

---

# Binarized Diffusion Model for Image Super-Resolution

---

Zheng Chen<sup>1</sup>, Haotong Qin<sup>2\*</sup>, Yong Guo<sup>3</sup>, Xiongfei Su<sup>4</sup>,  
Xin Yuan<sup>4</sup>, Linghe Kong<sup>1</sup>, Yulun Zhang<sup>1\*</sup>

<sup>1</sup>Shanghai Jiao Tong University, <sup>2</sup>ETH Zürich,

<sup>3</sup>Max Planck Institute for Informatics, <sup>4</sup>Westlake University

## Abstract

Advanced diffusion models (DMs) perform impressively in image super-resolution (SR), but the high memory and computational costs hinder their deployment. Binarization, an ultra-compression algorithm, offers the potential for effectively accelerating DMs. Nonetheless, due to the model structure and the multi-step iterative attribute of DMs, existing binarization methods result in significant performance degradation. In this paper, we introduce a novel binarized diffusion model, BI-DiffSR, for image SR. First, for the model structure, we design a UNet architecture optimized for binarization. We propose the consistent-pixel-downsample (CP-Down) and consistent-pixel-upsample (CP-Up) to maintain dimension consistent and facilitate the full-precision information transfer. Meanwhile, we design the channel-shuffle-fusion (CS-Fusion) to enhance feature fusion in skip connection. Second, for the activation difference across timestep, we design the timestep-aware redistribution (TaR) and activation function (TaA). The TaR and TaA dynamically adjust the distribution of activations based on different timesteps, improving the flexibility and representation ability of the binarized module. Comprehensive experiments demonstrate that our BI-DiffSR outperforms existing binarization methods. Code is released at: <https://github.com/zhengchen1999/BI-DiffSR>.

## 1 Introduction

Image super-resolution (SR) is a fundamental task in low-level vision and image processing. It aims to reconstruct high-resolution (HR) images from low-resolution (LR) counterparts. Currently, the mainstream methods for this task are deep neural networks, which employ learning-based techniques to map LR images to HR images [10, 70, 31, 54, 6, 68]. Among these methods, generative models [62, 9, 44] have garnered widespread attention for their ability to restore more realistic results.

Especially, the diffusion model (DM) [16, 58, 52], a newly proposed generative model, exhibits impressive performance. With its superior generation quality and more stable training, diffusion model is widely used in various image tasks, including image SR [54, 63]. Specifically, the diffusion model transforms a standard Gaussian distribution into a high-quality image through a stochastic iterative denoising process. In image SR, it further constrains the generation scope by conditioning on the LR image to produce the targeted HR image.

However, to produce high-quality results, diffusion models require thousands of iterative steps, leading to slow inference processes and high computational costs. Some methods [58, 40, 37] implement faster sampling strategies via learning sample trajectories, effectively reducing the number of iterations to tens. Yet, a single inference step still demands substantial memory usage and floating-point computations, especially for SR tasks involving high-resolution images. Meanwhile, most edge devices (*e.g.*, mobile and IoT devices), have limited storage and computational resources. This hampers the deployment of diffusion models on these platforms and limits their application. Therefore, it is essential to compress diffusion models to accelerate inference speed and reduce computational costs while maintaining model performance.

---

\*Corresponding authors: Haotong Qin, [qinhaotong@gmail.com](mailto:qinhaotong@gmail.com); Yulun Zhang, [yulun100@gmail.com](mailto:yulun100@gmail.com)

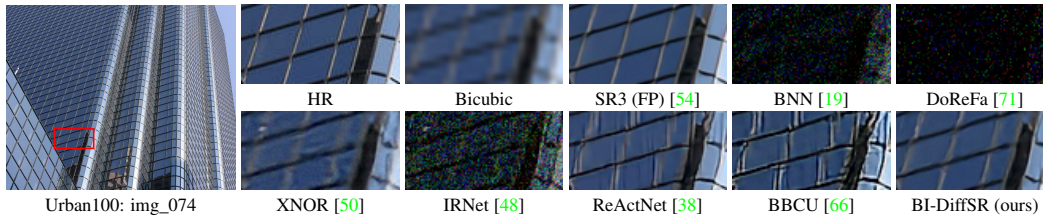


Figure 1: Visual comparison ( $\times 4$ ) of binarization methods. Some methods (*e.g.*, BNN [19]) cannot work on diffusion models. Several methods (*e.g.*, BBCU [66]) suffer from blurring and artifacts. In contrast, our proposed BI-DiffSR outperforms other methods with accurate results.

Common compression approaches include pruning [11], distillation [61], and quantization [45, 66, 26]. Among these, 1-bit quantization (*i.e.*, binarization) stands out for its effectiveness. As the most aggressive form of bit-width reduction, binarization significantly reduces memory and computational costs by quantizing the weights and activations of full-precision (32-bit) models to 1-bit.

Nonetheless, existing binarization research primarily deals with higher-level tasks (*e.g.*, classification) and end-to-end models [49, 19, 39]. Applying existing binarization methods directly to current diffusion model architectures results in a significant performance drop. This is primarily due to two aspects: **(1) Model Structure.** Diffusion models typically apply the UNet architecture [53] for noise estimation, which is not easy to binarize directly. **I. Dimension Mismatch:** The identity shortcut is crucial for the binarized SR model, since it facilitates the transfer of full-precision (FP) information, compensating for the binarized model [66]. However, in UNet, the feature dimensions change since downsampling/upsampling. The dimension mismatch prevents the usage of shortcuts, cutting off the full-precision propagation. **II. Fusion Difficulty:** The UNet structure uses skip connections to transfer information from encoder to decoder. However, the typical fusion method, concatenation, leads to the dimension mismatch. Alternatively, other methods (*e.g.*, addition) also struggle to achieve effective fusion due to significant differences in value ranges between encoder and decoder features. **(2) Activation Distribution.** Due to the multi-step iterative nature of diffusion models, the activation distribution dramatically changes with timesteps. Furthermore, the activation binarization will even amplify activation differences [50]. The difference increases the learning challenges for binarized modules (*e.g.*, binarized convolution), thereby hindering the effective representation of features. Consequently, the SR performance of the binarized diffusion model is limited.

Based on the above analysis, we propose a novel binarized diffusion model, BI-DiffSR, to achieve effective image SR. Our design comprises two main aspects: structure and activation. **(1) Structure.** We develop a simple yet effective convolutional UNet architecture, which is suitable for binarization. **I. Dimension Consistency:** We propose consistent-pixel-downsample (CP-Down) and consistent-pixel-upsample (CP-Up) to ensure dimensional consistency in binarized computation. CP-Down and CP-Up maintain the full-precision information transfer during feature scaling. **II. Feature Fusion:** We develop the channel-shuffle-fusion (CS-Fusion) to facilitate the fusion of different features within skip connections and suit binarized modules. Through channel shuffle, we combine two input features into two shuffled features to balance their activation value ranges. **(2) Activation.** Considering the activation differences at different timesteps, we design the timestep-aware redistribution (TaR) and timestep-aware activation function (TaA). The TaR and TaA adjust the binarized module input and output activations according to different timesteps. This timestep-aware adjustment improves the flexibility and representational ability of the binarized module to various activation distributions.

Extensive experiments demonstrate that our proposed BI-DiffSR significantly outperforms existing binarization methods. As shown in Fig. 1, our BI-DiffSR restores more perceptually pleasing results than other methods. Overall, our contributions are as follows:

- We design the novel binarized model, BI-DiffSR, for image SR. To the best of our knowledge, this is the first binarized diffusion model applied to SR.
- We develop a UNet architecture optimized for binarization, with consistent-pixel-downsample (CP-Down) and upsample (CP-Up), and channel-shuffle-fusion (CS-Fusion).
- We introduce the timestep-aware redistribution (TaR) and activation function (TaA) to adapt activation distributions by timestep, enhancing the capabilities of the binarized module.
- Our BI-DiffSR surpasses current state-of-the-art binarization methods, and offers comparable perceptual performance to full-precision diffusion models.

