SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

March 2025, Vol. 68, Iss. 3, 132108:1–132108:18
https://doi.org/10.1007/s11432-023-4181-0

Distribution-flexible subset quantization for post-quantizing super-resolution networks

Yunshan ZHONG[1,2], Mingbao LIN[4], Jingjing XIE[2,3], Yuxin ZHANG[2,3], Fei CHAO[2,3] & Rongrong JI[1,2,3,5*]

[1]*Institute of Artificial Intelligence, Xiamen University, Xiamen 361005, China*
[2]*Key Laboratory of Multimedia Trusted Perception and Efficient Computing, Ministry of Education of China, Xiamen University, Xiamen 361005, China*
[3]*Department of Artificial Intelligence, School of Informatics, Xiamen University, Xiamen 361005, China*
[4]*Tencent Youtu Lab, Shanghai 200233, China*
[5]*Peng Cheng Laboratory, Shenzhen 518000, China*

Received 22 December 2023/Revised 29 February 2024/Accepted 25 March 2024/Published online 8 February 2025

Abstract This paper introduces distribution-flexible subset quantization (DFSQ), a post-training quantization method for super-resolution networks. Our motivation for developing DFSQ is based on the distinctive activation distributions of current super-resolution models, which exhibit significant variance across samples and channels. To address this issue, DFSQ conducts channel-wise normalization of the activations and applies distribution-flexible subset quantization (SQ), wherein the quantization points are selected from a universal set consisting of multi-word additive log-scale values. To expedite the selection of quantization points in SQ, we propose a fast quantization points selection strategy that uses K-means clustering to select the quantization points closest to the centroids. Compared to the common iterative exhaustive search algorithm, our strategy avoids the enumeration of all possible combinations in the universal set, reducing the time complexity from exponential to linear. Consequently, the constraint of time costs on the size of the universal set is greatly relaxed. Extensive evaluations of various super-resolution models show that DFSQ effectively improves performance even without fine-tuning. For example, for 4-bit EDSR×2 on the Urban benchmark, DFSQ obtains 0.242 dB PSNR gains.

Keywords super-resolution, post-training quantization, distribution-flexible, subset quantization, neural network

Citation Zhong Y S, Lin M B, Xie J J, et al. Distribution-flexible subset quantization for post-quantizing super-resolution networks. Sci China Inf Sci, 2025, 68(3): 132108, https://doi.org/10.1007/s11432-023-4181-0

1 Introduction

Image super-resolution (SR) is a fundamental low-level computer vision task that aims to restore high-resolution (HR) images from low-resolution input images (LR), presenting extensive applications in the area of medical image processing [1–4], video enhancement [5, 6], satellite video processing [7–10], and remote sensing image [11, 12]. Recently, due to the remarkable success of deep neural networks (DNNs), DNNs-based SR models have become a de facto standard for SR task [13–17]. However, the astonished performance of recent SR models typically relies on increasing network size and computational cost, thereby limiting their applications, especially in resource-hungry devices such as smartphones. Therefore, compressing SR models has gained extensive attention from both academia and industries. Various network compressing techniques have been investigated to realize model deployment [18–21].

Among these techniques, network quantization, which maps the full-precision weights and activations within networks to a low-bit format, harvests favorable interest from the community for its ability to reduce storage size and computation cost simultaneously [22–29]. For example, Refs. [25, 26] quantized the SR models using binary quantization, and Refs. [27–29] quantized SR models to low-bit such as 4-bit. Despite notable progress, these methods have to retrain quantized models on the premise of access to the entire training set, known as quantization-aware training (QAT). In real-world scenarios, however, acquiring original training data is sometimes prohibitive due to privacy, data transmission, and security issues. Besides, the heavy training and energy cost also prohibit its practical deployment.

