Dynamic Dual Trainable Bounds for Ultra-low Precision Super-Resolution Networks

Yunshan Zhong[1], Mingbao Lin[2,3], Xunchao Li[2], Ke Li[3],
Yunhang Shen[3], Fei Chao[1,2], Yongjian Wu[3], Rongrong Ji[1,2*]

[1]Institute of Artificial Intelligence, Xiamen University
[2]MAC Lab, Department of Artificial Intelligence, Xiamen University
[3]Tencent Youtu Lab
{zhongyunshan, lmbxmu, lixunchao}@stu.xmu.edu.cn,
{tristanli.sh, shenyunhang01}@gmail.com,
fchao@xmu.edu.cn, littlekenwu@tencent.com, rrji@xmu.edu.cn

Abstract. Light-weight super-resolution (SR) models have received considerable attention for their serviceability in mobile devices. Many efforts employ network quantization to compress SR models. However, these methods suffer from severe performance degradation when quantizing the SR models to ultra-low precision (*e.g.*, 2-bit and 3-bit) with the low-cost layer-wise quantizer. In this paper, we identify that the performance drop comes from the contradiction between the layer-wise symmetric quantizer and the highly asymmetric activation distribution in SR models. This discrepancy leads to either a waste on the quantization levels or detail loss in reconstructed images. Therefore, we propose a novel activation quantizer, referred to as Dynamic Dual Trainable Bounds (DDTB), to accommodate the asymmetry of the activations. Specifically, DDTB innovates in: 1) A layer-wise quantizer with trainable upper and lower bounds to tackle the highly asymmetric activations. 2) A dynamic gate controller to adaptively adjust the upper and lower bounds at runtime to overcome the drastically varying activation ranges over different samples. To reduce the extra overhead, the dynamic gate controller is quantized to 2-bit and applied to only part of the SR networks according to the introduced dynamic intensity. Extensive experiments demonstrate that our DDTB exhibits significant performance improvements in ultra-low precision. For example, our DDTB achieves a 0.70dB PSNR increase on Urban100 benchmark when quantizing EDSR to 2-bit and scaling up output images to ×4. Code is at https://github.com/zysxmu/DDTB.

1 Introduction

Single image super-resolution (SISR) is a classic yet challenging research topic in low-level computer vision. It aims to construct a high-resolution (HR) image from a given low-resolution (LR) image. Recent years have witnessed the revolution of deep convolutional neural networks (DCNN), which leads to many state-of-the-arts [24,46,5] in SISR task.

When looking back on the development of DCNN in SISR, we find that the record-breaking performance is accompanied by a drastically increasing model
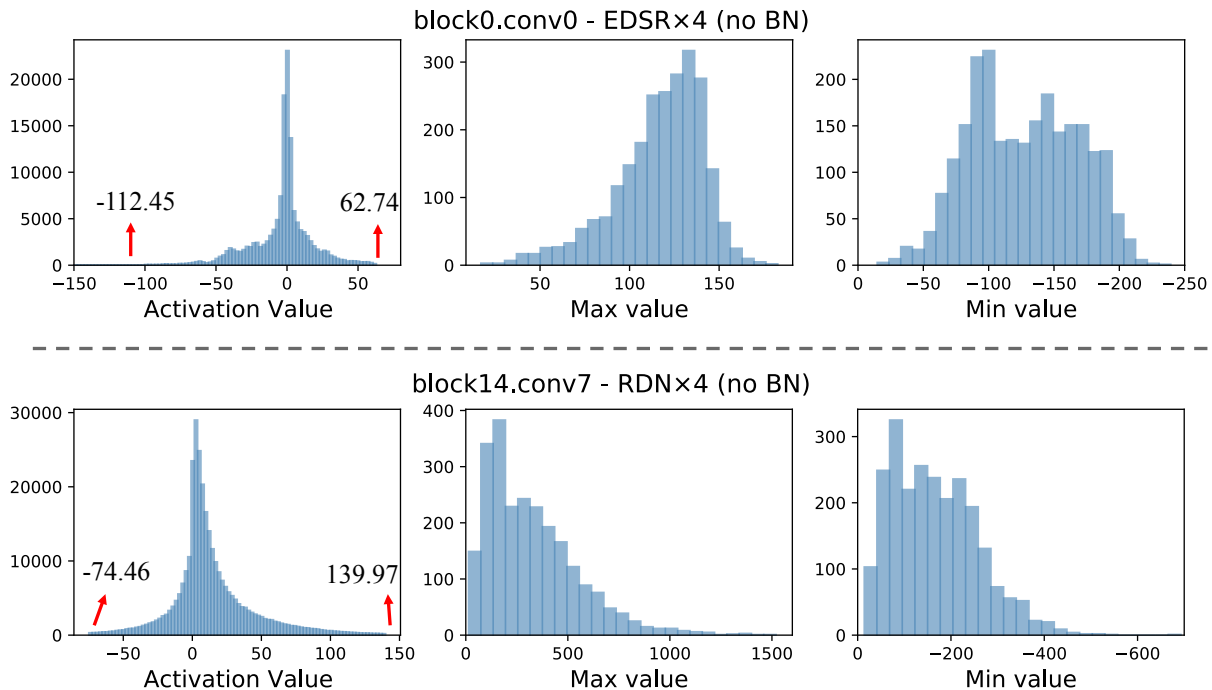
---

* Corresponding Author

Fig. 1. The first column shows the activation histograms. The second and third columns show the maximum and minimum activation values of different samples. We perform experiments with EDSR [24] and RDN [46] on DIV2K [37] dataset.

complexity. SRCNN [5], the first work to integrate DCNN to SR, has only three convolutional layers with a total of 57K parameters. Then, EDSR [24] constructs a 64-layer CNN with 1.5M parameters. Equipped with a residual dense block, RDN [46] introduces 151 convolutional layers with 22M parameters. Also, it requires around 5,896G float-point operations (FLOPs) to produce only one 1920×1080 image (upscaling factor ×4). On the one hand, the high memory footprint and computation cost of DCNN-based SR models barricade their deployment on many resource-hungry platforms such as smartphones, wearable gadgets, embedding devices, *etc.* On the other hand, SR is particularly popular on these devices where the photograph resolution must be enhanced after being taken by the users. Therefore, compressing DCNN-based SR models has gained considerable attention from both academia and industries. In recent years, a variety of methodologies are explored to realize practical deployment [25,20,10,8].

By discretizing the full-precision weights and activations within the DCNN, network quantization has emerged as one of the most promising technologies. It reduces not only memory storage for lower-precision representation but computation cost for more efficient integer operations. Earlier studies mostly focus on high-level vision tasks, such as classification [20,7,34,26] and segmentation [43,44]. A direct extension of these methods to SR networks has been proved infeasible since low-vision networks often have different operators with these high-level networks [23]. Consequently, excavating specialized quantization methods for DCNN-based SR models recently has aroused increasing attention in the research community. For example, PAMS [23] designs a layer-wise quantizer with a learnable clipping to tackle the large ranges of activations, but severe performance degradation occurs in ultra-low precision settings (*e.g.*, 2-bit and 3-bit) as shown in Sec. 4.2. A recent study DAQ [11] adopts a channel-wise distribution-aware quantization scheme. Despite the progress, the performance improvement
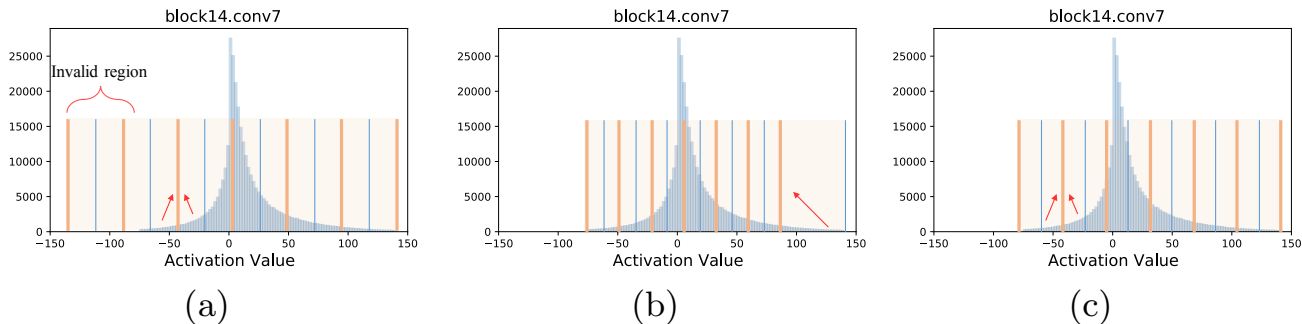
Fig. 2. Example of "quantization unfitness" using RDN [46] on DIV2K [37]. The orange bar denotes the quantization levels. The red arrows show the quantization level of activations. (a) Two quantization levels are wasted on the region without activations. (b) High-magnitude activations are quantized to a small quantization level. (c) The distribution of the quantization levels from dual bounds in our DDTB.

comes at the cost of considerable overhead from normalizing and de-normalizing feature maps, as well as the expensive per-channel quantizer. Therefore, existing studies are stuck in either heavy extra costs or severe performance drops when performing ultra-low precision quantization.

In this paper, we realize that the inapplicability of existing studies comes from the contradiction between the layer-wise symmetric quantizer and the asymmetric activation distribution in DCNN-based SR models. Specifically, it has been a wide consensus [24,46,45,9] that removing batch normalization (BN) layers increases the super-resolution performance since low-level vision is sensitive to the scale information of images while BN reduces the range flexibility of activations. However, the removal of BN leads to highly asymmetric activation distribution, as well as diverse maximum and minimum activations for different input samples. An illustrative example with EDSR [24], RDN [46] on DIV2K [37] dataset is given in Fig. 1. Note that many previous works equip SR networks with BN layers, which, however, have to compensate for the performance drops by other strategies. For example, SRResNet [22] constructs a highway that propagates the activations of the first convolutional layer to the outputs of every other block, which again leads to activation distributions like Fig. 1.

