# 2DQuant: Low-bit Post-Training Quantization for Image Super-Resolution

**Kai Liu**[1], **Haotong Qin**[2], **Yong Guo**[3], **Xin Yuan**[4],
**Linghe Kong**[1*], **Guihai Chen**[1], **Yulun Zhang**[1*]
[1]Shanghai Jiao Tong University, [2]ETH Zürich,
[3]Max Planck Institute for Informatics, [4]Westlake University

## Appendix / Supplemental material

## A Detailed structure of SwinIR

SwinIR comprises three core modules: shallow feature extraction, deep feature extraction, and high-quality (HQ) image reconstruction.

**Shallow and deep feature extraction.** Given a low-quality (LQ) input $I_{LQ} \in \mathbb{R}^{H \times W \times C_{in}}$ (where $H$, $W$, and $C_{in}$ represent the image height, width, and input channel number, respectively), a $3 \times 3$ convolutional layer $H_{SF}(\cdot)$ is employed to extract shallow features $F_0 \in \mathbb{R}^{H \times W \times C}$ as follows:

$$F_0 = H_{SF}(I_{LQ}), \tag{1}$$

where $C$ denotes the number of feature channels. Subsequently, deep features $F_{DF} \in \mathbb{R}^{H \times W \times C}$ are extracted from $F_0$ as:

$$F_{DF} = H_{DF}(F_0), \tag{2}$$

where $H_{DF}(\cdot)$ represents the deep feature extraction module, comprising $K$ residual Swin Transformer blocks (RSTB) and a $3 \times 3$ convolutional layer. Specifically, intermediate features $F_1, F_2, \ldots, F_K$ and the output deep feature $F_{DF}$ are sequentially extracted as follows:

$$\begin{gathered} F_i = H_{RSTB_i}(F_{i-1}), \quad i = 1, 2, \ldots, K, \\ F_{DF} = H_{CONV}(F_K), \end{gathered} \tag{3}$$

where $H_{RSTB_i}(\cdot)$ denotes the $i$-th RSTB, and $H_{CONV}$ is the concluding convolutional layer. Incorporating a convolutional layer at the end of feature extraction introduces the inductive bias of the convolution operation into the Transformer-based network, laying a robust foundation for subsequent aggregation of shallow and deep features.

**Image reconstruction.** In the context of image super-resolution (SR), the high-quality image $I_{RHQ}$ is reconstructed by combining shallow and deep features as follows:

$$I_{RHQ} = H_{REC}(F_0 + F_{DF}), \tag{4}$$

where $H_{REC}(\cdot)$ is the reconstruction module's function. The reconstruction module is implemented using a sub-pixel convolution layer to upsample the feature.Additionally, residual learning is utilized to reconstruct the residual between the LQ and HQ images instead of the HQ image itself, formulated as:

$$I_{RHQ} = H_{SwinIR}(I_{LQ}) + I_{LQ}, \tag{5}$$

where $H_{SwinIR}(\cdot)$ represents the SwinIR function.

[*]Corresponding authors: Yulun Zhang, yulun100@gmail.com, Linghe Kong, linghe.kong@sjtu.edu.cn

### A.1 Residual Swin Transformer block

The residual Swin Transformer block (RSTB) is a residual block incorporating Swin Transformer layers (STL) and convolutional layers. Given the input feature $F_{i,0}$ of the $i$-th RSTB, intermediate features $F_{i,1}, F_{i,2}, \ldots, F_{i,L}$ are first extracted by $L$ Swin Transformer layers as follows:

$$F_{i,j} = H_{STL_{i,j}}(F_{i,j-1}), \quad j = 1, 2, \ldots, L, \tag{6}$$

where $H_{STL_{i,j}}(\cdot)$ is the $j$-th Swin Transformer layer in the $i$-th RSTB. A convolutional layer is added before the residual connection, and the output of RSTB is formulated as:

$$F_{i,out} = H_{CONV_i}(F_{i,L}) + F_{i,0}, \tag{7}$$

where $H_{CONV_i}(\cdot)$ is the convolutional layer in the $i$-th RSTB.
**Swin Transformer layer.** Given an input of size $H \times W \times C$, the Swin Transformer first reshapes the input into a $\frac{HW}{M^2} \times M^2 \times C$ feature by partitioning the input into non-overlapping $M \times M$ local windows, where $\frac{HW}{M^2}$ is the total number of windows. It then computes the standard self-attention for each window (i.e., local attention). For a local window feature $X \in \mathbb{R}^{M^2 \times C}$, the *query*, *key*, and *value* matrices $Q$, $K$, and $V$ are computed as follows:

$$Q = XP_Q, \quad K = XP_K, \quad V = XP_V, \tag{8}$$

where $P_Q$, $P_K$, and $P_V$ are projection matrices shared across different windows. Typically, $Q, K, V \in \mathbb{R}^{M^2 \times d}$. The attention matrix is then computed via the self-attention mechanism within a local window as follows:

$$\text{Attention}(Q, K, V) = \text{SoftMax}(QK^T/\sqrt{d} + B)V, \tag{9}$$

where $B$ is the learnable relative positional encoding. In practice, the attention function is performed $h$ times in parallel, and the results are concatenated for multi-head self-attention (MSA).

Next, a multi-layer perceptron (MLP) with two fully-connected layers and GELU non-linearity between them is used for further feature transformations. The LayerNorm (LN) layer is added before both MSA and MLP, with residual connections employed for both modules. The entire process is formulated as:

$$\begin{aligned} X &= \text{MSA}(\text{LN}(X)) + X, \\ X &= \text{MLP}(\text{LN}(X)) + X. \end{aligned} \tag{10}$$

However, when the partition is fixed across different layers, there are no connections between local windows. Thus, regular and shifted window partitioning are used alternately to enable cross-window connections **?**, with shifted window partitioning involving shifting the feature by $(\lfloor \frac{M}{2} \rfloor, \lfloor \frac{M}{2} \rfloor)$ pixels before partitioning.

### A.2 Our settings

We use the SwinIR light version provided by the original authors. The light version has only 4 RSTBs in the body part while for each RSTB, there are only 6 STLs. For each STL's MSA, the number of heads is 6, the embedding dimension is 60, the window size is 8, and the MLP ratio is 2.

## B Detailed distribution of weights and activations

In code implementation, the RSTB is called layers while the STL is called blocks. We visualize all layers' distribution of the pre-trained SwinIR light model's weights in Figure 1. Bias is ignored as it is not quantized. Also, we visualize the distribution of activations from 32 image patches with a size of 3×64×64 in Figure 2, Figure 3, Figure 4, and Figure 5.

We can safely ignore the detailed value of each axis but just care about the shape of distributions.

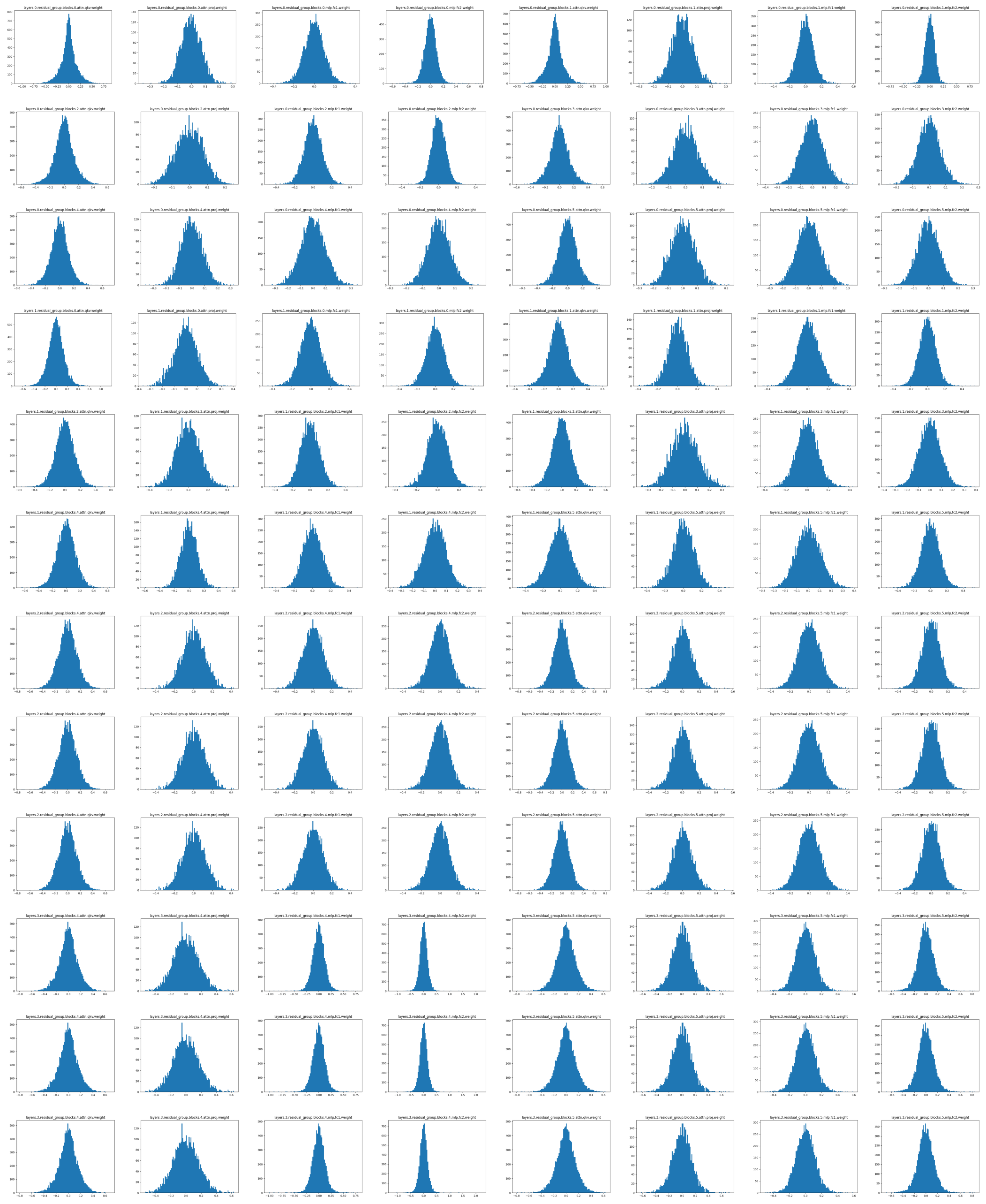
Figure 1: Visualization of SwinIR weights.