## 2 Related Work

### 2.1 Image Super-Resolution

Since the advent of SRCNN [10], deep neural networks have gradually become the mainstream for image SR. Numerous architectures [33, 70, 46, 31, 5] are designed to advance reconstruction accuracy. Concurrently, generative methods are widely applied to improve the quality of restored image details. This includes autoregressive model [23, 9], normalizing flow [51, 41, 32], and generative adversarial network (GAN) [13, 24]. For instance, SRFlow [41] utilizes normalizing flows to transform a Gaussian distribution into the HR image space. Meanwhile, SRGAN [24] employs GAN as supervision loss and combines it with perceptual loss to produce visually pleasing results. Subsequent methods [62, 4] further refine the network and loss to yield more natural results. Recently, the diffusion model (DM) [16, 8] has been introduced into SR, displaying impressive performance, especially regarding perception. Thereby, DM has been attracting widespread attention [54, 25, 65].

### 2.2 Diffusion Model

Through the Markov chain, the diffusion model (DM) generates images from the Gaussian distribution [16]. It has demonstrated exceptional performance in various tasks [3, 17, 52, 7, 14, 30, 29, 36, 35, 28, 15]. Naturally, DM has also been extensively researched in the field of image SR [54, 21, 63, 34, 65]. For instance, SR3 [54] achieves conditional diffusion by concatenating the resized LR image with the noise image as the input of the noise estimation network. Meanwhile, some methods, *e.g.*, DDNM [63], utilize an unconditional pre-trained diffusion model as a prior for zero-shot SR. Additionally, some approaches [34, 65] employ text-to-image diffusion models to achieve realistic and controllable SR. Despite promising results, these methods require hundreds or thousands of sampling steps to generate HR images. Although some acceleration algorithms [58, 37, 28] reduce the inference steps to tens, each denoising step still demands substantial resources. The high memory and computational costs hinder the practical application of DMs on resource-limited platforms (*e.g.*, mobile devices). To address this issue, we design a practical binarized SR diffusion model.

### 2.3 Binarization

Binarization is a popular model compression approach [49]. As an extreme case of quantization, it reduces the weights and activations of a full-precision neural network to 1-bit. This significantly decreases the model size and computational complexity, making it widely used in both high-level [19, 39, 48, 38, 67] and low-level [20, 66, 66, 69] vision tasks. For example, BNN [19] directly binarizes weights and activations during forward and backward processes. IRNet [48] retains information accurately through the proposed information retention network. ReActNet [38] proposes the RSign and RPreLU to enable explicit distribution reshape and shift of activations. Meanwhile, in the image SR field, BBCU [66] introduces a meticulously designed basic binary conv unit, which removes batch normalization (BN) in the binarized model. However, for DM, though some methods realize low-bit (*e.g.*, 4 or 8) quantization [55, 26, 27], implementing binarization remains challenging. Due to the structure of the noise estimation network and the multi-step iterative attribute, existing binarization methods often result in significant SR performance degradation.

## 3 Method

In this section, we introduce our proposed BI-DiffSR. First, we describe the structural designs suitable for binarization: *consistent-pixel-downsample* (CP-Down), *consistent-pixel-upsample* (CP-Up), and *channel-shuffle-fusion module* (CS-Fusion). The CP-Down and CP-Up achieve dimension adjustment and ensure the transfer of full-precision information. The CS-Fusion effectively integrates different features within the skip connection. Secondly, we present the dynamic designs tailored for varying activations: *timestep-aware redistribution* (TaR) and *activation function* (TaA). The TaR and TaA enhance the representational learning of the binarized modules across multiple timesteps.

### 3.1 Model Structure

**Overall.** We employ a convolutional UNet [53] as the noise estimation network. Details of the diffusion model for SR are provided in the supplementary materials. As the common choice within DMs, using UNet as the backbone for binarization offers generalizability. Moreover, for binarized models, the design should be compact and well-defined. Compared to the non-local self-attention operations, convolution is simpler and easier to implement. Our architecture is shown in Fig. 2a, featuring an encoder-bottleneck-decoder ( $\mathcal{E}\text{-}\mathcal{B}\text{-}\mathcal{D}$ ) design.

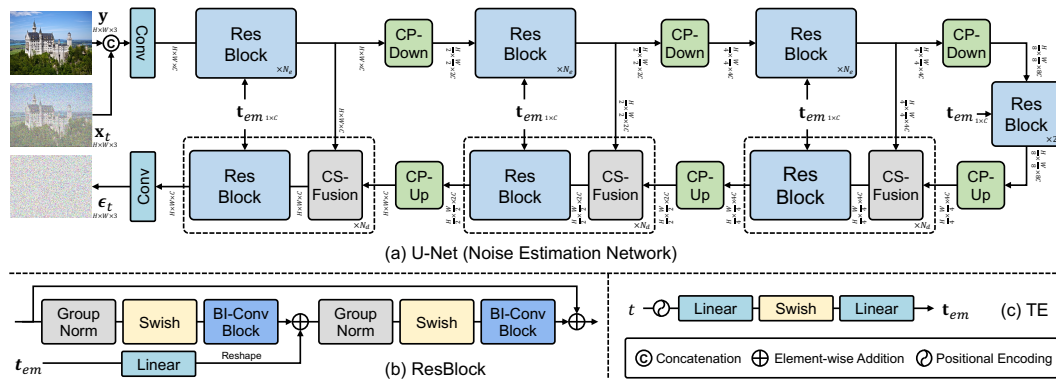


Figure 2: The overall structure of the noise estimation network. (a) UNet: The model consists of ResBlock, CP-Down, CP-Up, and CS-Fusion. It predicts noise  $\epsilon_t$  with the upscaled LR image  $y$ , noise image  $x_t$ , and timestep  $t$ . (b) ResBlock: Residual block, utilizes the binarized convolution (BI-Conv) block. The input and output dimensions of the block remain consistent, making it suitable for binarization. (c) TE: Time encoding, encoders timestep  $t$  to produce the timestep embedding  $t_{em}$ .

Given the noise image  $x_t \in \mathbb{R}^{H \times W \times 3}$  at  $t$ -th timestep, and the LR image  $y \in \mathbb{R}^{H \times W \times 3}$  (bicubic to HR resolution), two images are concatenated along the channel dimension as the UNet input, where  $H \times W$  is the resolution. For timestep  $t$ , the sinusoidal position encoding [60] is applied to obtain the timestep embedding  $t_{em} \in \mathbb{R}^C$ . The input images first pass through a convolutional layer to produce the shallow feature  $F_s \in \mathbb{R}^{H \times W \times C}$ , where  $C$  is the channel number. Then, the shallow feature  $F_s$  are further refined by the  $\mathcal{E}$ - $\mathcal{B}$ - $\mathcal{D}$  into the deeper feature  $F_d \in \mathbb{R}^{H \times W \times C}$ . Each level of the  $\mathcal{E}$ - $\mathcal{B}$ - $\mathcal{D}$  is composed of multiple ( $N_e$  in  $\mathcal{E}$  and  $N_d$  in  $\mathcal{D}$ ) residual blocks (ResBlocks), with details illustrated in Fig. 2b. Within the ResBlocks, the timestep embedding  $t_{em}$  is incorporated to provide temporal information. In the encoder  $\mathcal{E}$ , downsample module (*i.e.*, CP-Down) progressively reduces feature resolution and increases channel number. Conversely, in the decoder  $\mathcal{D}$ , upsample module (*i.e.*, CP-Up) gradually restores the high-resolution representation. Moreover, to compensate for information loss during downsampling, the skip connection is used to link features between the encoder and decoder. Finally, through one convolution, the predicted noise  $\epsilon_t \in \mathbb{R}^{H \times W \times 3}$  is obtained.

