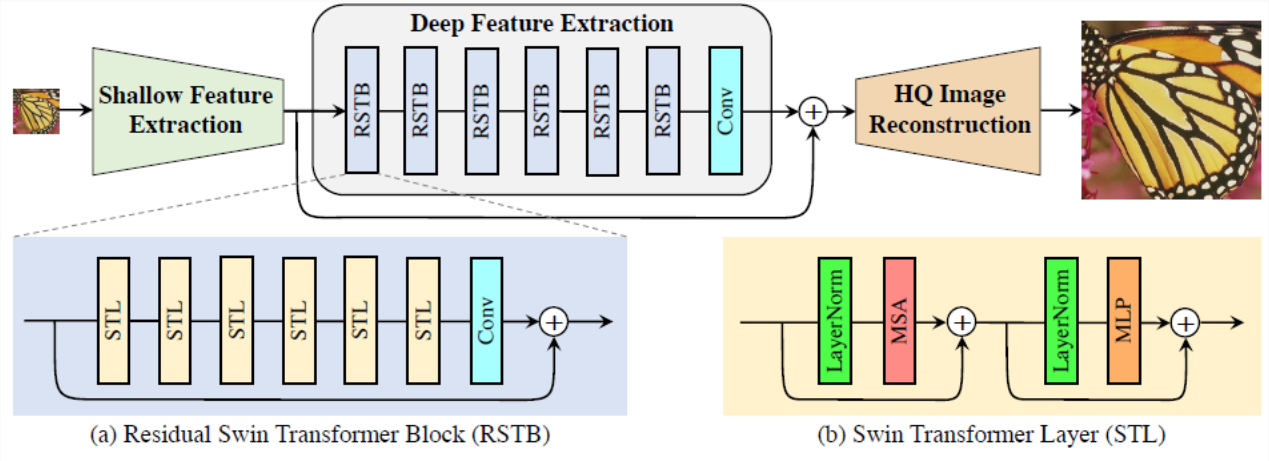
SwinIR



SwinIR architecture

- Shallow feature extraction
  - Input : LR image $\mathbf{Y} \in \mathbb{R}^{c \times h \times w}$
  - 3×3 convolutional layer $H_{SF}(\cdot)$
  - Output : Shallow feature $F_0 = H_{SF}(\mathbf{Y}) \in \mathbb{R}^{C \times h \times w}$

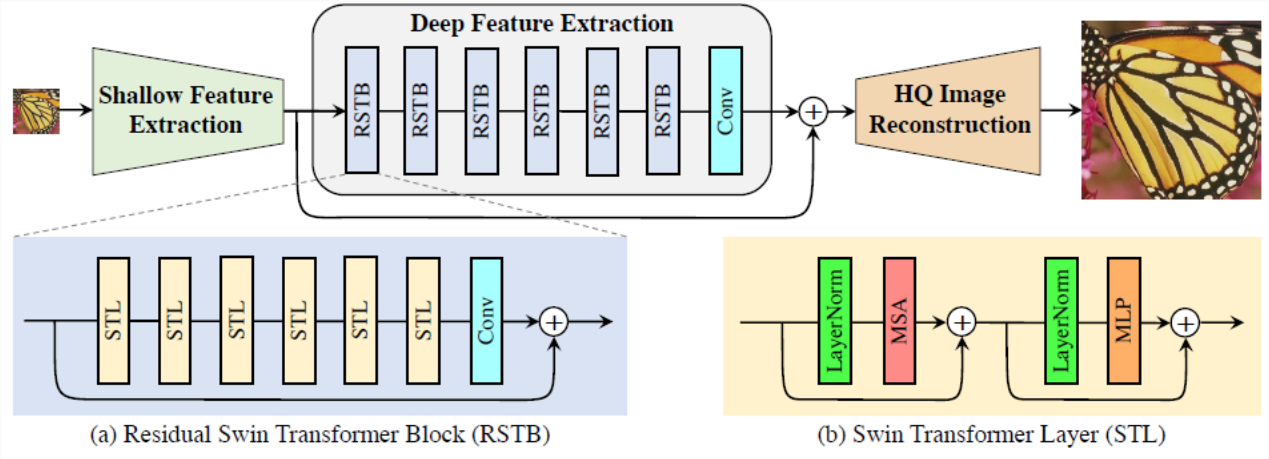
SwinIR



SwinIR architecture

- Deep feature extraction
  - Input : Shallow feature F_0
  - K Residual Swin Transformer Blocks (RSTB) and a 3×3 convolutional layer
  - Output : Deep feature
    $F_{DF} = H_{conv}(H_{RSTB_K}(H_{RSTB_{K-1}}(\cdots(H_{RSTB_1}(F_0))))) \in \mathbb{R}^{C\times h\times w}$