Although these abnormal activation distributions benefit a full-precision SR network, they are unfriendly to the quantized version which often constructs a symmetric quantizer with only one clipping bound to perform network quantization, resulting in the issue of "quantization unfitness". Fig. 2 illustrates a toy example. With an asymmetric activation distribution, a large clipping wastes two quantization levels on the region without any activation items, while a small clipping quantizes large-magnitude activations to a small quantization level which leads to large quantization error and causes details loss in high-resolution images. Thus, an adaptive quantizer to the activation distribution is crucial to the quality of the reconstructed high-resolution images.

Motivated by the above analysis, in this paper, we propose dynamic dual trainable bounds (DDTB) to quantize the activations in SR models. Our DDTB innovates in: 1) A layer-wise quantizer with trainable upper and lower bounds to tackle the highly asymmetric activations; 2) A dynamic gate controller to adaptively adjust the upper and lower bounds based on different input samples

at runtime for overcoming the drastically varying activation ranges over different samples. To minimize the extra costs on computation and storage, the dynamic gate is represented in a 2-bit format and applied to part of the network according to the dynamic intensity in each layer. Also, we introduce an initializer to provide a warmup for DDTB. Specifically, we first use the activation statistics from the full-precision network instead of the quantized version to initialize the upper and lower bounds. Then, the dynamic gate controller is trained individually towards its target output of 1 for all inputs in the early several epochs. Combing with the proposed initializer, DDTB provides significant performance improvements, especially when SR models are quantized to the case of ultra-low bits. For instance, compared with the state-of-the-art, our DDTB achieves performance gains by 0.70dB PSNR on Urban100 benchmark [12] when EDSR [24] is quantized to 2-bit and outputs ×4 reconstructed images.

2   Related Work

2.1   Single Image Super Resolution

Owing to the strength of deep convolutional neural networks, DCNN-based SR methods have gained great performance boosts and dominated the field of SISR. SRCNN [5] is the first to construct an end-to-end CNN-based mapping between the low- and high-resolution images. By increasing the network depth, VDSR [17] manifests significant performance improvements. Nevertheless, the increasing depth also weakens the capacity of overcoming the gradient vanishing problem and retaining image detail. Consequently, the skip-connection based blocks, such as the residual block [22] and the dense block [38], have become a basic component of the SR models [24,46]. Also, many other complex network structures like channel attention mechanism [45,29] and non-local attention [32,31] are also integrated into SR for a better performance.

With the increasing popularity of CNNs on resource-hungry devices, developing efficient SR models has aroused much attention recently. Most works in this area are indicated to devising lightweight network architectures. For example, DRCN [18] and DRRN [36] adopt the recursive structure to increase the network depth while mitigating the model parameters. To escape from the expensive up-sampling operator, FSRCNN [6] introduces a de-convolutional layer while ESPCN [35] devises a sub-pixel convolution module. Many others resort to enhance the efficiency of intermediate feature representation [21,1,13,27]. Nevertheless, these computation savings are very limited compared to costs from the full-precision convolution.

2.2   Quantized SR Models

As a promising technique to compress SR models, network quantization has received ever-growing attention [28,42,15,11,23,40]. Ma *et al.* [28] pioneered 1-bit quantization over the weights of SR model. Performance drops severely if binarizing the activations as well. To remedy this issue, the structure of SR models is

often adjusted. For example, BAM [42] and BTM [15] introduce multiple feature map aggregations and skip connection layers.

Except for 1-bit quantization, other ultra-low quantization precision such as 2-bit, 3-bit and 4-bit is discussed in many studies as well. To handle the unstable activation ranges, Li [23] proposed a symmetric layer-wise linear quantizer that adopts a trainable clipping bound to clamp the abnormal activations. As for weights, the same symmetric quantizer is adopted but the clipping variable is simply set to the maximum magnitude of the weights. Moreover, the quantized model is enhanced by the structured knowledge transfer from its full-precision counterpart. Wang *et al.* [40] chose to quantize all layers of SR models and perform both weights and activations quantization using a symmetric layer-wise quantizer equipped with a trainable clipping variable. DAQ [11] observes that each channel has non-zero distributions and the activation values also vary drastically to the input image. Based on this observation, a channel-wise distribution-aware quantizer is adopted where the activations are normalized before discretizing and de-normalized after convolution.

3 Methodology

3.1 Preliminaries

Previous low-bit SR adopt a symmetric quantizer to perform quantization upon the premise of activations and weights with a symmetric distribution. Specifically, denoting b as the bit-width, $\boldsymbol{x}$ as weights or activations, α as the clipping bound, the symmetric linear quantizer is defined as:

$$\bar{\boldsymbol{x}} = round\big(\frac{clip(\boldsymbol{x}, \alpha)}{s}\big) \cdot s, \tag{1}$$

where $clip(\boldsymbol{x}, \alpha) = min\big(max(\boldsymbol{x}, -\alpha), \alpha\big)$, $\bar{\boldsymbol{x}}$ is the corresponding de-quantized value of $\boldsymbol{x}$, $round(\cdot)$ rounds its input to the nearest integer and s denotes the scaling factor that projects a floating-point number to a fixed-point integer which can be calculated as $s = \frac{2\alpha}{2^{b-1}-1}$.

An appropriate clipping bound is crucial to the quantization performance since it not only eliminates outlier activations but refers to retaining details in the reconstructed images. To this end, PAMS [23] regards α as a learnable variable in quantizing activations, while setting α to the maximum magnitude of the weights in quantizing weights.

3.2 Our Insights

Despite the progress, the performance of earlier low-bit SR quantization methods [23,11,40] remains an open issue when quantizing the full-precision counterpart to ultra-low precision with a layer-wise quantizer. After an in-depth analysis, we attribute the poor performance to the contradiction between the layer-wise symmetric quantizer and the asymmetric activation distribution in SR models.

Concretely, we observe the activations from different SR models. Fig. 1 shows the histograms, maximum and minimum of activations collected from the pre-trained EDSR [24] and RDN [46] on DIV2K [37] benchmark. From the first column of Fig. 1, we realize that the activations of SR models are indeed highly asymmetric in an irregular state. For example, the magnitude of the minimum activation is almost twice that of the maximum activation in EDSR. A similar phenomenon can be observed in RDN. Furthermore, as shown in the second and third columns of Fig. 1, the maximum and minimum of activations also vary drastically for different samples. Such an asymmetric activation distribution mostly results from the removal of BN layers in modern SR networks since BN reduces the scale information of images which however is crucial to SR tasks. Though some studies retain BN layers, however, other remedies have to be taken to regain the performance. For example, SRResNet [22] propagates outputs of the first layer to the outputs of all following blocks, which indeed causes asymmetric distributions as well. It is natural that the symmetric quantizer in existing studies cannot well fit the symmetric activation distributions, which we term as "quantization unfitness" in this paper.

To be specific, using only one clipping bound fails to handle the symmetric activations, whatever the value of α. To demonstrate this, we quantize the activations of RDN [46] to 3-bit using the quantizer in Eq. (1). As displayed in Fig. 2(a), the lack of activations along the negative axis direction causes a waste of two quantization levels if α is set to a large value such as the maximum of the activation magnitude. The wastes are even more severe as the bit-width goes down. Taking the activations in Fig. 2(a) as an example, we experimentally observe 37.5% are wasted in 3-bit quantization, while it increases sharply to 50% in 2-bit quantization. When it comes to a small α such as the absolute value of the minimum of activation magnitude, as illustrated in Fig. 2(b), though avoiding the waste on quantization levels, only the small-magnitude activations are covered, leading to large quantization error since many high-magnitude activations are quantized to a small quantization level, *i.e.*, α. Recall that SR models are sensitive to the scale information of images. Consequently, representing the large full-precision activations with a small quantization level inevitably brings about detail loss in the reconstructed high-resolution images, leading to significant quality degradation. Similar observations can be found if the same clipping bound is applied to all the input images since the maximum and minimum activations over different images also drastically vary as illustrated in Fig. 1.

Overall, an appropriate quantizer is vital to the final performance of a quantized SR model.

3.3   Our Solutions

In the following, we first detail dynamic dual trainable bounds (DDTB) specifically designed to quantize activations of SR models. Then, we elaborate on our initializer for DDTB. Finally, we describe the quantizer for weights. The overall computational graph is presented in Fig. 3(a).
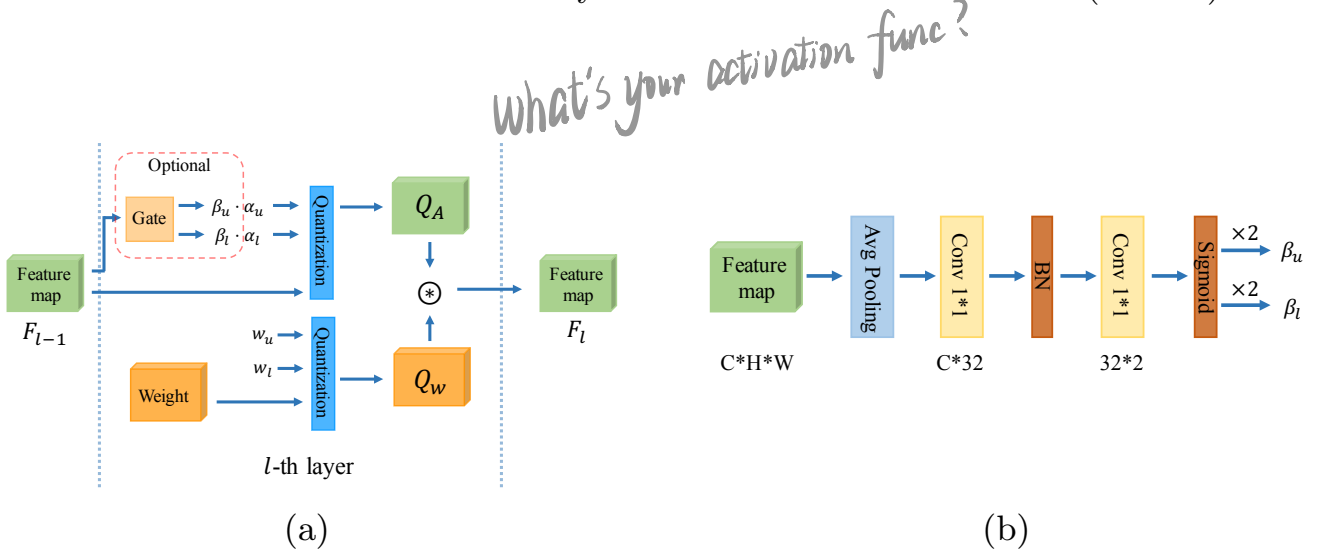
what's your activation func?