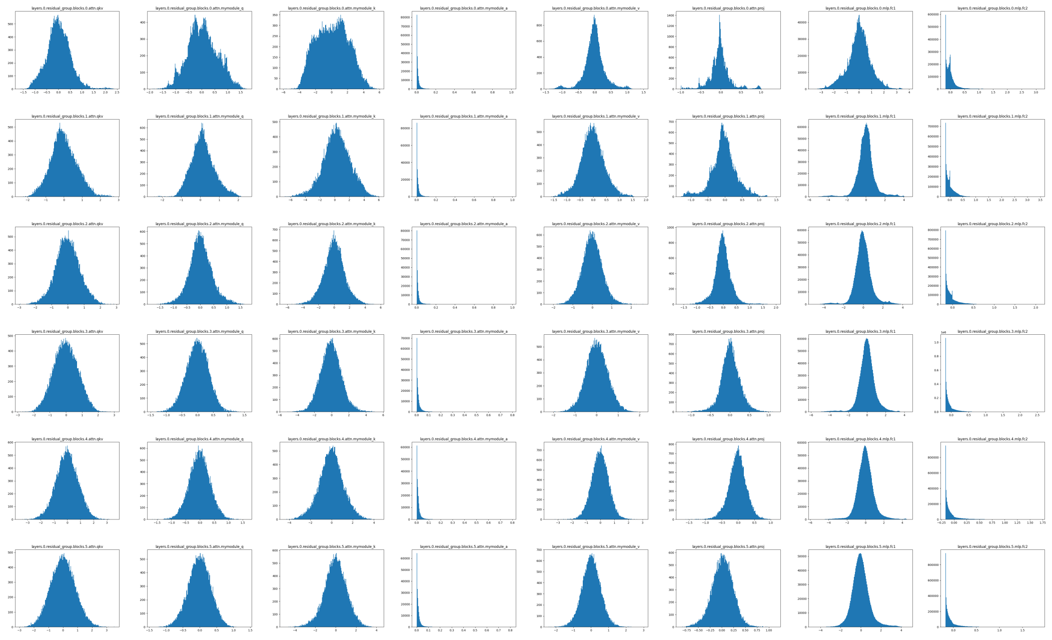

Figure 2: Visualization of SwinIR first layer activation.