* Corresponding author (email: rrji@xmu.edu.cn)

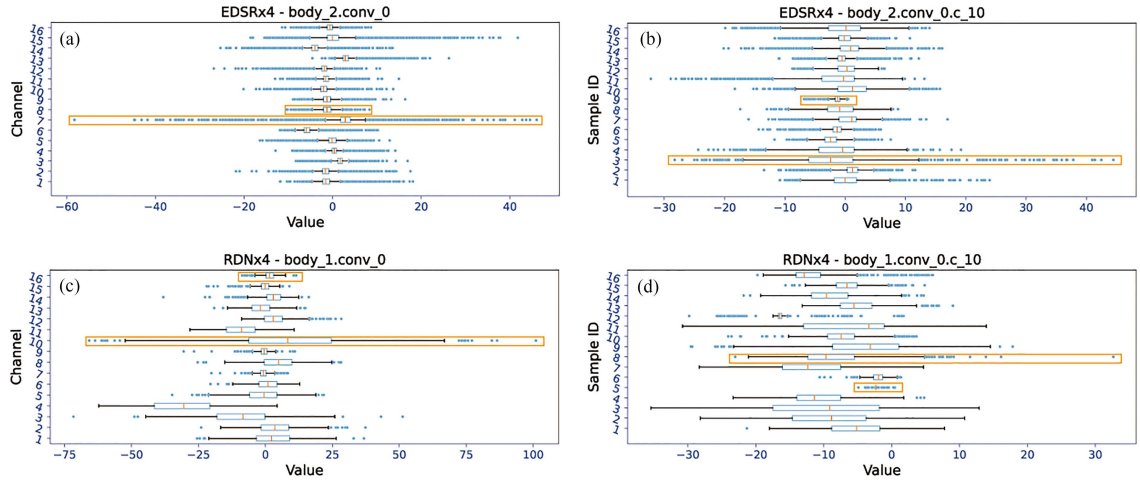

Figure 1 (Color online) Variation of activation distributions of (a) EDSR×4-body_2.conv_0, (b) EDSR×4-body_2.conv_0.c_10 [13], (c) RDN×4-body_1.conv_0, and (d) RDN×4-body_1.conv_0.c_10 [14]. We show activation distributions of different channels given the same input and of different inputs given the same channel. The orange box depicts the data with the maximum discrepancy.

Post-training quantization (PTQ) methods, which perform quantization with only a tiny portion of the original training set, require no or a little retraining, by nature can be a potential way to solve the above problems [30–34]. However, advanced PTQ methods are mainly designed for high-level vision tasks, a direct extension of which to SR models is infeasible since low-level models comprise different structures [27, 29]. Recently, PTQ4SR [35] introduces the first SR post-training method where the clipping values are determined in a two-stage coarse-to-fine manner. Nevertheless, PTQ4SR is an optimization-based method that involves gradient calculation and still suffers from considerable performance degradation.

The main obstacle of quantizing SR models, as a wide consensus, lies in the high activation variations that result from the removal of BN in SR models [13, 14, 17, 36]. As shown in Figure 1(a), the activations present considerable distribution discrepancy among different channels of the same sample and among the same channel of different samples. At first, it can be seen the interquartile range (IQR) of channel_7 and channel_8 differs a lot. The IQR of the former is greater than 0 while the latter is less than 0. Second, the outlier distribution of channel_7 is far wider than that of channel_8, causing the activation range to differ by 5×–6×. Moreover, as illustrated in Figure 1(b), different samples present significant distribution discrepancies in channel_10. In particular, sample_3 manifests 10× more IQR than that of sample_9. Also, the activation of sample_3 ranges from −30–48, while sample_9 only ranges from −8–0. The high variance of activation across channels and samples makes it difficult to solve with current methods. Specifically, current methods [32, 35, 37] do not consider the variance of activations across channels and samples. In addition, the uniform quantization adopted by them is unable to well approximate the non-uniform activations. Thus, they result in limited performance, which is experimentally demonstrated in the experimentation section.