Fig. 3. (a) Computational graph in the *l*-th quantized convolutional layer. (b) Structure of the dynamic gate controller.

Activation Quantization. Our DDTB consists of two parts: 1) A layer-wise quantizer with a trainable upper bound and a trainable lower bound; 2) A dynamic gate controller with adaptive upper and lower bounds to the inputs.

Dual trainable bounds: As mentioned in Sec. 3.2, only using one clipping variable to clamp the asymmetric activations leads to the "quantization unfitness" problem. To address this, we introduce two trainable clipping variables of α_u and α_l to respectively determine the upper bound and the lower bound of the activations. Equipped with these two dual trainable clipping variables, the quantized integer q for the input x can be obtained as:

$$q = round\left(\frac{clip(x, \alpha_l, \alpha_u)}{s}\right) + Z, \tag{2}$$

where $clip(x, \alpha_l, \alpha_u) = min\left(max(x, \alpha_l), \alpha_u\right)$, $Z = round(\frac{-\alpha_l}{s})$ is a zero-point integer corresponding to the real value 0. Accordingly, the scaling factor s is calculated as: $s = \frac{\alpha_u - \alpha_l}{2^b - 1}$. Finally, the de-quantized value $\bar{a}$ is:

$$\bar{a} = (q - Z) \cdot s. \tag{3}$$

Combining Eq. (2) and Eq. (3) completes our activation quantizer. As depicted in Fig. 2(c), the quantization levels generated by this quantization scheme well fit the activations when α_u and α_l are set to the maximum and minimum of activations, respectively. In order to accommodate samples with drastically varying ranges, we employ the stochastic gradient descent to adaptively learn α_u and α_l by minimizing the finally loss function. Denoting the clipped activations as $\tilde{a} = min(max(a, \alpha_l), \alpha_u)$, $\frac{\partial \bar{a}}{\partial \tilde{a}}$ is set to 1 by using the straight-through estimator (STE) [4]. Then, the gradients of α_u, α_l can be calculated as:

$$\frac{\partial \bar{a}}{\partial \alpha_u} = \frac{\partial \bar{a}}{\partial \tilde{a}} \frac{\partial \tilde{a}}{\partial \alpha_u} \overset{STE}{\approx} \begin{cases} 1, & a \geq \alpha_u \\ 0, & a \leq \alpha_u \end{cases}, \quad \frac{\partial \bar{a}}{\partial \alpha_l} = \frac{\partial \bar{a}}{\partial \tilde{a}} \frac{\partial \tilde{a}}{\partial \alpha_l} \overset{STE}{\approx} \begin{cases} 0, & a \geq \alpha_l \\ 1, & a \leq \alpha_l \end{cases}. \tag{4}$$

With the dual α_l and α_u, "quantization unfitness" can be alleviated. Note that, early study [16] also introduces two trainable parameters to determine the quantization range. This paper differs in: 1) [16] utilizes the center and radius of

the quantization region to parameterize the quantization range while we explicitly introduce the upper and lower bounds. 2) [16] requires an expensive transformation to normalize activations to a fixed range before performing quantization. 3) Though trainable, the final quantization range is consistent with different inputs. Thus, we design a dynamic gate controller to provide an instance-wise quantization range adaptive to different inputs as illustrated in the next section.

Dynamic gate controller: Though the trainable upper and lower bounds partially alleviate the "quantization unfitness" problem, it is still suboptimal if only applying the same pair of (α_l, α_u) for different inputs since their activation ranges also dramatically vary as shown in Fig. 1.

To well address this issue, we devise a dynamic gate controller to adapt (α_l, α_u) to every input sample at runtime. Fig. 3(b) depicts the structure of our gate controller, which is composed of a series of consequently stacked operators including two convolutional layers, a BN layer, a ReLU function, and a sigmoid function multiplied by 2. It takes the feature maps from the SR model as inputs and outputs two scaling coefficients β_l and β_u for each input image. Then, β_l and β_u are respectively used to re-scale the lower bound α_l and the upper bound α_u, results of which serve as the final clipping bounds of the corresponding input images as shown in Fig. 3(a). Denoting the $\alpha_u' = \beta_u \cdot \alpha_u$ and $\alpha_l' = \beta_l \cdot \alpha_l$, the clipped activations $\tilde{a}$ now can be reformulated as: $\tilde{a} = min(max(a, \alpha_l'), \alpha_u')$. Similar to Eq. (4), the gradient can be calculated as:

$$\frac{\partial \bar{a}}{\partial \alpha_u'} = \frac{\partial \bar{a}}{\partial \tilde{a}} \frac{\partial \tilde{a}}{\partial \alpha_u'} \overset{STE}{\approx} \begin{cases} 1, & a \geq \alpha_u' \\ 0, & a \leq \alpha_u' \end{cases}, \quad \frac{\partial \bar{a}}{\partial \alpha_l'} = \frac{\partial \bar{a}}{\partial \tilde{a}} \frac{\partial \tilde{a}}{\partial \alpha_l'} \overset{STE}{\approx} \begin{cases} 0, & a \geq \alpha_l' \\ 1, & a \leq \alpha_l' \end{cases}. \quad (5)$$

Then, the gradients of $\beta_u, \beta_l, \alpha_u, \alpha_l$ can be obtained by the chain rule. With these gradients, the gate and clipping bounds can be trained together.

Our gate controller enables the clipping bounds adaptive to each of the input samples, with which the gate outputs are dynamically correlated at runtime. Nevertheless, the extra costs on computation and storage from the introduced gate controller are expensive, causing a contradiction with our motive to reduce the complexity of SR models. Consequently, we reduce the gate complexity from two perspectives: 1) We quantize the weights and activations of the gate to a 2-bit using the quantizer defined by Eq. (2) and Eq. (3). The clipping bounds are simply set as the maximum and minimum of the weights/activations. We empirically find that such a simple setting is sufficient. During the inference phase, the BN layer in our gate controller can be folded into the previous convolutional layer [40]. Therefore, the extra floating-point operations are very cheap in this situation. 2) We choose to apply our dynamic gate controller to some layers of the SR model. Concretely, we define the dynamic intensity in the l-th layer as $DI^l = V_{max}^l + V_{min}^l$, where V_{max}^l and V_{min}^l are variances of the maximum activations and minimum activations. The overhead is trivial as it forwards the training data to the full-precision network only once and can be completed offline. It is intuitive that a larger DI^l indicates a more dynamic activation range. Thus, we apply the dynamic gate to these layers with the top-$P\%$ largest dynamic intensity, where the gate ratio P is a hyper-parameter.

DDTB Initializer. One common way to initialize the α_u and α_l is to use the activation statistics such as the maximum and minimum values in the quantized network. However, for ultra-low bit cases, quantization error accumulates drastically, leading to unreliable statistics in deep layers. Instead, we derive the activation statistics by feeding a patch of images to the full-precision network, and then use the M-th and $(100 - M)$-th percentiles of obtained activations to initialize the α_u and α_l, which are more stable even in the ultra-low bit situation since the full-precision network is not challenged by the quantization error issue.

In addition, we observe that the initial β_u and β_l are either too small or too large in the early training phase due to the random initialization of gate weights, which obstacles the learning of α_u and α_l, and further leads to inferior performance. Therefore, in the first K training epochs, we do not apply β_u and β_l to scaling α_u, α_l, and the gate is trained individually to push its outputs close to 1 whatever the inputs. After K epochs, we then apply β_u and β_l which start from a stable initial value, *i.e.*, around 1. Then, the dynamic gate controller and the clipping variables are jointly trained to accommodate the drastically varying activation ranges. In all our experiments, we set $K = 5$.

Weight Quantization. The symmetric quantizer is also adopted to quantize weights in previous works. We suggest utilizing the asymmetric quantizer to quantize the weights. Similar to our activation quantization, an upper bound w_u and a lower bound w_l are used to clip the weight range: $\tilde{\boldsymbol{w}} = min(max(\boldsymbol{w}, w_l), w_u)$. The quantized integer and de-quantized value can be obtained similar to Eq. (2) and Eq. (3). This quantization scheme is also compatible with the symmetric case when $w_u = -w_l$. In all our experiments, w_u is set to the 99-th percentile and w_u is set to 1-th percentile of the full-precision weights since we observe no performance gains if setting them to trainable parameters.

非对称. 不列练

3.4   Training Loss

Following PAMS [23], we use L_1 loss and L_{SKT} loss in all the experiments. Denoting $D = \{I_{LR}^i, I_{HR}^i\}_{i=1}^N$ as the training dataset with N pairs of LR and HR images. The L_1 loss and L_{SKT} loss are defined as:

$$L_1 = \frac{1}{N} \sum_{i=1}^N \|I_{LR}^i - I_{HR}^i\|_1, \tag{6}$$

$$L_{SKT} = \frac{1}{N} \sum_{i=1}^N \|\frac{F_S'(I_{LR}^i)}{\|F_S'(I_{LR}^i)\|_2} - \frac{F_T'(I_{LR}^i)}{\|F_T'(I_{LR}^i)\|_2}\|_2, \tag{7}$$

where $F_S'(I_{LR}^i)$ and $F_T'(I_{LR}^i)$ are structure features of I_{LR}^i from the quantized network and the full-precision network, respectively. The structure feature can be calculated by $F'(I_{LR}^i) = \sum_{c=1}^N |F_c(I_{LR}^i)|^2 \in \mathbb{R}^{H \times W}$, where $F(I_{LR}^i) \in \mathbb{R}^{C \times H \times W}$ is the feature map after the last layer in the high-level feature extractor. Then, the overall loss function L is:

$$L = L_1 + \lambda L_{SKT}. \tag{8}$$

where $\lambda = 1,000$ in all our experiments.