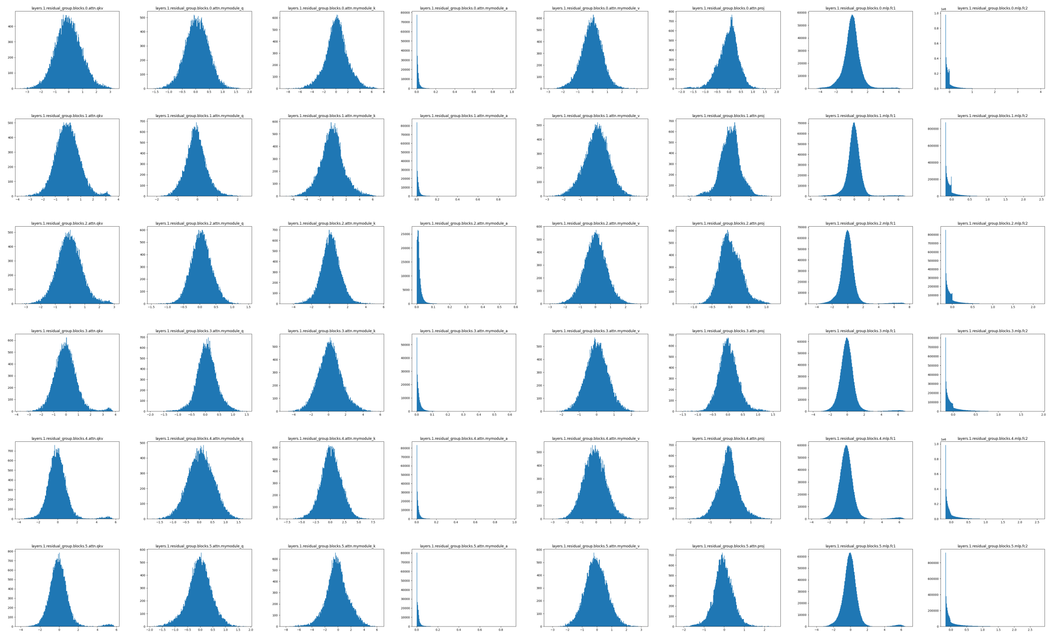

Figure 3: Visualization of SwinIR second layer activation.

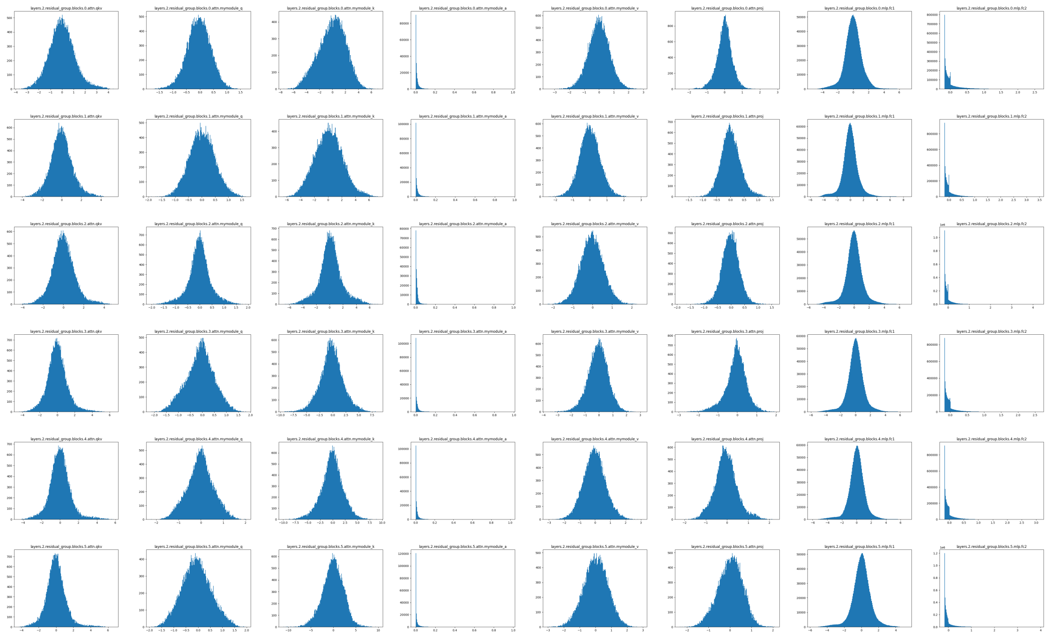
Figure 4: Visualization of SwinIR third layer activation.