**Structure Analysis.** Although the UNet architecture is suitable for diffusion models, its unique structure poses challenges for direct binarization, which results in a substantial accuracy decrease compared to full-precision models. We identify two main issues/challenges that contribute to the problem: *dimension mismatch* and *fusion difficulty*.

**Challenge I: Dimension Mismatch.** In the binarized model, 1-bit quantization leads to significant information loss, limiting the capability for feature representation and the ultimate SR performance. Compared to binary activations, full-precision activations contain more information. Therefore, we can apply the identity shortcut to preserve the full-precision information. This operation effectively compensates for the information loss caused by binarization. However, in UNet, the frequent changes in feature resolution and channel size lead to dimension mismatches. This prevents the effective use of the identity shortcut and cuts off the propagation of full-precision information.

**Challenge II: Fusion Difficulty.** Another crucial structure of UNet is the skip connection, which links encoder and decoder features. The typical approach is to concatenate these features along the channel dimension and pass them to subsequent layers. However, concatenate causes dimension mismatch. As analyzed in **Challenge I**, it is unsuitable for binarization. Furthermore, we find that there is a significant difference in the activation ranges between the two inputs (from encoder and decoder) of the skip connection (Fig. 3d). This imbalance makes other fusion methods, *e.g.*, addition, also unsuitable, since the smaller range activation is masked by the larger one, as illustrated in Fig. 3d.

To better adapt binarization for the UNet architecture, we propose two structures: **Consistent-Downsample/Upsample** and **Channel-Shuffle Fusion**, as illustrated in Fig. 3.

**Consistent-Pixel-Downsample/Upsample.** To address the dimension mismatch in the UNet structure, we first confine all feature reshaping operations to the Upsample and Downsample modules. That is to ensure that the dimension of the main module, *i.e.*, ResBlock, remains matched. Meanwhile, we propose the consistent-pixel-downsample (CP-Down) and consistent-pixel-upsample (CP-Up).

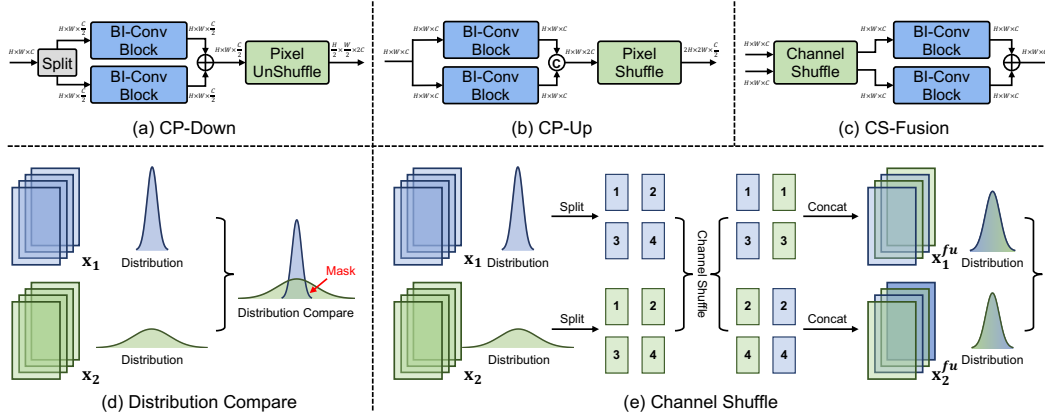


Figure 3: (a) CP-Down: Consistent-pixel-downsample. (b) CP-Up: Consistent-pixel-upsample. (c) CS-Fusion: Channel-shuffle fusion. (d) In the skip connection, the value ranges of two features ( $x_1$ ,  $x_2$ ) may be significant differences, which impedes effective fusion. (e) The illustration of channel shuffle. the shuffled features ( $x_1^{sh}$ ,  $x_2^{sh}$ ) have closely matched value ranges.

**(1) CP-Down:** We evenly split the input features  $\mathbf{x}_{in}^{do} \in \mathbb{R}^{H \times W \times C}$  along the channel dimension and process them through two convolutions with identical input and output dimensions. The stable (matching) dimension allows the usage of identity shortcuts. Finally, by applying Pixel-UnShuffle [57], we reduce the resolution of the features while increasing the channel number. The formula is:

$$\mathbf{x}_{out}^{do} = [\mathbf{x}_s^1, \mathbf{x}_s^2], \quad \mathbf{x}_s^i \in \mathbb{R}^{H \times W \times \frac{C}{2}}, \quad \mathbf{x}_{out}^{do} = \mathcal{PS}^{-1}(\mathcal{C}_1(\mathbf{x}_s^1) + \mathcal{C}_2(\mathbf{x}_s^2)), \quad (1)$$

where  $\mathbf{x}_{out}^{do} \in \mathbb{R}^{\frac{H}{2} \times \frac{W}{2} \times 2C}$  is the output of CP-Down;  $\mathcal{C}_1(\cdot)$  and  $\mathcal{C}_2(\cdot)$  represent two (binarized) convolutions;  $\mathcal{PS}^{-1}$  denotes the Pixel-UnShuffle operation.

**(2) CP-Up:** Similarly, feature upsampling is achieved through two convolutions combined with Pixel-Shuffle. The operation can be mathematically expressed as follows:

$$\mathbf{x}_{out}^{up} = \mathcal{PS}(\text{Concat}(\mathcal{C}_1(\mathbf{x}_{in}^{up}), \mathcal{C}_2(\mathbf{x}_{in}^{up}))), \quad (2)$$

where,  $\mathbf{x}_{in}^{up} \in \mathbb{R}^{H \times W \times C}$  and  $\mathbf{x}_{out}^{up} \in \mathbb{R}^{2H \times 2W \times \frac{C}{2}}$  denotes the input and output of CP-Up;  $\text{Concat}(\cdot)$  represents the channel concatenation operation;  $\mathcal{PS}$  is the Pixel-Shuffle operation.

With the above design, we ensure the flow of full-precision information throughout the UNet, effectively improving feature representation and enhancing SR performance.

**Channel-Shuffle Fusion.** To effectively fuse the features in the skip connection while meeting the requirements for dimension matching in binarization, we propose the channel-shuffle fusion (CS-Fusion), as shown in Fig. 3c. Given two features  $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^{H \times W \times C}$ , we first employ the channel-shuffle operation to mitigate the differences in their value ranges, as illustrated in Fig. 3e. Specifically, we split the two features according to the odd and even channel indexes. Then, we pair and concatenate features along the channel dimension, based on odd and even indexes, to produce two new shuffle features  $\mathbf{x}_1^{sh}, \mathbf{x}_2^{sh} \in \mathbb{R}^{H \times W \times C}$ . This process can be formulated as follows:

$$\mathbf{x}_n = [\mathbf{x}_n^1, \mathbf{x}_n^2, \dots, \mathbf{x}_n^{C-1}, \mathbf{x}_n^C], \quad n \in \{1, 2\}, \quad (3)$$

$$\mathbf{x}_m^{sh} = \text{Concat}\left(\left\{\mathbf{x}_j^{2i+(m-1)} \mid i = 1, \dots, C/2, j = 1, 2\right\}\right), \quad m \in \{1, 2\},$$

Through visualization in Fig. 3e, we can observe that the value range of features after channel shuffle becomes balanced. Subsequently, we process the shuffled features through two convolutions and addition to produce the final fused feature  $\mathbf{x}_{out}^{sh} \in \mathbb{R}^{H \times W \times C}$ , in a manner similar to Eq. (1), as:

$$\mathbf{x}_{out}^{sh} = \mathcal{C}_1^{sh}(\mathbf{x}_1^{sh}) + \mathcal{C}_2^{sh}(\mathbf{x}_2^{sh}), \quad (4)$$

where  $\mathcal{C}_1^{sh}(\cdot)$  and  $\mathcal{C}_2^{sh}(\cdot)$  are two (binarized) convolutions. This process realizes the fusion of two features, ensuring that dimensions are matched within the fusion process and in subsequent modules (e.g., ResBlock). Meanwhile, the matched dimension allows the usage of the identity shortcut, thus effectively transferring full-precision information. Overall, our proposed CS-Fusion achieves effective feature integration in the skip connection. Therefore, the binarized model can better represent features and improve SR performance. Furthermore, our CS-Fusion does not introduce additional memory or computational overhead since the channel shuffle only involves feature transformation operations. Experiments in Sec. 4.2 further reveal the impacts of CS-Fusion.