Table 1. PSNR/SSIM comparisons between existing low-bit SR methods and our DDTB in quantizing EDSR [24] of scale 4 and scale 2 to the low-bit format. Results of the full-precision model are displayed below the dataset name.

Model	Dataset	Bit	Dorefa [47]	TF Lite [14]	PACT [3]	PAMS [23]	**DDTB(Ours)**
EDSR ×4	Set5 [2] 32.10/0.894	2	29.90/0.850	29.96/0.851	30.03/0.854	29.51/0.835	**30.97/0.876**
		3	30.76/0.870	31.05/0.877	30.98/0.876	27.25/0.780	**31.52/0.883**
		4	30.91/0.873	31.54/0.884	31.32/0.882	31.59/0.885	**31.85/0.889**
	Set14 [22] 28.58/0.781	2	27.08/0.744	27.12/0.745	27.21/0.747	26.79/0.734	**27.87/0.764**
		3	27.66/0.759	27.92/0.765	27.87/0.764	25.24/0.673	**28.18/0.771**
		4	27.78/0.762	28.20/0.772	28.07/0.769	28.20/0.773	**28.39/0.777**
	BSD100 [30] 27.56/0.736	2	26.66/0.704	26.68/0.705	26.73/0.706	26.45/0.696	**27.09/0.719**
		3	26.97/0.716	27.12/0.721	27.09/0.719	25.38/0.644	**27.30/0.727**
		4	27.04/0.719	27.31/0.727	27.21/0.724	27.32/0.728	**27.44/0.732**
	Urban100 [12] 26.04/0.785	2	24.02/0.705	24.03/0.705	24.12/0.708	23.72/0.688	**24.82/0.742**
		3	24.59/0.732	24.85/0.743	24.82/0.741	22.76/0.641	**25.33/0.761**
		4	24.73/0.739	25.28/0.760	25.05/0.751	25.32/0.762	**25.69/0.774**
EDSR ×2	Set5 [2] 37.93/0.960	2	36.12/0.952	36.23/0.952	36.58/0.955	35.30/0.946	**37.25/0.958**
		3	37.13/0.957	37.33/0.957	37.36/0.958	36.76/0.955	**37.51/0.958**
		4	37.22/0.958	37.64/0.959	37.57/0.958	37.67/0.958	**37.72/0.959**
	Set14 [22] 33.46/0.916	2	32.09/0.904	32.14/0.904	32.38/0.907	31.63/0.899	**32.87/0.911**
		3	32.73/0.910	32.98/0.912	32.99/0.912	32.50/0.907	**33.17/0.914**
		4	32.82/0.911	33.24/0.914	33.20/0.914	33.20/0.915	**33.35/0.916**
	BSD100 [30] 32.10/0.899	2	31.03/0.884	31.07/0.885	31.26/0.887	30.66/0.879	**31.67/0.893**
		3	31.57/0.892	31.76/0.894	31.77/0.894	31.38/0.889	**31.89/0.896**
		4	31.63/0.893	31.94/0.896	31.93/0.897	31.94/0.897	**32.01/0.898**
	Urban100 [12] 31.71/0.925	2	28.71/0.886	28.77/0.886	29.22/0.894	28.11/0.875	**30.34/0.910**
		3	30.00/0.906	30.48/0.912	30.57/0.912	29.50/0.898	**31.01/0.919**
		4	30.17/0.908	31.11/0.919	31.09/0.919	31.10/0.919	**31.39/0.922**

4   Experiments

4.1   Implementation Details

All the models are trained on the training set of DIV2K including 800 images [37], and tested on four standard benchmarks including Set5 [2], Set14 [22], BSD100 [30] and Urban100 [12]. Two upscaling factors of ×2 and ×4 are evaluated. The quantized SR models include EDSR [24], RDN [46], and SRRes-Net [22]. We quantize them to 4, 3, and 2-bit and compare with the SOTA competitors including DoReFa [47], Tensorflow Lite (TF Lite) [14], PACT [3], and PAMS [23]. The PSNR and SSIM [41] over the Y channel are reported.

The full-precision models and compared quantization methods are implemented based on their open-source code. For the quantized models, following PAMS [23], we quantize both weights and activations of the high-level feature extraction module. The low-level feature extraction and reconstruction modules are set to the full-precision[1]. The batch size is set to 16 and the optimizer is Adam [19] with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. We set the initial learning rate to 10^{-4} and halve it every 10 epochs. For EDSR, the gate ratio P is set to 30 and the initialization coefficient M is 99. As for RDN, P and M are 50 and 95. For SRResNet, $P = 10$ and $M = 99$. The total training epochs are set to 60. The training images are pre-processed by subtracting the mean RGB. During training, random horizontal flip and vertical rotation are adopted to augment data. All experiments are implemented with PyTorch [33].

---

[1] Results of fully quantized SR models are provided in the Appendix.

Table 2. PSNR/SSIM comparisons between existing low-bit SR methods and our DDTB in quantizing RDN [46] of scale 4 and scale 2 to the low-bit format. Results of the full-precision model are displayed below the dataset name.

Model	Dataset	Bit	Dorefa [47]	TF Lite [14]	PACT [3]	PAMS [23]	**DDTB(Ours)**
RDN ×4	Set5 [2] 32.24/0.896	2	29.90/0.849	29.93/0.850	28.78/0.820	29.73/0.843	**30.57/0.867**
		3	31.24/0.881	30.13/0.854	31.30/0.879	29.54/0.838	**31.49/0.883**
		4	31.51/0.885	31.08/0.874	31.93/0.890	30.44/0.862	**31.97/0.891**
	Set14 [22] 28.67/0.784	2	27.08/0.743	27.11/0.744	26.33/0.717	26.96/0.739	**27.56/0.757**
		3	28.02/0.769	27.21/0.744	28.06/0.767	26.82/0.734	**28.17/0.772**
		4	28.21/0.773	27.98/0.764	28.44/0.778	27.54/0.753	**28.49/0.780**
	BDS100 [30] 27.63/0.738	2	26.65/0.703	26.64/0.703	26.16/0.681	26.57/0.700	**26.91/0.714**
		3	27.20/0.724	26.71/0.705	27.21/0.722	26.47/0.696	**27.30/0.728**
		4	27.30/0.727	27.16/0.720	27.46/0.732	26.87/0.710	**27.49/0.735**
	Urban100 [12] 26.29/0.792	2	23.99/0.702	23.99/0.703	23.38/0.672	23.87/0.696	**24.50/0.728**
		3	25.07/0.754	24.27/0.713	25.17/0.754	23.83/0.692	**25.35/0.764**
		4	25.36/0.764	25.25/0.755	25.83/0.779	24.52/0.726	**25.90/0.783**
RDN ×2	Set5 [2] 38.05/0.961	2	36.20/0.952	36.12/0.951	36.55/0.954	35.45/0.946	**36.76/0.955**
		3	37.44/0.958	36.38/0.953	37.39/0.958	35.25/0.942	**37.61/0.959**
		4	37.61/0.959	36.83/0.955	37.82/0.959	36.53/0.953	**37.88/0.960**
	Set14 [22] 33.59/0.917	2	32.14/0.904	32.08/0.903	32.34/0.905	31.67/0.899	**32.54/0.908**
		3	33.08/0.914	32.46/0.907	33.08/0.914	31.52/0.893	**33.26/0.915**
		4	33.23/0.915	32.72/0.910	33.47/0.916	32.39/0.905	**33.51/0.917**
	BSD100 [30] 32.20/0.900	2	31.06/0.885	31.01/0.884	31.21/0.886	30.69/0.879	**31.44/0.890**
		3	31.87/0.896	31.34/0.888	31.86/0.896	30.62/0.874	**31.91/0.897**
		4	31.98/0.897	31.63/0.892	32.09/0.898	31.27/0.885	**32.12/0.899**
	Urban100 [12] 32.13/0.927	2	28.81/0.888	28.72/0.885	29.15/0.892	28.14/0.874	**29.77/0.903**
		3	30.96/0.918	29.83/0.903	30.97/0.918	28.30/0.873	**31.10/0.920**
		4	31.33/0.922	30.49/0.912	31.69/0.925	29.70/0.898	**31.76/0.926**

4.2 Experimental Results

Table 1, Table 2, and Table 3 respectively show the quantitative results of EDSR, RDN, and SRResNet on different datasets. As can be seen, our DDTB consistently outperforms all the compared methods on these quantized SR models with different bit-widths. More details are discussed below.

Evaluation on EDSR. In the case of 4-bit, our DDTB outperforms the advanced PAMS by a large margin. For instance, for 4-bit EDSR×4, DDTB obtains 0.37dB PSNR gains on Urban100. More noticeable improvements can be observed when performing ultra-low bit quantization. For instance, our DDTB obtains performance gains by 0.94dB, 0.66dB, 0.36dB, and 0.70dB on Set5, Set14, BSD100, and Urban100 when quantizing EDSR×4 to 2-bit.

Evaluation on RDN. When quantizing the model to 4-bit, our DDTB slightly outperforms the existing SOTA of PACT. When it comes to ultra-low bit, the superior performance is in particular obvious. In detail, for 2-bit RDN×4, the performance gains of our DDTB are 0.64dB, 0.45dB, 0.26dB, and 0.51dB on Set5, Set14, BSD100, and Urban100.