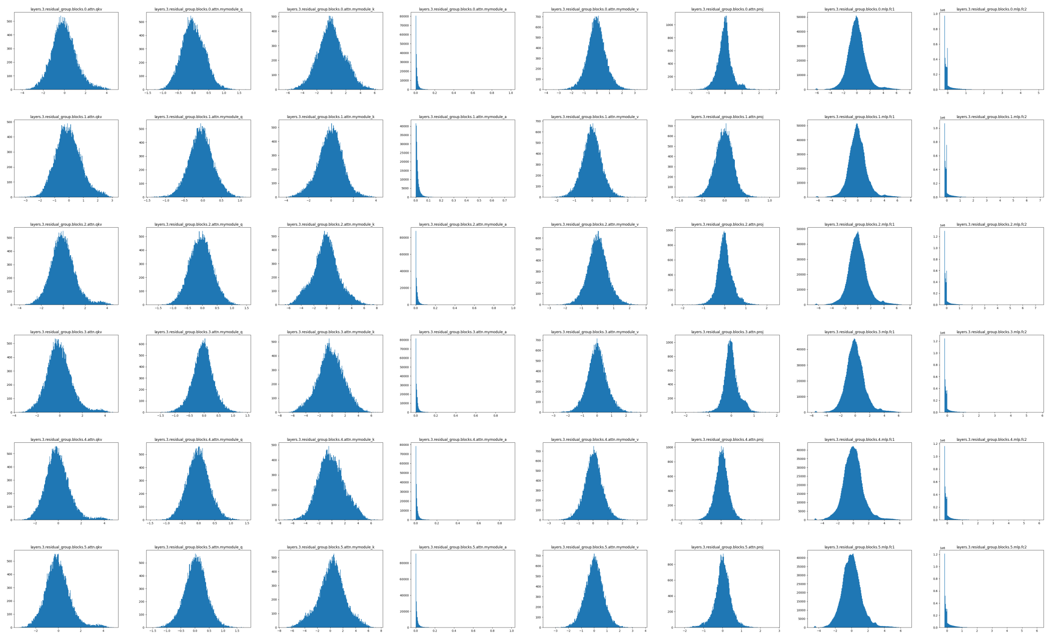
Figure 5: Visualization of SwinIR fourth layer activation.

## C The derivation of the backward gradient propagation formula

In this section, we provide the derivation of our backpropagation formula.We follow the STE **?** style to process the round term, which is

$$\frac{\partial \text{Round (x)}}{\partial x} = 1 \tag{11}$$

As for the clip function, we take a similar approach, which is

$$\begin{aligned}
\frac{\partial \text{Clip}(x, l, u)}{\partial x} &= \begin{cases} 1 & \text{if } l \leq x \leq u \\ 0 & \text{if } x < l \text{ or } x > u \end{cases} \\
\frac{\partial \text{Clip}(x, l, u)}{\partial l} &= \begin{cases} 1 & \text{if } x < l \\ 0 & \text{if } x \geq l \end{cases} \\
\frac{\partial \text{Clip}(x, l, u)}{\partial u} &= \begin{cases} 1 & \text{if } x > u \\ 0 & \text{if } x \leq u \end{cases}
\end{aligned} \tag{12}$$

With Eqs. (**??**), (11), and (12), we first derive $\frac{\partial v_q}{\partial u}$

$$\begin{aligned}
\frac{\partial v_q}{\partial u} &= \frac{\partial}{\partial u}(\frac{u-l}{2^N-1}v_r + l) \\
&= \frac{1}{2^N-1}v_r + \frac{u-l}{2^N-1}\frac{\partial v_r}{\partial u} \\
&= \frac{1}{2^N-1}v_r + \frac{u-l}{2^N-1}(-\frac{2^N-1}{(u-l)^2}(v_c - l) + \frac{2^N-1}{u-l}\frac{\partial v_c}{\partial u}) \\
&= \frac{\partial v_c}{\partial u} + \frac{1}{2^n-1}v_r - \frac{v_c - l}{u-l}
\end{aligned} \tag{13}$$

$\frac{\partial v_q}{\partial l}$ can be derived roughly the same, which can be written as

$$\begin{aligned}
\frac{\partial v_q}{\partial l} &= \frac{\partial}{\partial l}(\frac{u-l}{2^N-1}v_r + l) \\
&= -\frac{1}{2^N-1}v_r + \frac{u-l}{2^N-1}\frac{\partial v_r}{\partial u} + 1 \\
&= -\frac{1}{2^N-1}v_r + \frac{u-l}{2^N-1}(\frac{2^N-1}{(u-l)^2}(v_c - l) + \frac{2^N-1}{u-l}(\frac{\partial v_c}{\partial u} - 1)) + 1 \\
&= \frac{\partial v_c}{\partial u} - \frac{1}{2^n-1}v_r + \frac{v_c - l}{u-l}
\end{aligned} \tag{14}$$