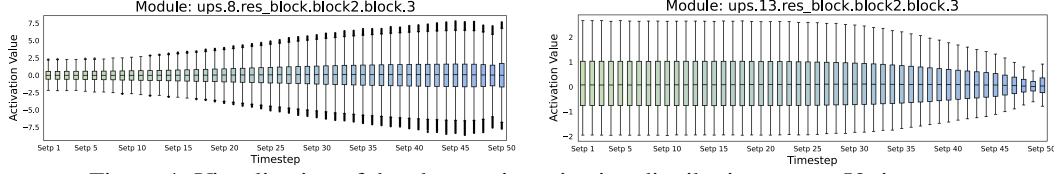


Figure 4: Visualization of the changes in activation distribution across 50 timesteps.

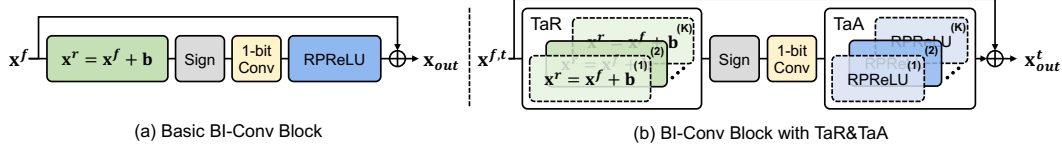


Figure 5: (a) The basic binarized convolutional (BI-Conv) block. The learnable bias  $\mathbf{b}$  and the activation function RPreLU adjust the activations. (b) In timestep-aware redistribution (TaR) and activation function (TaA), multiple pairs of  $\mathbf{b}$  and RPreLU are applied to adapt to the multi-step in DM. At each step  $t$ , only one pair of  $\mathbf{b}$  and RPreLU is used (the darker modules with solid lines).

### 3.2 Activation Distribution

**Basic Binarized Convolutional Block.** We first introduce the basic binarized module, as illustrated in Fig. 5a. For the full-precision activation  $\mathbf{x}^f \in \mathbb{R}^{H \times W \times C}$ , we initially shift its distribution and binarize the shifted activation to 1-bit activations with sign function  $\text{Sign}(\cdot)$ . The process is:

$$\mathbf{x}^r = \mathbf{x}^f + \mathbf{b}, \quad \mathbf{x}^b = \text{Sign}(\mathbf{x}^r) = \begin{cases} +1, & x^r \geq 0 \\ -1, & x^r < 0 \end{cases} \quad (\forall x^r \in \mathbf{x}^r, \forall x^b \in \mathbf{x}^b), \quad (5)$$

where  $\mathbf{b} \in \mathbb{R}^C$  is a learnable parameter;  $\mathbf{x}^b \in \mathbb{R}^{H \times W \times C}$  is the 1-bit activation. Meanwhile, for the binarized convolution, the full-precision weight  $\mathbf{w}^f \in \mathbb{R}^{C_{out} \times C_{in} \times K_h \times K_w}$  is also binarized to 1-bit weight  $\mathbf{w}^b \in \mathbb{R}^{C_{out} \times C_{in} \times K_h \times K_w}$ . To compensate for the differences between binary and full-precision weights, we scale  $\mathbf{w}^b$  using the mean absolute value of  $\mathbf{w}^f$  [50]. The total operation is:

$$\mathbf{w}^b = \frac{\|\mathbf{w}^f\|_1}{n} \cdot \text{Sign}(\mathbf{w}^f), \quad \forall w^f \in \mathbf{w}^f, \forall w^b \in \mathbf{w}^b, \quad (6)$$

where  $n$  is the number of  $\mathbf{w}^f$  values. Subsequently, the floating-point matrix multiplication in full-precision convolution can be replaced by logical XNOR and bit-counting operations as:

$$\mathbf{x}_{out}^b = \mathbf{x}^b * \mathbf{w}^b = \text{bit-count}(\text{XNOR}(\mathbf{x}^b, \mathbf{w}^b)) \quad (7)$$

where  $*$  means the convolutional operation;  $\mathbf{x}_{out}^b \in \mathbb{R}^{H \times W \times C}$  is the output of 1-bit convolution. Then, we adjust  $\mathbf{x}_{out}^b$  with the activation function RPreLU [38], resulting in  $\mathbf{x}_{act}^b \in \mathbb{R}^{H \times W \times C}$ .

Finally, we combine  $\mathbf{x}_{act}^b$  with full-precision activation  $\mathbf{x}^f$  via an identity shortcut to get the final output  $\mathbf{x}_{out} \in \mathbb{R}^{H \times W \times C}$ . Moreover, since the sign function  $\text{Sign}(\cdot)$  is non-differentiable, we use the straight-through estimator (STE) [1] for backpropagation to train binarized models.

**Distribution Analysis.** In diffusion models, the multi-step iterative design leads to changes in the activation distribution as the timestep changes. By visualizing the activation distributions at different timesteps in Fig. 4, we can observe that activation distributions of adjacent timesteps are similar, whereas those separated by larger intervals show significant differences.

For full-precision models, the impact of these variations may be small due to the real-valued weight and activation. In contrast, for binarized modules, the activation distribution has a substantial impact on feature representation, and consequently, affects the SR performance. This is because 1-bit modules, due to the binary weights, struggle to effectively learn representations from different distributions, thereby limiting their modeling capabilities. Meanwhile, during the activation binarization, the sign function further amplifies activation differences, particularly for values around zero [38].

The basic binarized module utilizes the learnable bias and the activation function RPreLU to adjust the input and output activations. This approach mitigates the representational challenges posed by activation distribution differences across timestep to some extent. However, these static designs are insufficient to cope with the extreme activation changes across multiple timesteps in diffusion models. Consequently, the SR performance of the binarized diffusion model is limited. Experiments in Sec. 4.2, further demonstrate the above analyses.

**Timestep-aware Redistribution/Activation Function.** To cope with the variability of activation distribution with timestep, we propose the timestep-aware redistribution (TaR) and timestep-aware activation function (TaA). The module details are illustrated in Fig. 5b. The design of TaR and TaA is inspired by the mixture of experts (MoE) [56], applying a set of learnable biases and RPRReLU activation functions to accommodate different timesteps.

Specifically, we apply  $K$  pairs of bias and RPRReLU for TaR ( $\mathbf{b}^{(i)} \in \mathbb{R}^C$ ) and TaA (RPRReLU $^{(i)}$ ), where  $i \in \{1, 2, \dots, K\}$ . Given the total timesteps (e.g.,  $\{1, 2, \dots, T\}$ ), we evenly divide them into  $K$  groups in sequence. For the input activation  $\mathbf{x}^{f,t} \in \mathbb{R}^{H \times W \times C}$  at  $t$ -th timestep ( $t \in \{1, 2, \dots, T\}$ ), we select the corresponding pair of bias and RPRReLU based on the group associated with  $t$ , to adjust its input and output activation. The process can be formulated as:

$$\begin{aligned}\mathbf{x}^{r,t} &= \text{TaR}(\mathbf{x}_{in}^t) = \mathbf{x}_{in}^t + \sum_{i=1}^K \mathbf{1}_{i=\lfloor K \times t/T \rfloor} \cdot \mathbf{b}^{(i)}, \\ \mathbf{x}_{act}^{b,t} &= \text{TaA}(\mathbf{x}_{out}^{b,t}) = \sum_{i=1}^K \mathbf{1}_{i=\lfloor K \times t/T \rfloor} \text{RPRReLU}^{(i)}(\mathbf{x}_{out}^{b,t}),\end{aligned}\tag{8}$$

where  $\mathbf{1}_{(\cdot)}$  is the indicator function;  $\mathbf{x}^{r,t}$ ,  $\mathbf{x}_{out}^{b,t}$ ,  $\mathbf{x}_{act}^{b,t} \in \mathbb{R}^{H \times W \times C}$ , represent, at  $t$ -th timestep, the shifted input activation, the output of the 1-bit convolution, the output of the RPRReLU activation function, respectively. Since the activations at adjacent timesteps exhibit a certain degree of similarity (as shown in Fig. 4), we employ the fixed grouping sampling strategy (defined in Eq. (8)).