Evaluation on SRResNet. The results of SRResNet also manifest that the performance gains of our DDTB are more prominent with ultra-low precision. For 2-bit SRResNet×4, our DDTB improves the performance by 0.65dB, 0.47dB, 0.30dB, and 0.69dB on Set5, Set14, BSD100, and Urban100, while the performance gains are 1.15dB, 0.79dB, 0.67dB, and 1.80dB for 2-bit SRResNet×2.

Analysis. These results demonstrate in Table 1, Table 2, and Table 3 well demonstrate the effectiveness of our DDTB and also verify the correctness of

Table 3. PSNR/SSIM comparison between existing low-bit SR methods and our DDTB in quantizing SRResNet [22] of scale 4 and scale 2 to the low-bit format. Results of the full-precision model are displayed below the dataset name.

Model	Dataset	Bit	Dorefa [47]	TF Lite [14]	PACT [3]	PAMS[23]	**DDTB(Ours)**
SRResNet ×4	Set5 [2] 32.07/0.893	2	30.25/0.860	30.33/0.861	30.86/0.874	30.25/0.861	**31.51/0.887**
		3	30.34/0.862	31.58/0.886	31.62/0.887	31.68/0.888	**31.85/0.890**
		4	30.32/0.861	31.82/0.890	31.85/0.890	31.88/0.891	**31.97/0.892**
	Set14 [22] 28.50/0.780	2	27.33/0.749	27.39/0.751	27.76/0.761	27.36/0.750	**28.23/0.773**
		3	27.39/0.751	28.24/0.773	28.25/0.773	28.27/0.774	**28.39/0.776**
		4	27.41/0.751	28.40/0.777	28.38/0.775	28.41/0.777	**28.46/0.778**
	BSD100 [30] 27.52/0.735	2	26.78/0.708	26.80/0.709	27.03/0.717	26.79/0.709	**27.33/0.728**
		3	26.81/0.709	27.31/0.726	27.33/0.727	27.34/0.728	**27.44/0.731**
		4	26.82/0.709	27.42/0.730	27.41/0.730	27.45/0.732	**27.48/0.733**
	Urban100 [12] 25.86/0.779	2	24.17/0.711	24.21/0.713	24.68/0.733	24.19/0.713	**25.37/0.762**
		3	24.24/0.714	25.33/0.759	25.39/0.761	25.46/0.765	**25.64/0.770**
		4	24.26/0.714	25.62/0.780	25.61/0.769	25.68/0.773	**25.77/0.776**
SRResNet ×2	Set5 [2] 37.89/0.960	2	35.27/0.946	35.34/0.946	36.31/0.953	34.75/0.942	**37.46/0.958**
		3	35.30/0.946	37.50/0.958	37.42/0.958	37.52/0.958	**37.67/0.959**
		4	35.39/0.946	37.69/0.959	37.65/0.959	37.71/0.959	**37.78/0.960**
	Set14 [22] 33.40/0.916	2	31.54/0.899	31.61/0.899	32.23/0.905	31.31/0.896	**33.02/0.913**
		3	31.56/0.899	33.05/0.913	32.92/0.911	33.09/0.914	**33.24/0.915**
		4	31.63/0.899	33.26/0.915	33.24/0.915	33.26/0.915	**33.32/0.916**
	BSD100 [30] 32.08/0.898	2	30.61/0.879	30.66/0.879	31.11/0.885	30.48/0.877	**31.78/0.895**
		3	30.62/0.879	31.81/0.894	31.70/0.893	31.85/0.896	**31.95/0.897**
		4	30.67/0.880	31.99/0.897	31.96/0.897	31.99/0.897	**32.03/0.898**
	Urban100 [12] 31.60/0.923	2	27.98/0.871	28.04/0.872	28.77/0.885	27.86/0.868	**30.57/0.913**
		3	27.99/0.871	30.64/0.913	30.43/0.910	30.69/0.914	**31.15/0.919**
		4	28.06/0.872	31.25/0.920	31.19/0.919	31.20/0.920	**31.40/0.921**

our motivation in designing an appropriate quantizer adaptive to the activation distribution. Moreover, it is worth noticing that DDTB provides more stable improvements over different SR models and bit-widths. Taking EDSR×4 as an example, though PAMS obtains the best results among the compared methods when performing 4-bit quantization, it was outperformed by TF Lite in 3-bit and PACT in 2-bit. In contrast, DDTB achieves the best results in all these three bit-widths. Such stability further illustrates the advanced generalization of our DDTB.

Qualitative Visualizations. Fig. 4 exhibits the qualitative visualizations of 2-bit EDSR[2]. Compared with others methods, the reconstructed HR image of our DDTB provides sharper edges and richer details.

4.3 Model Analysis

Model Complexity. To measure the complexity of the quantized network, we use the parameters and Bit-Operations (BOPs) [39] as the metric. BOPs are the number of multiplication operations multiplied by the bit-widths of two operands. As shown in Table 4, our DDTB significantly reduces the model size and computation cost. When being quantized to 2-bit, the size and computation costs of RDN are reduced by 92% and 94%, respectively. Moreover, the extra computation cost from our dynamic gate controller is negligible, as well as the gate size. For EDSR and SRResNet, the gate occupies less than 1% in model size,

---

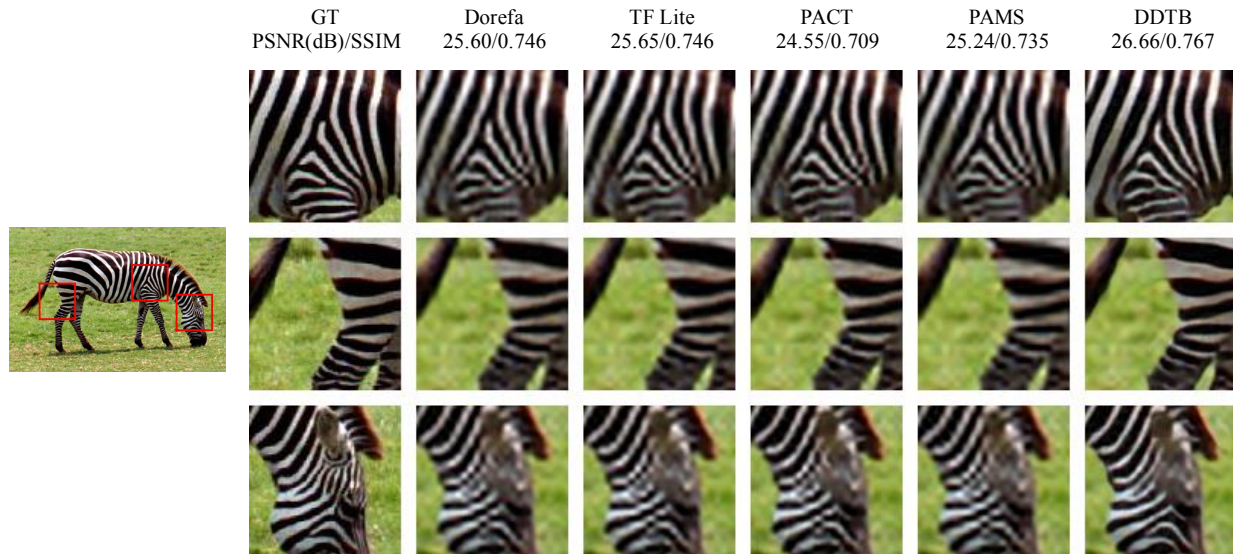[2] More qualitative visualizations are presented in the Appendix.

Fig. 4. Reconstructed results of 2-bit EDSR×4

Table 4. Complexity analysis. The number in brackets indicates the parameters in the high-level feature extraction module. We compute BOPs by generating a 1920×1080 image (upscaling factor ×4). Results of 2-bit are displayed and more can be found in the Appendix.

Model	Bit	Params	Gate Params(*ratio*)	BOPs	Gate BOPs(*ratio*)
EDSR [24]	32	1.52M	0	532T	0
EDSR_DDTB	2	0.41M(0.08M)	0.6%	219T	0.0000013%
RDN [46]	32	22.3M	0	6038T	0
RDN_DDTB	2	1.76M(1.42M)	2.8%	239T	0.0000066%
SRResNet [22]	32	1.543M	0	591T	0
SRResNet_DDTB	2	0.44M(0.07M)	0.1%	278T	0.0000002%

while for RDN, the gate only occupies 2.8%. Note that, the full-precision low-level feature extraction and reconstruction modules occupy most of the memory and computation costs when quantizing the network to the case of ultra-low bit.

Gate Effect. Fig. 5(a) displays the β_u values of 15 randomly selected test images. We select the results of 2-bit EDSR×4. It can be seen that the β_u of different images varies a lot, which proves that the dynamic gate controller can well adjust the clipping bounds adaptive to the input images.

4.4   Ablation Study

This section conducts the ablation study of our DDTB. All experiments are conducted with 2-bit EDSR×4.