## D More visual examples

We provide more visual illustrations to demonstrate the superiority of our method, as shown in Figure 6. In img_016, our method does not distort straight lines. In img_040, our method does not introduce noise to the camera and does not alter the shape at the camera lens. In img_072, we once again outperform the full-precision model by not adding vertical stripes to the curtains. In img_096, we ensure the shape of each window to the greatest extent. These images prove that we can surpass the current SOTA methods in visual effects and avoid misleading results in some tricky cases, generating correct results.

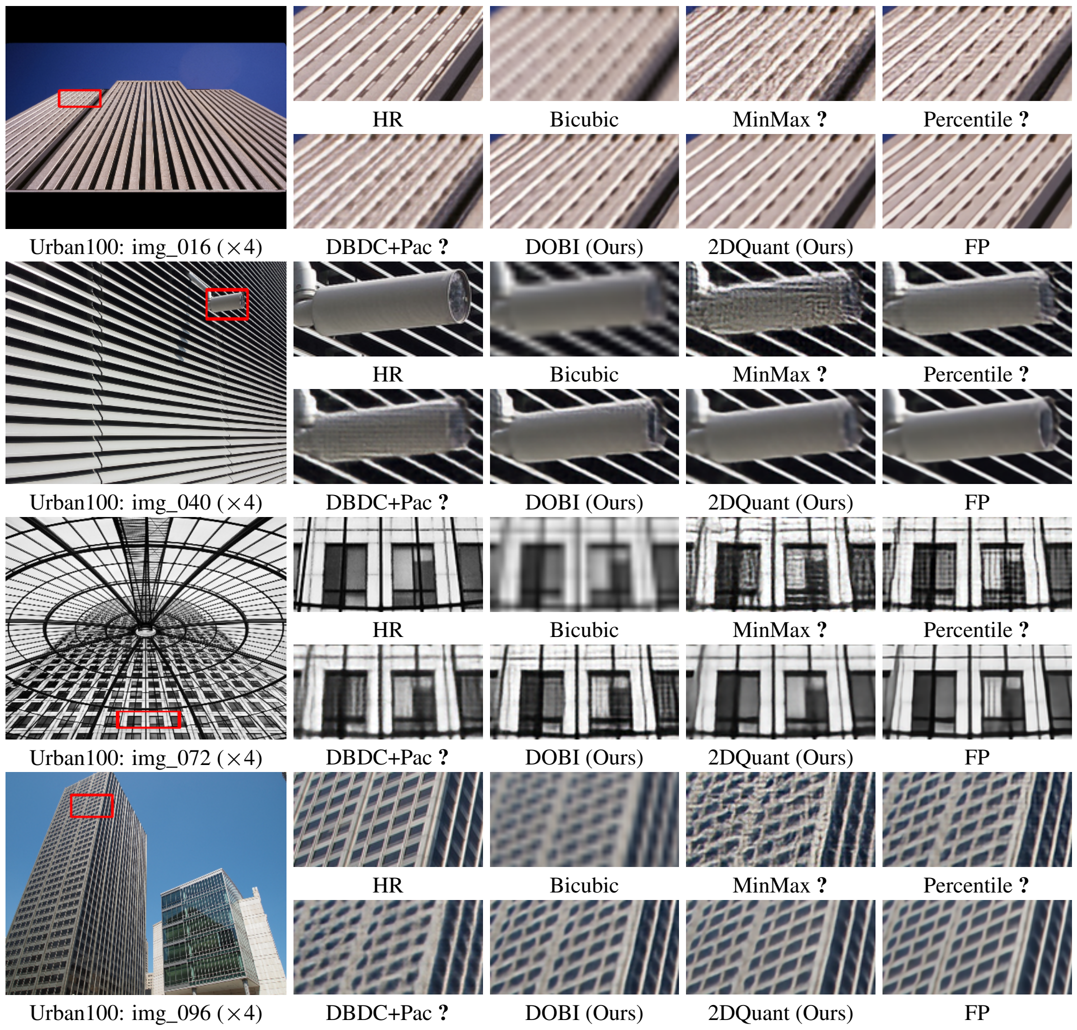

Figure 6: Visual comparison for image SR (×4) in some challenging cases.