Essentially, the TaR and TaA segment the multi-step process into smaller groups, limiting the range of activation changes. This reduces the difficulty of adjusting activations, allowing the binarized module to better adapt to changing activations. Therefore, the proposed TaR and TaA can effectively enhance the representation ability of the binarized module and ultimately improve SR performance. Meanwhile, compared to the basic module, there are no additional computational costs in our TaR and TaA. This is because, for each timestep, only one pair of bias and RPRReLU are selected for use.

## 4 Experiments

### 4.1 Experimental Settings

**Data and Evaluation.** We take DIV2K [59] and Flickr2K [33] as the training dataset. Meanwhile, we evaluate the models with four benchmark datasets: Set5 [2], B100 [42], Urban100 [18], and Manga109 [43]. Experiments are conducted under two upscale factors:  $\times 2$  and  $\times 4$ . The LR images are generated from HR images through bicubic downsampling degradation. We apply two distortion-based metrics, PSNR and SSIM [64], which are calculated on the Y channel (*i.e.*, luminance) of the YCbCr space. We also use the perceptual metrics: LPIPS [12]. Following previous work [66, 49], the total parameters (**Params**) of the model are calculated as  $\text{Params} = \text{Params}^b + \text{Params}^f$ , and the overall operations (**OPs**) as  $\text{OPs} = \text{OPs}^b + \text{OPs}^f$ , where  $\text{Params}^b = \text{Params}^f / 32$  and  $\text{OPs}^b = \text{OPs}^f / 64$ ; the superscripts  $f$  and  $b$  denote full-precision and binarized modules, respectively.

**Implementation Details.** For the noise estimation network, we set the encoder and decoder level to 4. In each level of the encoder, we use 2 Residual Blocks (ResBlocks), while in the decoder, we apply 3 ResBlocks. The number of channels  $C$  is set to 64. We set the number of bias and RPRReLU in TaR and TaA as  $K=5$ . For the diffusion model, we set the total number of timesteps to  $T=2,000$ . During the inference phase, we employ the DDIM sampler with 50 timesteps.

**Training Settings.** We train models with the  $\mathcal{L}_1$  loss. We employ the Adam optimizer [22] with  $\beta_1=0.9$  and  $\beta_2=0.99$ , and a learning rate of  $1 \times 10^{-4}$ . The batch size is set to 16, with a total of 1,000K iterations. Input LR images are randomly cropped to size  $64 \times 64$ . Random rotations of  $90^\circ$ ,  $180^\circ$ , and  $270^\circ$  and horizontal flips are used for data augmentation. Our model is implemented based on PyTorch [47] with two Nvidia A100-80G GPUs.

### 4.2 Ablation Study

In this section, we conduct all experiments on the  $\times 2$  scale factor. We apply DIV2K [59] and Flickr2K [33] as the training dataset, and Manga109 [43] as the testing dataset. The training iterations are set to 500K. Other settings are the same as defined in Sec. 4.1. We test the computational complexity (*i.e.*, OPs) of one single sampling step on the output size  $3 \times 256 \times 256$ .

| Method     | Baseline | +Identity | +CP-Down&Up | +CS-Fusion | +TaR&TaA |
|------------|----------|-----------|-------------|------------|----------|
| Params (M) | 4.29     | 4.29      | 4.29        | 4.30       | 4.58     |
| OPs (G)    | 36.67    | 36.67     | 36.67       | 36.67      | 36.67    |
| PSNR (dB)  | 27.66    | 29.29     | 31.08       | 31.99      | 32.66    |
| LPIPS      | 0.0780   | 0.0658    | 0.0327      | 0.0261     | 0.0200   |

(a) Break-down ablation.

| Method | TaR | TaA | Params (M) | Ops (G) | PSNR (dB) | LPIPS  |
|--------|-----|-----|------------|---------|-----------|--------|
| w/o    |     |     | 4.30       | 36.67   | 31.99     | 0.0261 |
| In     | ✓   |     | 4.37       | 36.67   | 29.27     | 0.0337 |
| Out    |     | ✓   | 4.51       | 36.67   | 29.13     | 0.0308 |
| All    | ✓   | ✓   | 4.58       | 36.67   | 32.66     | 0.0200 |

(c) Ablation on time aware module (TaR and TaA).

| Method    | Params (M) | OPs (G) | PSNR (dB) | LPIPS  |
|-----------|------------|---------|-----------|--------|
| Add       | 4.10       | 33.40   | 18.89     | 0.1695 |
| Concat    | 4.29       | 36.67   | 31.08     | 0.0327 |
| Split     | 4.30       | 36.67   | 29.67     | 0.0384 |
| CS-Fusion | 4.30       | 36.67   | 31.99     | 0.0261 |

(b) Ablation on feature fusion.

| #Pair      | 1      | 2      | 5      |
|------------|--------|--------|--------|
| Params (M) | 4.30   | 4.37   | 4.58   |
| OPs (G)    | 36.67  | 36.67  | 36.67  |
| PSNR (dB)  | 31.99  | 32.42  | 32.66  |
| LPIPS      | 0.0261 | 0.0229 | 0.0200 |

(d) Numbers (#) of bias and RReLU pair.

Table 1: Ablation study. We train models on DIV2K and Flickr2K, and evaluate on Manga109 ( $\times 2$ ).

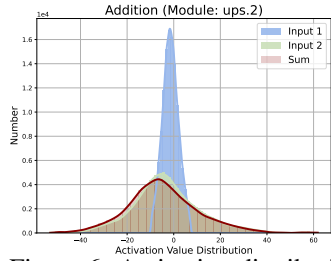


Figure 6: Activation distribution in the skip connection. Input 1(2):  $\mathbf{x}_1, \mathbf{x}_2$ . Sum:  $\mathbf{x}_1 + \mathbf{x}_2$ . Fusion 1(2):  $\mathbf{x}_1^{sh}, \mathbf{x}_2^{sh}$ .

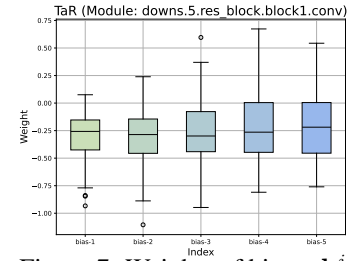
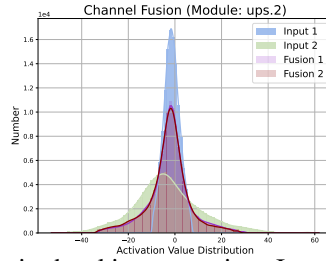


Figure 7: Weights of biases  $\mathbf{b}^i$  ( $i \in \{1, \dots, 5\}$ ) in TaR.

**Break Down.** We first execute a break-down ablation on different components of our method. The results are listed in Tab. 1a. The baseline is established by using binarized convolution (BI-Conv) and Pixel(Un)Shuffle for dimension scaling in the downsample, upsample, and fusion (skip connection) modules of the UNet. Meanwhile, the basic BI-Conv block (Fig. 5) is employed without the identity shortcut. The baseline performance is poor, with the PSNR of 27.66 dB. Then, we add identity shortcut, consistent-pixel-downsample (CP-Down) and upsample (CP-Up), channel-shuffle-fusion module (CS-Fusion), and timestep-aware redistribution (TaR) and activation function (TaA) in sequence. We can find that the performance gradually increases. Ultimately, the final model achieves gains of 5 dB in PSNR and 0.0580 in LPIPS, compared to the baseline.