Gate Parameters. Fig. 5(b) exhibits the results of different gate ratio P on Set14 [22] dataset. The best result is observed when the P is set to 30. Compared with the case without our gate controller, its PSNR increases by 0.16dB. Using more layers cannot bring improvements and even does slight damage to the performance. Moreover, using the full-precision gate does not provide better results, which indicates that a 2-bit gate is sufficient. To reduce search overhead, we find the best P on the model of scale 4 and apply it to the corresponding model of scale 2. Though not optimal, it is sufficient to show SOTA performance.
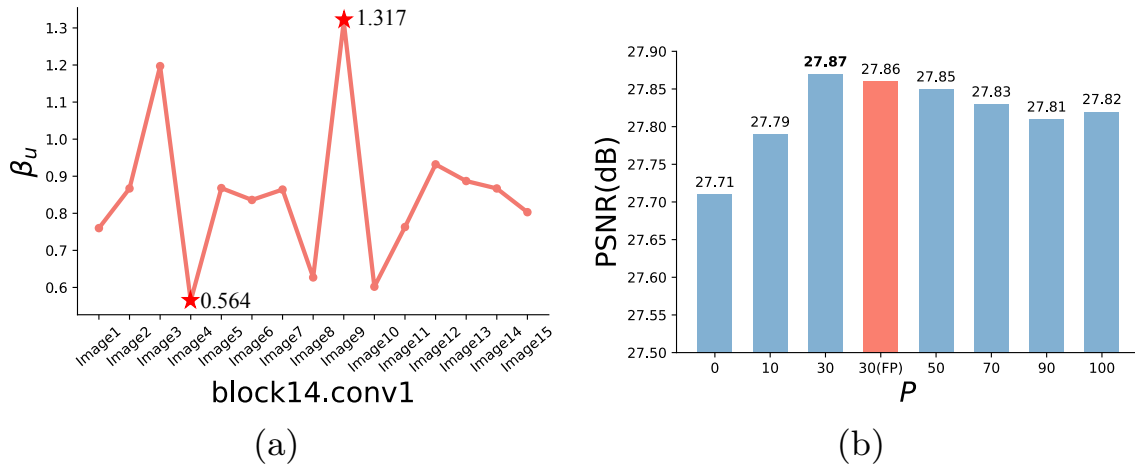
Fig. 5. (a) The β_u in 2-bit EDSR$\times$4. (b) Influence of the gate ratio P in 2-bit EDSR$\times$4.

Table 5. Effect of different components in our methods. "DW": dual bounds for weight quantization. "DI": DDTB initializer. The PSNR/SSIM are reported.

Components			Results			
DDTB	DW	DI	Set5 [2]	Set14 [22]	BSD100 [30]	Urban100 [12]
PAMS [23]			29.51/0.835	26.79/0.734	26.45/0.696	23.72/0.688
✓			30.00/0.853	27.15/0.745	26.69/0.705	24.06/0.707
	✓		29.82/0.846	27.02/0.741	26.62/0.702	23.98/0.702
	✓	✓	30.19/0.852	27.30/0.747	26.76/0.703	24.19/0.711
✓		✓	30.23/0.858	27.33/0.750	26.78/0.709	24.23/0.714
✓	✓		30.80/0.871	27.68/0.760	26.99/0.717	24.64/0.735
✓	✓	✓	**30.97/0.876**	**27.87/0.764**	**27.09/0.719**	**24.82/0.742**

Components. We use PAMS as the baseline to show the effect of different components including dynamic dual training bounds (DDTB) for activation quantization, dual bounds for weight quantization, and DDTB initializer (DI). Table 5 shows the experimental results. When DDTB and DW are individually added, the performance increases compared with the baseline. The DDTB significantly boosts the baseline which proves its ability to fit the asymmetric activation distribution. By combining the initializer, the performance continues to increase. When all of them are applied, the best performance can be obtained.

5   Conclusion

This paper presents a novel quantization method, termed Dynamic Dual Trainable Bounds (DDTB) to solve the asymmetric activation distribution in the DCNN-based SR model. Our DDTB introduces trainable upper and lower bounds, to which a dynamic gate controller is applied in order to adapt to the input sample at runtime. The gate is represented in a 2-bit format and only applied to part of the network to minimize the extra overhead. Moreover, we design a special DDTB initializer for stable training. Our DDTB shows its superiority over many competitors with different quantized SR models on many benchmarks, especially when performing ultra-low precision quantization.

Acknowledgement

References

1. Ahn, N., Kang, B., Sohn, K.A.: Fast, accurate, and lightweight super-resolution with cascading residual network. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 252–268 (2018)
2. Bevilacqua, M., Roumy, A., Guillemot, C., Morel, M.L.A.: Low-complexity single-image super-resolution based on nonnegative neighbor embedding. In: British Machine Vision Conference (BMVC) (2012)
3. Choi, J., Wang, Z., Venkataramani, S., Chuang, P.I.J., Srinivasan, V., Gopalakrishnan, K.: Pact: Parameterized clipping activation for quantized neural networks. arXiv preprint arXiv:1805.06085 (2018)
4. Courbariaux, M., Hubara, I., Soudry, D., El-Yaniv, R., Bengio, Y.: Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. arXiv preprint arXiv:1602.02830 (2016)
5. Dong, C., Loy, C.C., He, K., Tang, X.: Learning a deep convolutional network for image super-resolution. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 184–199 (2014)
6. Dong, C., Loy, C.C., Tang, X.: Accelerating the super-resolution convolutional neural network. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 391–407 (2016)
7. Esser, S.K., McKinstry, J.L., Bablani, D., Appuswamy, R., Modha, D.S.: Learned step size quantization. In: Proceedings of the International Conference on Learning Representations (ICLR) (2020)
8. Han, S., Pool, J., Tran, J., Dally, W.J., et al.: Learning both weights and connections for efficient neural network. In: Proceedings of the Advances in Neural Information Processing Systems (NeurIPS). pp. 1135–1143 (2015)
9. Haris, M., Shakhnarovich, G., Ukita, N.: Deep back-projection networks for super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1664–1673 (2018)
10. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
11. Hong, C., Kim, H., Baik, S., Oh, J., Lee, K.M.: Daq: Channel-wise distribution-aware quantization for deep image super-resolution networks. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 2675–2684 (2022)
12. Huang, J.B., Singh, A., Ahuja, N.: Single image super-resolution from transformed self-exemplars. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5197–5206 (2015)

13. Hui, Z., Gao, X., Yang, Y., Wang, X.: Lightweight image super-resolution with information multi-distillation network. In: Proceedings of the 27th acm international conference on multimedia (ACMMM). pp. 2024–2032 (2019)
14. Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., Kalenichenko, D.: Quantization and training of neural networks for efficient integer-arithmetic-only inference. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2704–2713 (2018)
15. Jiang, X., Wang, N., Xin, J., Li, K., Yang, X., Gao, X.: Training binary neural network without batch normalization for image super-resolution. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI). pp. 1700–1707 (2021)
16. Jung, S., Son, C., Lee, S., Son, J., Han, J.J., Kwak, Y., Hwang, S.J., Choi, C.: Learning to quantize deep networks by optimizing quantization intervals with task loss. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4350–4359 (2019)
17. Kim, J., Lee, J.K., Lee, K.M.: Accurate image super-resolution using very deep convolutional networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1646–1654 (2016)
18. Kim, J., Lee, J.K., Lee, K.M.: Deeply-recursive convolutional network for image super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1637–1645 (2016)
19. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: Proceedings of the International Conference on Learning Representations (ICLR) (2014)
20. Krishnamoorthi, R.: Quantizing deep convolutional networks for efficient inference: A whitepaper. arXiv preprint arXiv:1806.08342 (2018)
21. Lai, W.S., Huang, J.B., Ahuja, N., Yang, M.H.: Deep laplacian pyramid networks for fast and accurate super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 624–632 (2017)
22. Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., et al.: Photo-realistic single image super-resolution using a generative adversarial network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4681–4690 (2017)
23. Li, H., Yan, C., Lin, S., Zheng, X., Zhang, B., Yang, F., Ji, R.: Pams: Quantized super-resolution via parameterized max scale. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 564–580 (2020)
24. Lim, B., Son, S., Kim, H., Nah, S., Mu Lee, K.: Enhanced deep residual networks for single image super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 136–144 (2017)
25. Lin, M., Ji, R., Wang, Y., Zhang, Y., Zhang, B., Tian, Y., Shao, L.: Hrank: Filter pruning using high-rank feature map. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1529–1538 (2020)
26. Lin, M., Ji, R., Xu, Z., Zhang, B., Wang, Y., Wu, Y., Huang, F., Lin, C.W.: Rotated binary neural network. In: Proceedings of the Advances in Neural Information Processing Systems (NeurIPS). pp. 7474–7485 (2020)
27. Luo, X., Xie, Y., Zhang, Y., Qu, Y., Li, C., Fu, Y.: Latticenet: Towards lightweight image super-resolution with lattice block. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 272–289 (2020)
28. Ma, Y., Xiong, H., Hu, Z., Ma, L.: Efficient super resolution using binarized neural network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 694–703 (2019)

29. Magid, S.A., Zhang, Y., Wei, D., Jang, W.D., Lin, Z., Fu, Y., Pfister, H.: Dynamic high-pass filtering and multi-spectral attention for image super-resolution. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 4288–4297 (2021)
30. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 416–423 (2001)
31. Mei, Y., Fan, Y., Zhou, Y.: Image super-resolution with non-local sparse attention. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3517–3526 (2021)
32. Mei, Y., Fan, Y., Zhou, Y., Huang, L., Huang, T.S., Shi, H.: Image super-resolution with cross-scale non-local attention and exhaustive self-exemplars mining. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5690–5699 (2020)
33. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. In: Proceedings of the Advances in Neural Information Processing Systems (NeurIPS). pp. 8026–8037 (2019)
34. Rastegari, M., Ordonez, V., Redmon, J., Farhadi, A.: Xnor-net: Imagenet classification using binary convolutional neural networks. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 525–542 (2016)
35. Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A.P., Bishop, R., Rueckert, D., Wang, Z.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1874–1883 (2016)
36. Tai, Y., Yang, J., Liu, X.: Image super-resolution via deep recursive residual network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3147–3155 (2017)
37. Timofte, R., Agustsson, E., Van Gool, L., Yang, M.H., Zhang, L.: Ntire 2017 challenge on single image super-resolution: Methods and results. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). pp. 114–125 (2017)
38. Tong, T., Li, G., Liu, X., Gao, Q.: Image super-resolution using dense skip connections. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4799–4807 (2017)
39. Van Baalen, M., Louizos, C., Nagel, M., Amjad, R.A., Wang, Y., Blankevoort, T., Welling, M.: Bayesian bits: Unifying quantization and pruning. In: Proceedings of the Advances in Neural Information Processing Systems (NeurIPS). pp. 5741–5752 (2020)
40. Wang, H., Chen, P., Zhuang, B., Shen, C.: Fully quantized image super-resolution networks. In: Proceedings of the 29th ACM International Conference on Multimedia (ACMMM). pp. 639–647 (2021)
41. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing (TIP) **13**(4), 600–612 (2004)
42. Xin, J., Wang, N., Jiang, X., Li, J., Huang, H., Gao, X.: Binarized neural network for single image super resolution. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 91–107 (2020)