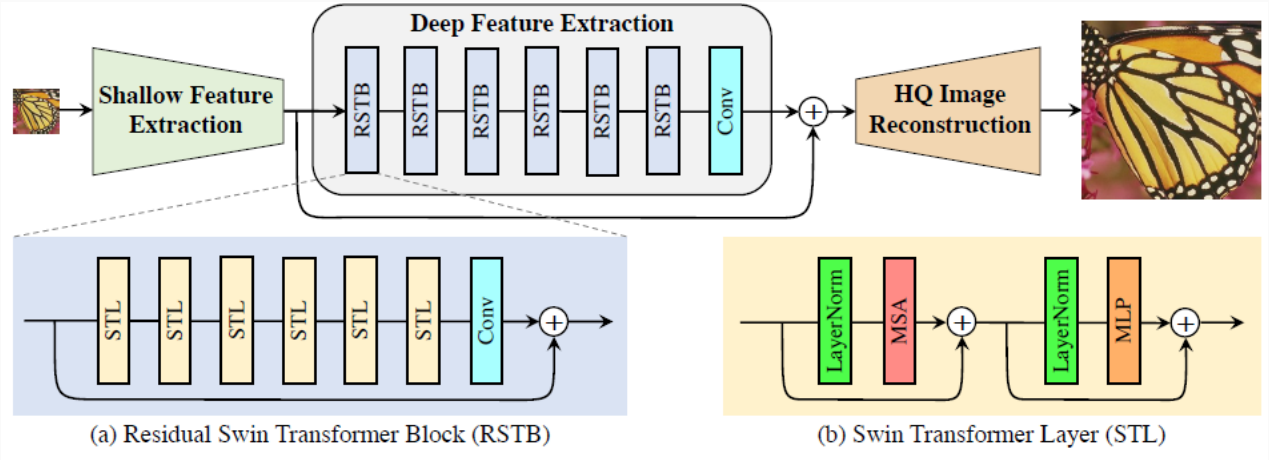
SwinIR



SwinIR architecture

- High-quality image reconstruction
  - Input : $F_0 + F_{DF} \in \mathbb{R}^{C \times h \times w}$
  - Sub-pixel convolutional layer :
    $(C, h, w) \rightarrow (cr^2, h, w) \rightarrow (c, hr, wr) = (c, H, W)$
  - Output : $\hat{\mathbf{X}} \in \mathbb{R}^{c \times H \times W}$

SwinIR

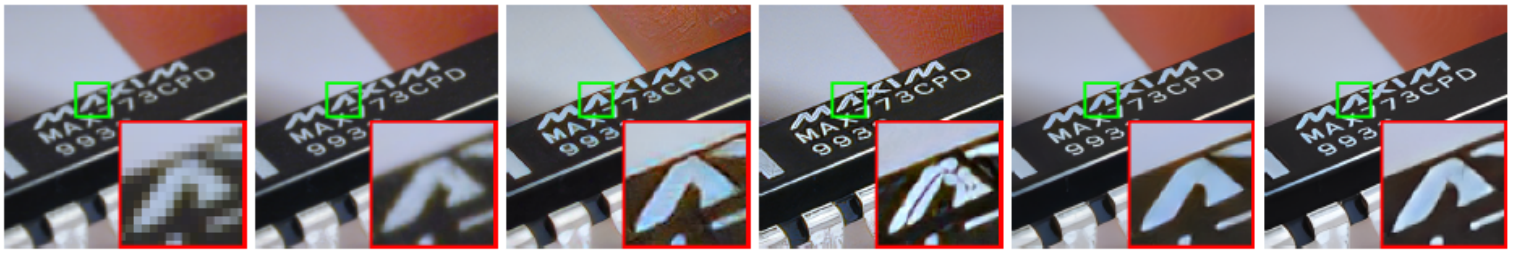


SwinIR architecture

- Final output : $\hat{\mathbf{X}}^* = \hat{\mathbf{X}} + \text{upsample}(\mathbf{Y})$
- Objective function:

$$\mathcal{L} = \|\hat{\mathbf{X}}^* - \mathbf{X}\|_1$$

SwinIR



Visual comparison of real-world SR ($\times 4$) methods on RealSRSet

Others

Other works..

One Diffusion Step to Real-World Super-Resolution via Flow Trajectory Distillation (FluxSR), ICML 2025

InvFusion: Bridging Supervised and Zero-shot Diffusion for Inverse Problems, NeurIPS 2025

Chain-of-Zoom: Extreme Super-Resolution via Scale Autoregression and Preference Alignment, NeurIPS 2025

KernelFusion: Zero-Shot Blind Super-Resolution via Patch Diffusion, ICLR 2026

New Idea

Idea

- Quantized Image Denoising with Discrete Diffusion models
- Explicit prior definition using TV prior in ROF