**Channel-Shuffle Fusion.** We experiment on the fusion module for the skip connection. We attempt four methods: directly add two features (Add); concatenation and adjust dimension by binarized convolution (Concat); process each feature via binarized convolution and add them; and our proposed CS-Fusion. The results are shown in Tab. 1b. Due to the differences between features, direct addition (Add) can hardly work, even with convolution (Split). Moreover, since the concatenation changes the dimensions, the Method (Concat) also degrades the performance. In contrast, our proposed CS-Fusion, eliminates the distribution imbalances by channel fusion, thereby achieving effective fusion. The visualization in Fig. 6, further indicates that addition cannot fuse data with narrow value distributions, whereas channel shuffle can effectively integrate.

**Timestep-aware Module.** We conduct experiments on the time-aware redistribution (TaR) and activation function (TaA). Firstly, we experiment with the combinations of TaR and TaA in Tab. 1c. We find that effective improvements are only achieved when both TaR and TaA are employed. This may be because both input and output activation impact the learning of the binarized module. Then, in Tab. 1d, we experiment with the pair number (#Pair) of bias and RReLU. The experiments show that 5 pairs already lead to effective improvements. Considering the additional parameters, we adopt 5 as the pair number in BI-DiffSR. Moreover, we present the weights of five learnable biases in the TaR (module position shown at the image top) in Fig. 7. The difference in weights indicates that TaR can effectively adapt to the varying activation distributions at different timesteps.

### 4.3 Comparison with State-of-the-Art Methods

We compare our proposed BI-DiffSR with recent binarization methods, including BNN [19], DoReFa [71], XNOR [50], IRNet [48], ReActNet [38], and BBCU [66]. To ensure a fair comparison, we set the parameters (Params) and complexity (OPs) of all binarization methods to be similar. We also compare our BI-DiffSR with the full-precision (FP) model, SR3 [54].

| Method           | Scale | Params<br>(M) | Ops<br>(G) | Set5  |        |        | B100  |        |        | Urban100 |        |        | Manga109 |        |        |
|------------------|-------|---------------|------------|-------|--------|--------|-------|--------|--------|----------|--------|--------|----------|--------|--------|
|                  |       |               |            | PSNR  | SSIM   | LPIPS  | PSNR  | SSIM   | LPIPS  | PSNR     | SSIM   | LPIPS  | PSNR     | SSIM   | LPIPS  |
| Bicubic          | ×2    | N/A           | N/A        | 33.67 | 0.9303 | 0.1274 | 29.55 | 0.8431 | 0.2508 | 26.87    | 0.8403 | 0.2064 | 30.82    | 0.9349 | 0.1025 |
| SR3 [54]         | ×2    | 55.41         | 176.41     | 36.69 | 0.9513 | 0.0310 | 30.41 | 0.8683 | 0.0700 | 30.29    | 0.9060 | 0.0430 | 35.11    | 0.9682 | 0.0161 |
| BNN [19]         | ×2    | 4.78          | 37.93      | 13.97 | 0.5210 | 0.4529 | 13.73 | 0.4553 | 0.5784 | 12.75    | 0.4236 | 0.5575 | 9.29     | 0.3035 | 0.7489 |
| DoReFa [71]      | ×2    | 4.78          | 37.93      | 16.43 | 0.6553 | 0.2662 | 16.11 | 0.5912 | 0.3972 | 15.09    | 0.5495 | 0.4055 | 12.35    | 0.4609 | 0.5047 |
| XNOR [50]        | ×2    | 4.78          | 37.93      | 32.34 | 0.8661 | 0.0782 | 27.94 | 0.7548 | 0.1665 | 27.47    | 0.8225 | 0.1153 | 31.99    | 0.9428 | 0.0326 |
| IRNet [48]       | ×2    | 4.78          | 37.93      | 32.55 | 0.9340 | 0.0446 | 27.76 | 0.8199 | 0.1115 | 26.34    | 0.8452 | 0.0913 | 23.89    | 0.7621 | 0.1820 |
| ReActNet [38]    | ×2    | 4.85          | 37.93      | 34.30 | 0.9271 | 0.0351 | 28.36 | 0.8158 | 0.0943 | 27.43    | 0.8563 | 0.0731 | 32.16    | 0.9441 | 0.0379 |
| BBCU [66]        | ×2    | 4.82          | 37.75      | 34.31 | 0.9281 | 0.0393 | 28.39 | 0.8202 | 0.0905 | 28.05    | 0.8669 | 0.0620 | 32.88    | 0.9508 | 0.0272 |
| BI-DiffSR (ours) | ×2    | 4.58          | 36.67      | 35.68 | 0.9414 | 0.0277 | 29.73 | 0.8478 | 0.0682 | 28.97    | 0.8815 | 0.0522 | 33.99    | 0.9601 | 0.0172 |
| Bicubic          | ×4    | N/A           | N/A        | 28.43 | 0.8111 | 0.3398 | 25.95 | 0.6678 | 0.5244 | 23.14    | 0.6579 | 0.4729 | 24.90    | 0.7876 | 0.3210 |
| SR3 [54]         | ×4    | 55.41         | 176.41     | 31.03 | 0.8798 | 0.1127 | 26.11 | 0.6933 | 0.2247 | 25.52    | 0.7702 | 0.1438 | 28.77    | 0.8854 | 0.0646 |
| BNN [19]         | ×4    | 4.78          | 37.93      | 12.21 | 0.3103 | 0.8310 | 12.30 | 0.2128 | 0.9519 | 11.30    | 0.2191 | 0.9592 | 8.96     | 0.1833 | 1.0117 |
| DoReFa [71]      | ×4    | 4.78          | 37.93      | 10.40 | 0.246  | 0.9855 | 9.78  | 0.1709 | 1.0793 | 8.79     | 0.1614 | 1.1186 | 7.52     | 0.1464 | 1.1169 |
| XNOR [50]        | ×4    | 4.78          | 37.93      | 28.06 | 0.8274 | 0.1381 | 25.25 | 0.6552 | 0.3101 | 23.13    | 0.6647 | 0.2564 | 23.84    | 0.7839 | 0.1559 |
| IRNet [48]       | ×4    | 4.78          | 37.93      | 15.52 | 0.3514 | 0.7548 | 16.38 | 0.3121 | 0.7072 | 15.23    | 0.3043 | 0.7068 | 11.82    | 0.2442 | 0.8354 |
| ReActNet [38]    | ×4    | 4.85          | 37.93      | 29.23 | 0.8362 | 0.1472 | 23.56 | 0.5670 | 0.3339 | 22.32    | 0.6440 | 0.2276 | 25.32    | 0.7854 | 0.1721 |
| BBCU [66]        | ×4    | 4.82          | 37.75      | 25.44 | 0.7795 | 0.1650 | 21.46 | 0.5472 | 0.3206 | 20.52    | 0.6293 | 0.2290 | 23.02    | 0.7966 | 0.1496 |
| BI-DiffSR (ours) | ×4    | 4.58          | 36.67      | 29.63 | 0.8374 | 0.1109 | 25.84 | 0.6779 | 0.2754 | 24.11    | 0.7177 | 0.1823 | 26.95    | 0.8548 | 0.0889 |

Table 2: Quantitative comparison with state-of-the-art binarization methods. The best and second best results are coloured with **red** and **blue**. Our method surpasses current approaches.