43. Xu, X., Lu, Q., Yang, L., Hu, S., Chen, D., Hu, Y., Shi, Y.: Quantization of fully convolutional networks for accurate biomedical image segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8300–8308 (2018)
44. Zhang, R., Chung, A.C.: Medq: Lossless ultra-low-bit neural network quantization for medical image segmentation. Medical Image Analysis **73**, 102200 (2021)
45. Zhang, Y., Li, K., Li, K., Wang, L., Zhong, B., Fu, Y.: Image super-resolution using very deep residual channel attention networks. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 286–301 (2018)
46. Zhang, Y., Tian, Y., Kong, Y., Zhong, B., Fu, Y.: Residual dense network for image super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 2472–2481 (2018)
47. Zhou, S., Wu, Y., Ni, Z., Zhou, X., Wen, H., Zou, Y.: Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. arXiv preprint arXiv:1606.06160 (2016)

Appendix

6 Model Complexity

Table 6 provides the complexity analyses of 4, 3, 2-bit SR models. The extra overhead of the dynamic gate controller is negligible.

Table 6. Complexity analysis. The number in brackets indicates the parameters in the high-level feature extraction module. We compute BOPs by generating a 1920×1080 image (upscaling factor ×4).

| Model	Bit	Params	Gate Params($ratio$)	BOPs	Gate BOPs($ratio$)
EDSR	32	1.52M	0	532T	0
EDSR_DDTB	2	0.41M(0.08M)	0.6%	219T	0.0000013%
EDSR_DDTB	3	0.45M(0.11M)	0.6%	220T	0.0000013%
EDSR_DDTB	4	0.49M(0.15M)	0.5%	222T	0.0000013%
RDN	32	22.3M	0	6038T	0
RDN_DDTB	2	1.76M(1.42M)	2.8%	239T	0.0000066%
RDN_DDTB	3	2.44M(2.10M)	2%	267T	0.0000059%
RDN_DDTB	4	3.13M(2.79M)	1.6%	307T	0.0000051%
SRResNet	32	1.543M	0	591T	0
SRResNet_DDTB	2	0.44M(0.07M)	0.1%	278T	0.0000002%
SRResNet_DDTB	3	0.47M(0.11M)	0.1%	280T	0.0000002%
SRResNet_DDTB	4	0.51M(0.15M)	0.1%	282T	0.0000002%

7   Results of Fully Quantized Models

In this section, we provide the comparisons between the existing fully quantized method FQSR [40] and our DDTB. Following FQSR [40], all layers and skip-connections of SR models are quantized. As shown in Table 7, DDTB outperforms FQSR by a large margin when performing 4-bit quantization. For instance, DDTB obtains performance gains by 0.98dB, 0.58dB, 0.37dB, and 0.77dB on Set5, Set14, BSD100, and Urban100, respectively.

Table 7. PSNR/SSIM comparisons between the existing fully quantized method and our DDTB. "SC" indicates the bit-width of skip-connections.

Model	Bit	SC	Methods	Set5	Set14	BSD100	Urban100
EDSR ×4	4	8	FQSR	30.93/0.870	27.82/0.761	27.07/0.715	24.93/0.744
			DDTB(Ours)	**31.91/0.889**	**28.40/0.777**	**27.44/0.732**	**25.70/0.775**
EDSR ×2	4	8	FQSR	37.04/0.951	32.84/0.908	31.67/0.889	30.65/0.911
			DDTB(Ours)	**37.83/0.960**	**33.44/0.916**	**32.07/0.898**	**31.60/0.924**
SRGAN ×4	4	8	FQSR	30.96/0.872	27.85/0.759	27.08/0.713	24.93/0.742
			DDTB(Ours)	**31.46/0.879**	**28.10/0.766**	**27.26/0.721**	**25.33/0.757**
SRGAN ×2	4	8	FQSR	36.69/0.950	32.64/0.906	31.57/0.888	30.37/0.908
			DDTB(Ours)	**36.84/0.950**	**32.79/0.907**	**31.69/0.889**	**30.70/0.910**
SRResNet ×4	4	8	FQSR	31.04/0.874	27.86/0.761	27.09/0.714	24.95/0.744
			DDTB(Ours)	**31.51/0.881**	**28.17/0.767**	**27.31/0.722**	**25.39/0.760**
SRResNet ×2	4	8	FQSR	36.34/0.945	32.40/0.901	31.37/0.882	29.98/0.899
			DDTB(Ours)	**36.85/0.951**	**32.73/0.907**	**31.63/0.890**	**30.63/0.910**

8   Visualization

Fig. 6, Fig. 7, and Fig. 8 exhibit the reconstructed results of the 2-bit EDSR, 2-bit RDN, and 2-bit SRResNet, respectively. The reported PSNR/SSIM are measured by the displayed image.
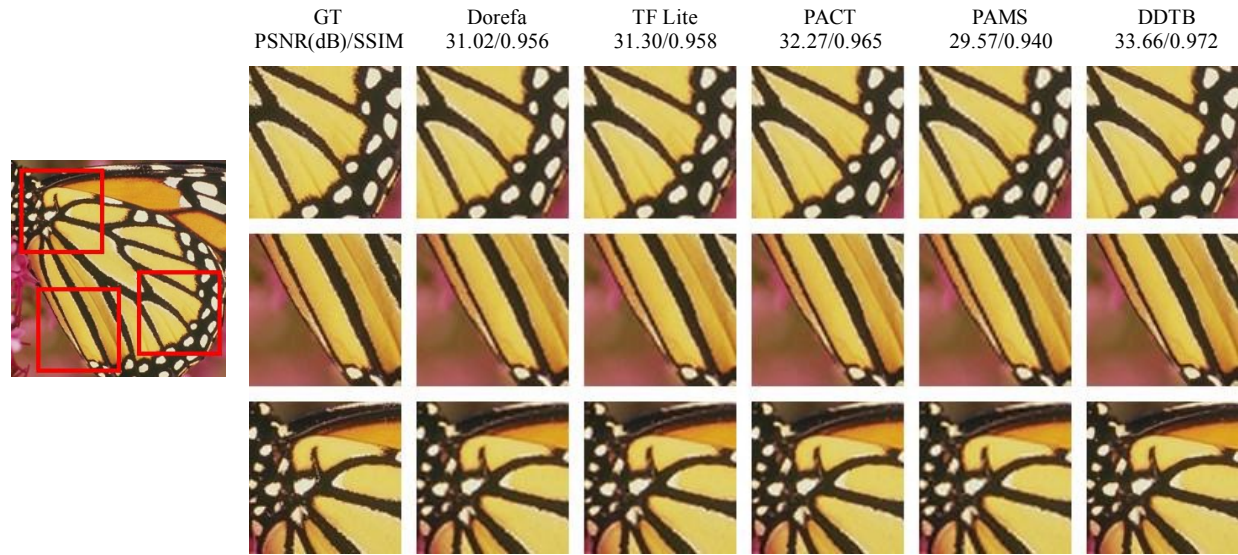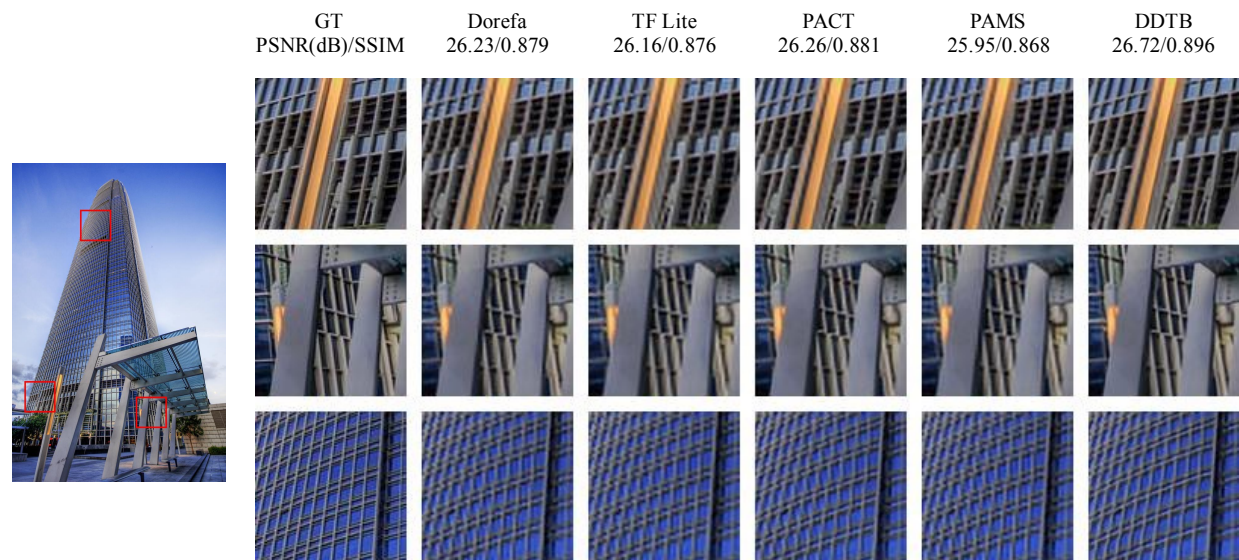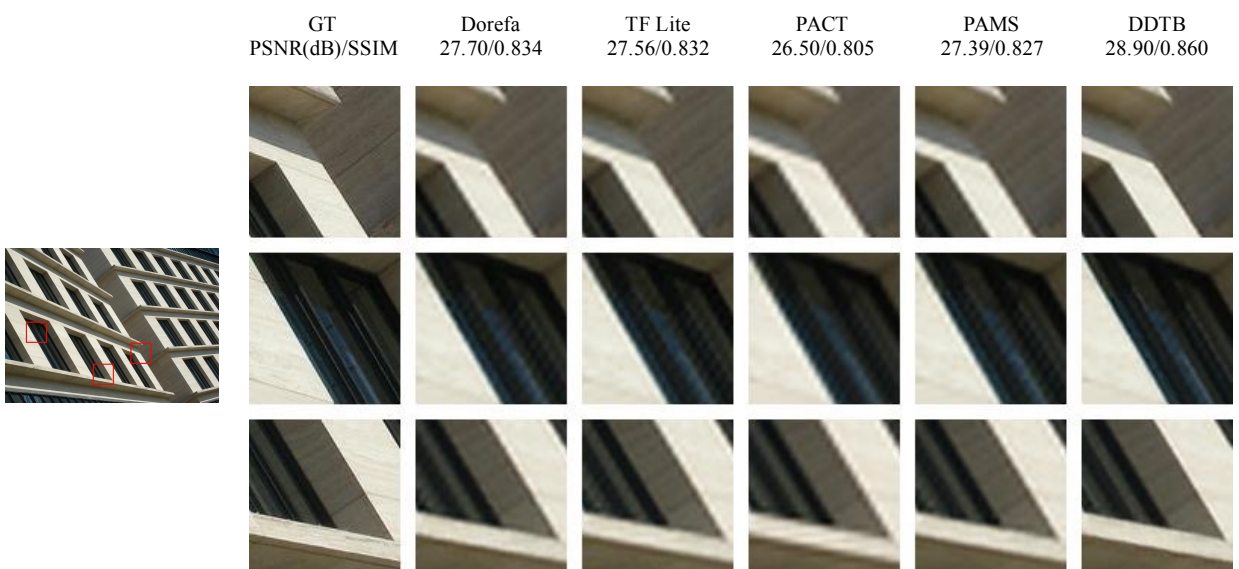