In this paper, we propose a distribution-flexible subset quantization (DFSQ) method[1)] to handle such highly variational activations. Specifically, considering the high variance among samples and channels, we first perform channel-wise normalization and then conduct distribution-flexible and hardware-friendly subset quantization (SQ) [38] to quantize the normalized activations. The normalization comprises two consecutive on-the-fly operations including subtracting the mean and dividing by the maximum absolute value for each channel of each sample. As a result, the range is normalized to −1–1 whatever the input sample and the activation channel, thereby eliminating the variance across samples and channels. Given the non-uniformity of normalized distribution, we suggest adopting the hardware-friendly SQ [38], which aims to find the best quantization points from a universal set that consists of log-scale values. However, the search of quantization points in [38] involves an iterative exhaustive search algorithm that makes the time costs exponential w.r.t. the size of the universal set, resulting in prohibitive time overhead and limiting the size of the universal set. Therefore, we introduce a fast quantization points selection strategy to speed up the selection of quantization points in SQ. In particular, we perform bit-width related K-means clustering at first. Then, from a given universal set, we select the quantization points closest to K

1) The code is available at https://github.com/zysxmu/DFSQ.

centroids. Our strategy circumvents the enumeration of all possible combinations of quantization points in the universal set and thereby reduces the time complexity from exponential to linear. Consequently, the limitation of time costs on the size of the universal set is greatly relaxed.

Extensive evaluations of EDSR (enhanced deep residual network) [13], RDN (residual dense network) [14], and SRResNet (super-resolution residual neural network) [39] on four benchmark datasets demonstrate the effectiveness of the proposed DFSQ. Notably, without any fine-tuning, DFSQ obtains comparable performance to the full-precision counterparts in high-bit cases such as 8- and 6-bit. For low-bit cases such as 4-bit, DFSQ still well retains the performance and improves the state-of-the-art (SOTA) method by a large margin. For example, on Urban100, DFSQ obtains 0.242 dB PSNR (peak signal-to-noise ratio) increase for 4-bit EDSR×2. Our contributions can be summarized as the following.

• We identify that the activations of SR models exhibit high variance across samples and channels, posing a significant challenge for PTQ methods.

• To handle such highly variational activations, we propose a DFSQ method by first performing normalization and applying SQ. We then introduce a fast quantization points selection strategy to speed up the selection of quantization points in SQ.

• The proposed DFSQ exhibits improved performance compared with the SOTA method across all bit-width configurations. Notably, in the low-bit cases, DFSQ significantly surpasses the SOTA method.

2 Related work

Single image super resolution. Along with the huge success of DNNs on many computer vision tasks, DNN-based SR models also obtain great performance increases and have dominated the field of image SR. As a pioneer, Ref. [15] proposed an end-to-end SRCNN to learn the mapping relationship between LR and HR images. VDSR [16] further improves performance by increasing network depth. Afterward, skip-connection based blocks [39, 40] are extensively adopted by the subsequent studies [13, 14] to alleviate the gradient vanishing issue and retain image details. For better performance, researchers introduce many complex structures to construct SR models such as channel attention mechanism [17, 41], non-local attention [42, 43], and transformer-based block [44, 45]. With the increasing demand for the deployment of SR models on resource-limited devices, many studies aim to design lightweight network architectures. DRCN [46] and DRRN [47] both adopt the recursive structure to increase the depth of models while reducing the model size. Some studies design modules to substitute for the expensive up-sampling operation. FSRCNN [48] introduces a de-convolutional layer, and ESPCN [49] instead devises a sub-pixel convolution module. Many other studies utilize the efficient intermediate feature representation [50–54] or network architecture search [55]. For example, Xiao et al. [50] introduced a top-k token selective method named residual token selective group (RTSG) to select the top-k crucial tokens dynamically. They also proposed a multi-scale feed-forward layer (MFL) for better feature aggregation.