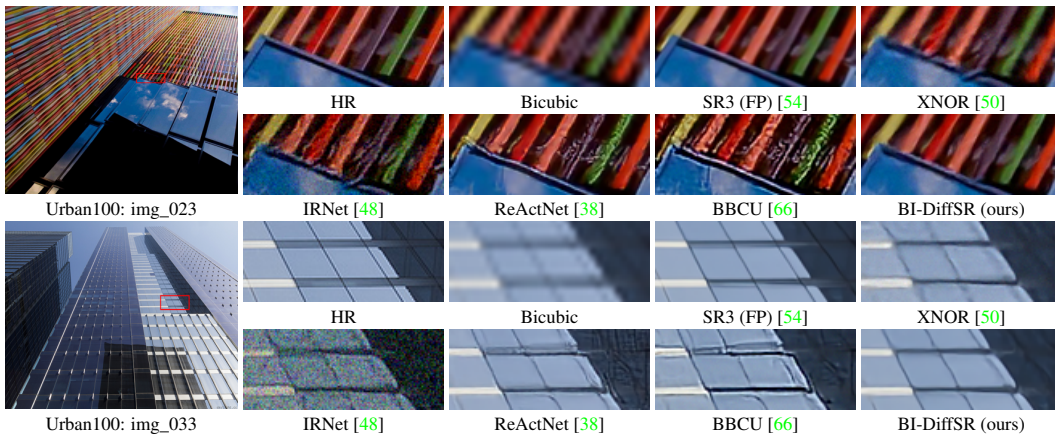


Figure 8: Visual comparison (×4) in some challenge cases.

**Quantitative Results.** We provide the quantitative comparisons in Tab. 2. We test OPs of single-step sampling on the output size  $3 \times 256 \times 256$ . Compared to other binarization methods, our BI-DiffSR achieves the best performance. Specifically, on Urban100 and Manga109 (×2), BI-DiffSR surpasses the second-best method, BBCU, with a PSNR gain of **0.92** and **1.11** dB, respectively. Moreover, compared to the full-precision model, SR3, our method achieves comparable or even better perceptual performance with only 8.3% Params and 20.8% OPs. For instance, BI-DiffSR achieves 93.6% LPIPS results of SR3 on Manga109. These results demonstrate the superiority of our method.

**Visual Results.** We present visual comparisons (×4) in Fig. 8. Previous binarization methods struggle to recover image details in challenging cases. In contrast, our method can restore clearer results with more texture details. Meanwhile, the difference between our BI-DiffSR and the full-precision model results is small. More visual results are provided in the supplementary material.

## 5 Conclusion

In this paper, we propose the BI-DiffSR, a novel binarized diffusion model for image SR. Specifically, we first design the UNet structure suitable for binarization. To ensure dimension consistency and full-precision information transfer, we design the consistent-pixel-downsample (CP-Down) and upsample (CP-Up). Meanwhile, we develop the channel-shuffle-fusion (CS-Fusion) to enhance information fusion within the skip connection. Furthermore, in response to the multi-step mechanism of diffusion models, we design the timestep-aware redistribution (TaR) and activation functions (TaA) to adapt to the varying activation distributions. The TaR and TaA enhance the representational capabilities of the binarized modules under multiple timesteps. Extensive experiments indicate that our method outperforms current binarization methods, and achieves comparable perceptual performance to the full-precision model, demonstrating substantial potential.

**Acknowledgments.** This work is supported by the National Natural Science Foundation of China (62141220, 62271414), Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102), the Fundamental Research Funds for the Central Universities, Zhejiang Provincial Distinguish Young Science Foundation (LR23F010001), Zhejiang “Pioneer” and “Leading Goose” R&D Program (2024SDXHDX0006, 2024C03182), the Key Project of Westlake Institute for Optoelectronics (2023GD007), and Ningbo Science and Technology Bureau, “Science and Technology Yongjiang 2035” Key Technology Breakthrough Program (2024Z126).

## References

- [1] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. 6
- [2] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie Line Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In *BMVC*, 2012. 7
- [3] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation. In *ICLR*, 2020. 3
- [4] Weimin Chen, Yuqing Ma, Xianglong Liu, and Yi Yuan. Hierarchical generative adversarial networks for single image super-resolution. In *CVPR*, 2021. 3
- [5] Zheng Chen, Yulun Zhang, Jinjin Gu, Linghe Kong, Xiaokang Yang, and Fisher Yu. Dual aggregation transformer for image super-resolution. In *ICCV*, 2023. 3
- [6] Zheng Chen, Yulun Zhang, Jinjin Gu, Yongbing Zhang, Linghe Kong, and Xin Yuan. Cross aggregation transformer for image restoration. In *NeurIPS*, 2022. 1
- [7] Zheng Chen, Yulun Zhang, Ding Liu, Bin Xia, Jinjin Gu, Linghe Kong, and Xin Yuan. Hierarchical integration diffusion model for realistic image deblurring. In *NeurIPS*, 2023. 3
- [8] Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. Diffusion models in vision: A survey. *TPAMI*, 2023. 3
- [9] Ryan Dahl, Mohammad Norouzi, and Jonathon Shlens. Pixel recursive super resolution. In *ICCV*, 2017. 1, 3
- [10] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *TPAMI*, 2016. 1, 3
- [11] Gongfan Fang, Xinyin Ma, and Xinchao Wang. Structural pruning for diffusion models. In *NeurIPS*, 2024. 2
- [12] Abhijay Ghildyal and Feng Liu. Shift-tolerant perceptual similarity metric. In *ECCV*, 2022. 7
- [13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 3
- [14] Chunming He, Chengyu Fang, Yulun Zhang, Kai Li, Longxiang Tang, Chenyu You, Fengyang Xiao, Zhenhua Guo, and Xiu Li. Reti-diff: Illumination degradation image restoration with retinex-based latent diffusion model. *arXiv preprint arXiv:2311.11638*, 2023. 3
- [15] Chunming He, Yuqi Shen, Chengyu Fang, Fengyang Xiao, Longxiang Tang, Yulun Zhang, Wangmeng Zuo, Zhenhua Guo, and Xiu Li. Diffusion models in low-level vision: A survey. *arXiv preprint arXiv:2406.11138*, 2024. 3
- [16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. 1, 3
- [17] Shengchao Hu, Li Chen, Penghao Wu, Hongyang Li, Junchi Yan, and Dacheng Tao. St-p3: End-to-end vision-based autonomous driving via spatial-temporal feature learning. In *ECCV*, 2022. 3
- [18] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *CVPR*, 2015. 7
- [19] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In *NeurIPS*, 2016. 2, 3, 8, 9