(a) 2-bit EDSR×2.
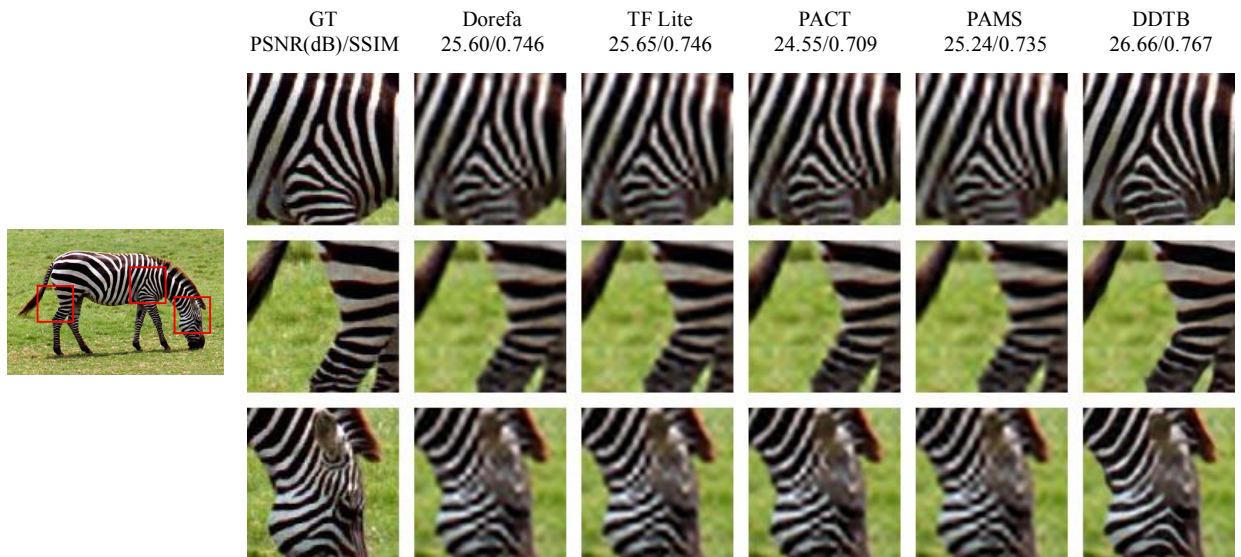
(b) 2-bit EDSR×4.

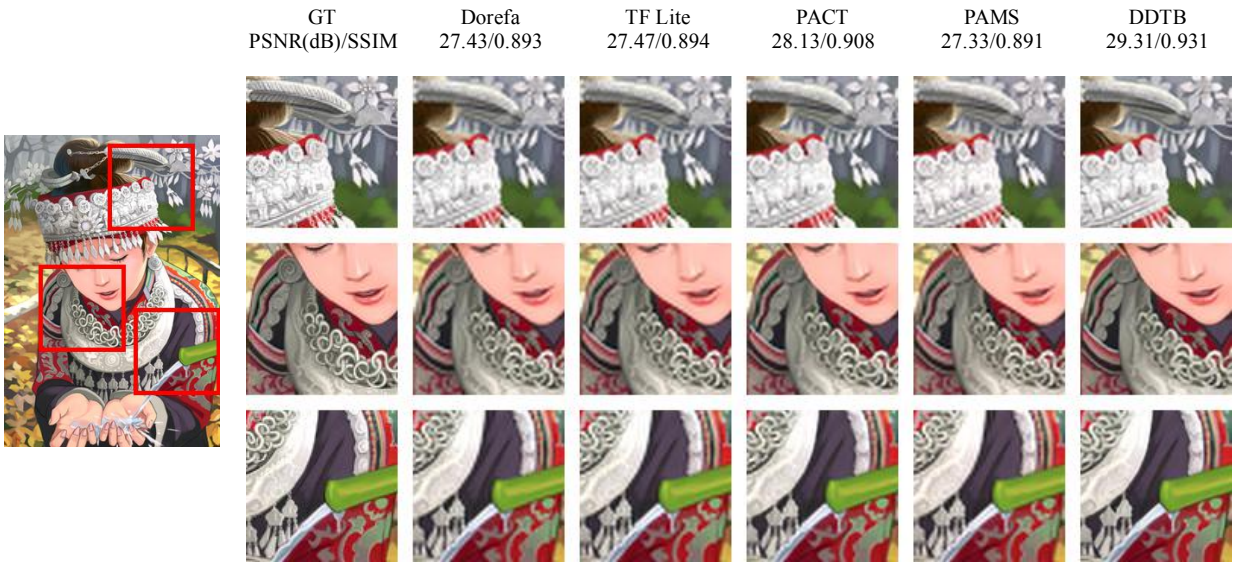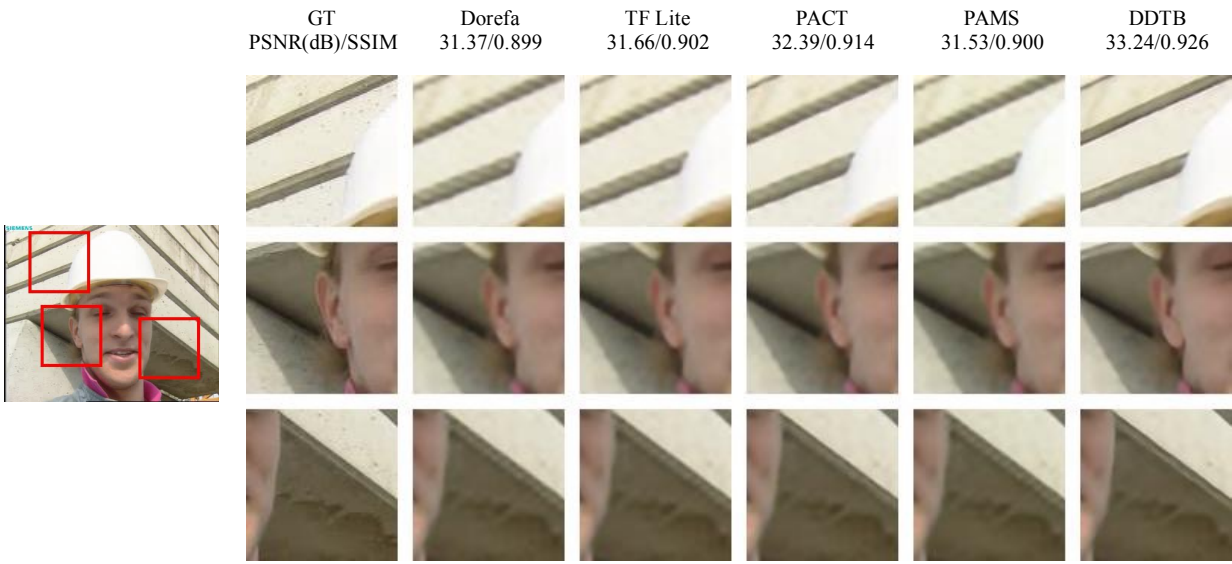**Fig. 6.** Reconstructed results of 2-bit EDSR×2 and EDSR×4.

(a) 2-bit RDN×2.

(b) 2-bit RDN×4.

Fig. 7. Reconstructed results of 2-bit RDN×2 and RDN×4.

| | GT PSNR(dB)/SSIM | Dorefa 27.43/0.893 | TF Lite 27.47/0.894 | PACT 28.13/0.908 | PAMS 27.33/0.891 | DDTB 29.31/0.931 |

(a) 2-bit SRResNet×2.

| | GT PSNR(dB)/SSIM | Dorefa 31.37/0.899 | TF Lite 31.66/0.902 | PACT 32.39/0.914 | PAMS 31.53/0.900 | DDTB 33.24/0.926 |

(b) 2-bit SRResNet×4.

Fig. 8. Reconstructed results of 2-bit SRResNet×2 and SRResNet×4.