Quantized SR models. Network quantization enjoys the merit of both reducing storage size and efficient low-bit operations and thereby harvesting ever-growing attention [22–29]. Ma et al. [26] proposed binary quantization for the weights within SR models. BAM [25] and BTM [24] further binarize the activation of SR models by introducing multiple feature map aggregations and skip connections to reduce the sharp performance drops caused by the binary activation. Other than binary quantization, many studies focus on performing low-bit quantization [22,23,27–29]. Li et al. [27] found unstable activation ranges and proposed a symmetric layer-wise linear quantizer with a learnable clipping value. A knowledge distillation loss is devised to transfer structured knowledge of the full-precision model to the quantized model. Wang et al. [22] designed a fully-quantization method for SR models, in which the weight and activation within all layers are quantized with a symmetric layer-wise quantizer. Zhong et al. [29] introduced two learnable clipping values and a dynamic gate for matching the high variant activation. In [28], a dynamic bit-width adjustment network is introduced for different input patches that have various structure information. Recently, PTQ4SR [35] introduces a post-training method for SR. They first search the clipping values to cut off the outliers and then fine-tune the values with pixel-aware calibration loss. HPTQ (hybrid post-training quantization) [56] proposes a hybrid post-training that consists of piecewise quantization, mix-precision quantization, and clustered quantization.

3 Methodology

QAT usually trains the quantized network for many epochs, by accessing the entire training set, to gradually accommodate the quantization effect [57]. Differently, PTQ is confined to a small portion of the original training set, leading to a severe over-fitting issue [32]. Thus, the key to PTQ has drifted to fitting the data distribution. Below, we first demonstrate the obstacle in performing PTQ for SR models lies in the high variance activation distributions. Then, we introduce the SQ and a corresponding fast selection strategy.

3.1 Observation

It is a wide consensus that the removal of the batch normalization layer in SR models improves the quality of output HR images [13, 14, 17, 36]. Unfortunately, as discussed in many previous studies [23, 27, 29], the removal of the BN layer creates the obstacle for quantization since the resulting activations of high variance make low-bit networks hard to fit. In particular, the high-variance activations are two folds: (1) considerable distribution discrepancy in different channels for a given input sample; (2) considerable distribution discrepancy in different samples for a given channel.

Figure 1 presents the example of activation distributions within EDSR×4 [13]. Specifically, Figure 1(a) presents the activation distributions of different channels given the same sample. It can be seen that the distribution exhibits significant discrepancy. For example, the IQR of channel_7 is greater than 0 while channel_8 is less than 0. Moreover, the range of channel_7 is far wider than that of channel_8, resulting in a range difference by 5×–6×. Figure 1(b) presents the activation distribution of different input samples given the same channel. As it shows, given the same channel_10, extreme discrepancies between the distribution of different samples are revealed. Taking sample_3 and sample_9 as examples, the IQR of the former ranges from −6–1.5, while the latter ranges from −1.64–−0.8, almost 10× difference. Also, their range differs a lot. The activation of sample_3 ranges from −30–48, while sample_9 only ranges from −8–0, still resulting in a 10× difference. Therefore, the high-variance activation distribution is reflected by extreme discrepancy among different channels and different samples.

To handle such activation distributions, previous SR quantization methods rely on QAT to adjust network weights to accommodate the quantization effect [57]. However, PTQ hardly succeeds in this manner since the availability of partial data easily causes over-fitting issue [32]. Thus, for current PTQ methods, this high-variance activation poses a significant challenge. Specifically, despite activations across channels and samples having considerable differences in range, current PTQ methods [32, 35, 37] handle all activations with the same quantization parameters, failing to accommodate their variance. Moreover, the highly nonuniform distribution makes the common linear uniform quantization adopted by current PTQ methods hard to fit the activations [58]. Consequently, these methods yield limited performance, as demonstrated in the experimentation section. Therefore, the core is to find a suitable quantizer that can well fit the activations as much as possible.

3.2 DFSQ

3.2.1 *Quantization process*

We denote a full-precision feature map as $X \in R^{B\times C\times H\times W}$, where B, C, H, W respectively denote mini-batch size, channel number, height, and width of the feature maps. Considering that the activation distributions vary quite a lot across samples and channels, we choose to perform quantization for each channel of each sample, denoted as $X_{i,c} \in R^{H\times W}$, where $X_{i,c}$ denotes the c-th feature map (channel) of the i-th input sample. We first normalize the activation of $X_{i,c}$ by

$$X^n_{i,c} = f_n(X_{i,c}) = \frac{X_{i,c} - \mu_{i,c}}{\mathcal{M}_{i,c} - \mu_{i,c}}, \tag{1}$$

where $\mu_{i,c}$ and $\mathcal{M}_{i,c}$ denote the mean value and maximum absolute value of $X_{i,c}$. The superscript "n" represents the operation of normalization.