- [20] Xinrui Jiang, Nannan Wang, Jingwei Xin, Keyu Li, Xi Yang, and Xinbo Gao. Training binary neural network without batch normalization for image super-resolution. In *AAAI*, 2021. 3
- [21] Bahjat Kawar, Jiaming Song, Stefano Ermon, and Michael Elad. Jpeg artifact correction using denoising diffusion restoration models. In *NeurIPS Workshop*, 2022. 3
- [22] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 7
- [23] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014. 3
- [24] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 2017. 3
- [25] Haoying Li, Yifan Yang, Meng Chang, Shiqi Chen, Huajun Feng, Zhihai Xu, Qi Li, and Yueting Chen. Srdiff: Single image super-resolution with diffusion probabilistic models. *Neurocomputing*, 2022. 3
- [26] Xiuyu Li, Yijiang Liu, Long Lian, Huanrui Yang, Zhen Dong, Daniel Kang, Shanghang Zhang, and Kurt Keutzer. Q-diffusion: Quantizing diffusion models. In *ICCV*, 2023. 2, 3
- [27] Yanjing Li, Sheng Xu, Xianbin Cao, Xiao Sun, and Baochang Zhang. Q-dm: An efficient low-bit quantized diffusion model. In *NeurIPS*, 2023. 3
- [28] Yanyu Li, Huan Wang, Qing Jin, Ju Hu, Pavlo Chemerys, Yun Fu, Yanzhi Wang, Sergey Tulyakov, and Jian Ren. Snapfusion: Text-to-image diffusion model on mobile devices within two seconds. In *NeurIPS*, 2024. 3
- [29] Yuchen Li, Haoyi Xiong, Linghe Kong, Zeyi Sun, Hongyang Chen, Shuaiqiang Wang, and Dawei Yin. Mpgraf: a modular and pre-trained graphformer for learning to rank at web-scale. In *ICDM*, 2023. 3
- [30] Yuchen Li, Haoyi Xiong, Linghe Kong, Rui Zhang, Fanqin Xu, Guihai Chen, and Minglu Li. Mhrr: Moocs recommender service with meta hierarchical reinforced ranking. *TSC*, 2023. 3
- [31] Jingyun Liang, Jiezhong Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *ICCVW*, 2021. 1, 3
- [32] Jingyun Liang, Andreas Lugmayr, Kai Zhang, Martin Danelljan, Luc Van Gool, and Radu Timofte. Hierarchical conditional flow: A unified framework for image super-resolution and image rescaling. In *ICCV*, 2021. 3
- [33] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *CVPRW*, 2017. 3, 7
- [34] Xinqi Lin, Jingwen He, Ziyang Chen, Zhaoyang Lyu, Ben Fei, Bo Dai, Wanli Ouyang, Yu Qiao, and Chao Dong. Diffbir: Towards blind image restoration with generative diffusion prior. In *ECCV*, 2024. 3
- [35] Chang Liu, Haoning Wu, Yujie Zhong, Xiaoyun Zhang, Yanfeng Wang, and Weidi Xie. Intelligent grimm-open-ended visual storytelling via latent diffusion models. In *CVPR*, 2024. 3
- [36] Chang Liu, Haoning Wu, Yujie Zhong, Xiaoyun Zhang, and Weidi Xie. Intelligent grimm–open-ended visual storytelling via latent diffusion models. In *CVPR*, 2024. 3
- [37] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. In *ICLR*, 2022. 1, 3
- [38] Zechun Liu, Zhiqiang Shen, Marios Savvides, and Kwang-Ting Cheng. Reactnet: Towards precise binary neural network with generalized activation functions. In *ECCV*, 2020. 2, 3, 6, 8, 9
- [39] Zechun Liu, Baoyuan Wu, Wenhan Luo, Xin Yang, Wei Liu, and Kwang-Ting Cheng. Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In *ECCV*, 2018. 2, 3
- [40] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. In *NeurIPS*, 2022. 1
- [41] Andreas Lugmayr, Martin Danelljan, Luc Van Gool, and Radu Timofte. Srflo: Learning the super-resolution space with normalizing flow. In *ECCV*, 2020. 3

- [42] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *ICCV*, 2001. 7
- [43] Yusuke Matsui, Kota Ito, Yuji Aramaki, Azuma Fujimoto, Toru Ogawa, Toshihiko Yamasaki, and Kiyoharu Aizawa. Sketch-based manga retrieval using manga109 dataset. *MTAP*, 2017. 7
- [44] Sachit Menon, Alexandru Damian, Shijia Hu, Nikhil Ravi, and Cynthia Rudin. Pulse: Self-supervised photo upsampling via latent space exploration of generative models. In *CVPR*, 2020. 1
- [45] Markus Nagel, Marios Fournarakis, Rana Ali Amjad, Yelysei Bondarenko, Mart Van Baalen, and Tijmen Blankevoort. A white paper on neural network quantization. *arXiv preprint arXiv:2106.08295*, 2021. 2
- [46] Ben Niu, Weilei Wen, Wenqi Ren, Xiangde Zhang, Lianping Yang, Shuzhen Wang, Kaihao Zhang, Xiaochun Cao, and Haifeng Shen. Single image super-resolution via a holistic attention network. In *ECCV*, 2020. 3
- [47] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 7
- [48] Haotong Qin, Ruihao Gong, Xianglong Liu, Mingzhu Shen, Ziran Wei, Fengwei Yu, and Jingkuan Song. Forward and backward information retention for accurate binary neural networks. In *CVPR*, 2020. 2, 3, 8, 9
- [49] Haotong Qin, Mingyuan Zhang, Yifu Ding, Aoyu Li, Zhongang Cai, Ziwei Liu, Fisher Yu, and Xianglong Liu. Bibench: Benchmarking and analyzing network binarization. In *ICML*, 2023. 2, 3, 7
- [50] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *ECCV*, 2016. 2, 6, 8, 9
- [51] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *ICML*, 2015. 3
- [52] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 1, 3
- [53] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 2, 3
- [54] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *TPAMI*, 2022. 1, 2, 3, 8, 9
- [55] Yuzhang Shang, Zhihang Yuan, Bin Xie, Bingzhe Wu, and Yan Yan. Post-training quantization on diffusion models. In *CVPR*, 2023. 3
- [56] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017. 7
- [57] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, 2016. 5
- [58] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2020. 1, 3
- [59] Radu Timofte, Eirikur Agustsson, Luc Van Gool, Ming-Hsuan Yang, Lei Zhang, Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, Kyoung Mu Lee, et al. Ntire 2017 challenge on single image super-resolution: Methods and results. In *CVPRW*, 2017. 7
- [60] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 4
- [61] Huan Wang, Suhas Lohit, Michael N Jones, and Yun Fu. What makes a "good" data augmentation in knowledge distillation-a statistical perspective. In *ICLR*, 2022. 2
- [62] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *ECCVW*, 2018. 1, 3

- [63] Yinhuai Wang, Jiwen Yu, and Jian Zhang. Zero-shot image restoration using denoising diffusion null-space model. In *ICLR*, 2023. 1, 3
- [64] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *TIP*, 2004. 7
- [65] Rongyuan Wu, Tao Yang, Lingchen Sun, Zhengqiang Zhang, Shuai Li, and Lei Zhang. Seesr: Towards semantics-aware real-world image super-resolution. In *CVPR*, 2024. 3
- [66] Bin Xia, Yulun Zhang, Yitong Wang, Yapeng Tian, Wenming Yang, Radu Timofte, and Luc Van Gool. Basic binary convolution unit for binarized image restoration network. In *ICLR*, 2022. 2, 3, 7, 8, 9
- [67] Yixing Xu, Kai Han, Chang Xu, Yehui Tang, Chunjing Xu, and Yunhe Wang. Learning frequency domain approximation for binary neural networks. In *NeurIPS*, 2021. 3
- [68] Eduard Zamfir, Zongwei Wu, Nancy Mehta, Yulun Zhang, and Radu Timofte. See more details: Efficient image super-resolution by experts mining. In *ICML*, 2024. 1
- [69] Yulun Zhang, Haotong Qin, Zixiang Zhao, Xianglong Liu, Martin Danelljan, and Fisher Yu. Flexible residual binarization for image super-resolution. In *ICML*, 2024. 3
- [70] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *CVPR*, 2018. 1, 3
- [71] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *ICLR*, 2016. 2, 8, 9

## NeurIPS Paper Checklist

### 1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: Please refer to our abstract and introduction.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

### 2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: We discuss the limitations in the supplementary file.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

### 3. Theory Assumptions and Proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[NA\]](#)

Justification: The paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

#### 4. Experimental Result Reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We have provided implementation details in the experiments section. We release all the code and models at: <https://github.com/zhengchen1999/BI-DiffSR>.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

#### 5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide the code and pre-trained models at: <https://github.com/zhengchen1999/BI-DiffSR>.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

## 6. Experimental Setting/Details

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We have provided implementation details, which cover the above questions.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

## 7. Experiment Statistical Significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Please refer to the experiment part.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

## 8. Experiments Compute Resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [\[Yes\]](#)

Justification: Please refer to experiment part.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

## 9. Code Of Ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [\[Yes\]](#)

Justification: The research conducted in the paper conforms, in every respect, with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

## 10. Broader Impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [\[Yes\]](#)

Justification: Please refer to the supplementary file.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

#### 11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This work poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

#### 12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We have credited most previous works in the paper. The license and terms are respected properly.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, [paperswithcode.com/datasets](https://paperswithcode.com/datasets) has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

### 13. New Assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We release code and pre-trained models at: <https://github.com/zhengchen1999/BI-DiffSR>. In the paper, we have provided implementation details and other contents to reproduce our results.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

### 14. Crowdsourcing and Research with Human Subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

### 15. Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.