After normalization, the activation is scaled to −1–1 whatever the input and channel are, thereby facilitating the following quantization:

$$X^q_{i,c} = Q(X^n_{i,c}), \tag{2}$$

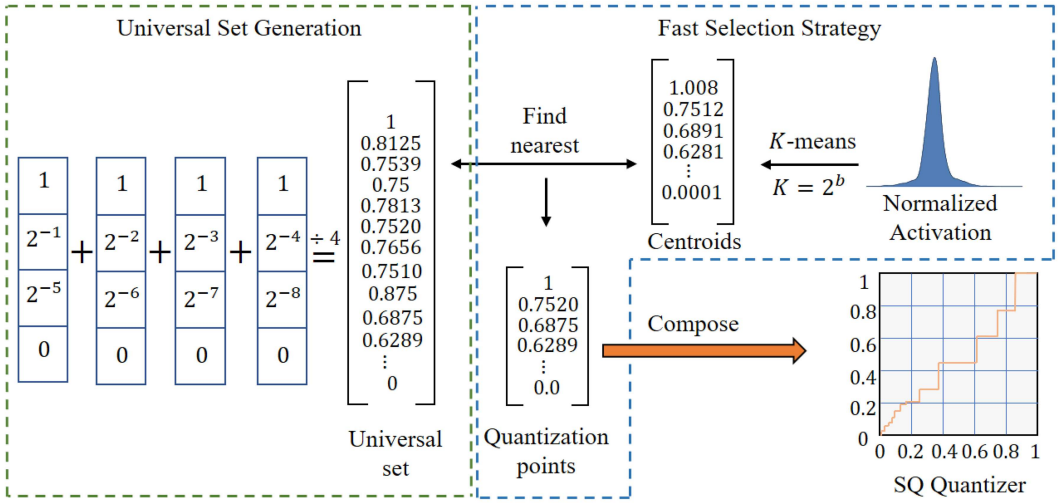

Figure 2 (Color online) Framework of the proposed DFSQ.

where $Q(\cdot)$ denotes the quantizer, which is elaborated in the next subsection. The superscript "q" denotes quantized results. Then, the de-quantized activation can be obtained by conducting de-normalization after getting $X^q_{i,c}$:

$$\bar{X}_{i,c} = f_n^{-1}(X^q_{i,c}) = X^q_{i,c} \cdot \mathcal{M}_{i,c} + \mu_{i,c}. \tag{3}$$

3.2.2 *SQ*

We elaborate on the aforementioned quantizer $Q(\cdot)$ in this subsection. Despite that the normalized activations conform to the same range across different channels and samples, they are still featured with high non-uniformity. Such erratic distributions are hard to be fitted by the common linear uniform quantization [58]. Therefore, we suggest the distribution-flexible and hardware-friendly SQ [38]. Specifically, SQ aims to find the best quantization points from a predefined universal set that usually consists of the additive of multi-word log-scale values [58–60]. Given a universal set Φ_u, bit-width b, and an input value x, the quantizer $Q(\cdot)$ is defined as

$$\begin{aligned} &Q(x) = \arg\min_{p \in \Phi_s} |x - p|, \\ &\text{s.t.} \quad \Phi_s = \{p_i \in \Phi_u | i = 1, \ldots, 2^b\}, \end{aligned} \tag{4}$$

where Φ_s is the set of selected quantization points. Note that the selection of Φ_s only performs once with a small calibration dataset. After selection, Φ_s is fixed and used for quantized inference. The key steps of SQ are the universal set generation and the quantization points selection.

Universal set generation. The universal set Φ_u should contain adequate candidate values to represent any given input distribution [38]. The universal set used in our paper is presented in the left part of Figure 2. Specifically, we adopt four word sets, each of which contains four elements that are either zero or log-scale values. The universal set is obtained by averaging all possible combinations of the elements from each word set. For example, the value 0.8125 in the universal set is obtained by averaging the sum of 1 from the first word set, 2^{-2} from the second word set, and 1s from the third and fourth word sets. As a result, we can obtain a universal set that consists of many non-negative values. For distribution with negative parts, it needs negative values. To do this, we add the negative values by changing the sign of non-negative values. For instance, if the value 0.8125 is present, its corresponding negative value, -0.8125, will also be added to the universal set. After duplicate removal, a total of 107 values are given as the universal set. Note that the division of 4 can be fused into elements of each word set. Therefore, elements of each universal set are either zero or 2^k, and the multiplication between quantized activation and quantized weight is equivalent to conducting the cheap and hardware-friendly shift operation.

Quantization points selection. The common selection strategy of quantization points involves an iterative exhaustive search algorithm, where all possible combinations in the universal set are exhaustively checked to find out the one that minimizes the quantization loss. However, such a strategy incurs intolerable time costs as the size of the universal set increases. In particular, the number of all possible

combinations is $C_n^{2^b} = \frac{n!}{2^b!(n-2^b)!}$, where b is the bit-width and n is the size of the universal set. It can be seen the increase of combinations is an exponential growth w.r.t. the size of the universal set. Given the 4-bit case and the universal set defined above, the total number of combinations is $C_{107}^{2^4} = 4.336 \times 10^{18}$. Clearly, for handling such a large amount, the time costs of the iterative exhaustive search are prohibitive. Therefore, the size of the universal set is limited and a fast selection strategy is necessary.

Fast selection strategy. We then introduce a fast quantization points selection strategy to speed up the selection of quantization points in SQ. We are mainly inspired by the K-means algorithm which can be viewed as a solution for the quantization loss minimization problem with a given distribution and bit-width setting [61]. In particular, given $X_{i,c}^n$, we perform K-means clustering at first by setting $K = 2^b$. To avoid the local optimum, in practice, we perform K-means by 3 times and select the results with the minimum sum of squared errors (SSE)

$$\Phi_\mu = \underset{\Phi_\mu^i \in \{\Phi_\mu^1, \Phi_\mu^2, \Phi_\mu^3\}}{\arg\min} \mathrm{SSE}_{\Phi_\mu^i}, \tag{5}$$

where $\mathrm{SSE}_{\Phi_\mu^i} = \sum_{x \in C_j^i} \|x - \mu_j^i\|^2$, Φ_μ^i denotes the centroids set of the i-th trial of K-means, C_j^i denotes the j-th cluster of the i-th trial, and μ_j^i denotes the centroid of C_j^i. Afterward, from a given universal set, the quantization points set Φ_s is built by selecting these points closest to K centroids of Φ_μ.

The time complexity of K-means is $\mathcal{O}(NTK)$, where $N = H \times W$ is the total number of elements in $X_{i,c}^n$, T is the number of iterations in the clustering process, and $K = 2^b$ [62]. Also, the selection of quantization points closest to K centroids only requires $\mathcal{O}(|\Phi_u|2^b)$. Note that the maximum bit-width b is 8, which gives the $K = 256$ at most. Thus, it is safe to say the time complexity of our fast selection strategy is linear. As a result, the time complexity of our strategy linearly depends on the N, K, T and the size of the universal set. Compared with the previous iterative exhaustive search algorithm, the time complexity is reduced from exponential to linear. The cumbersome drudgery of enumerating all possible combinations is avoided and therefore the limitation of time costs on the size of the universal set is greatly relaxed. For example, DFSQ finishes quantization in 15 min given one NVIDIA 1080Ti for 4-bit EDSR×4. In contrast, the time costs of the iterative exhaustive search algorithm used in [38] is nearly infinite as stated in the previous. Specifically, using the iterative exhaustive search to solve 57915 combinations requires 1 h. Thus, the time cost of solving 4.336×10^{18} combinations requires 3.119×10^{12} days, which is prohibitive.

3.3 Weight quantization

For weight quantization, we adopt kernel-wise linear uniform quantization. Given the weight $W \in R^{C_{\text{out}} \times C_{\text{in}} \times K \times K}$, where $C_{\text{out}}, C_{\text{in}}, K$ denote output channel number, input channel number, and kernel size, respectively. For a kernel W_k, the quantizer is defined as

$$W_k^q = \text{round}\left(\frac{W}{s}\right) + Z, \quad s = \frac{u_w - l_w}{2^b - 1}, \quad Z = \text{round}\left(\frac{-l_w}{s}\right), \tag{6}$$

where b, l_w, u_w, s, Z denote bit-width, weight minimum, weight maximum, step size, and zero-point integer corresponding to the full-precision 0 respectively. The de-quantized value is obtained by

$$\bar{W}_k = s \cdot (W_k^q - Z). \tag{7}$$

4 Experimentation

4.1 Implementation details

The quantized SR models include BN-free EDSR [13], RDN [14], and BN-inserted SRResNet [39]. For each SR model, we evaluate two upscaling factors of ×2 and ×4 and perform 8-, 6-, 4-, and 3-bit quantization, respectively. The calibration dataset contains 32 images random sampled from the training set of DIV2K [63]. The models are tested on four standard benchmarks including Set5 [64], Set14 [39], BSD100 [65] and Urban100 [66]. We report the PSNR and SSIM (structural similarity) [67] over the Y channel as the metrics.

Table 1 Effect of different components in our paper. "Cha.": channel-wise activation quantization. "Norm.": normalization. The results are obtained by quantizing EDSR×4 to 4-bit and the PSNR/SSIM are reported as the metrics. "–" denotes that the model collapses.

Components		Results			
Cha.	Norm.	Set5	Set14	BSD100	Urban100
		–	–	–	–
✓		–	–	–	–
	✓	31.543/0.8790	28.262/0.7675	27.358/0.7247	25.601/0.7586
✓	✓	31.755/0.8855	28.397/0.7749	27.430/0.7307	25.769/0.7736

Table 2 Universal set designing.

Settings	2×4	3×4	4×4	5×4
Word sets	$\{1, 2^{-1}, 2^{-3}, 0\}$, $\{1, 2^{-2}, 2^{-4}, 0\}$.	$\{1, 2^{-1}, 2^{-3}, 0\}$, $\{1, 2^{-2}, 2^{-4}, 0\}$, $\{1, 2^{-3}, 2^{-5}, 0\}$.	$\{1, 2^{-1}, 2^{-5}, 0\}$, $\{1, 2^{-2}, 2^{-6}, 0\}$, $\{1, 2^{-3}, 2^{-7}, 0\}$, $\{1, 2^{-4}, 2^{-8}, 0\}$.	$\{1, 2^{-1}, 2^{-6}, 0\}$, $\{1, 2^{-2}, 2^{-7}, 0\}$, $\{1, 2^{-3}, 2^{-8}, 0\}$, $\{1, 2^{-4}, 2^{-9}, 0\}$, $\{1, 2^{-5}, 2^{-10}, 0\}$.

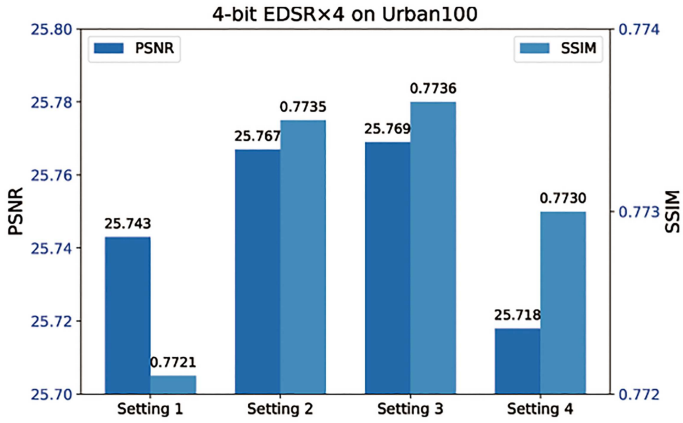

Figure 3 (Color online) Results of 4-bit EDSR×4 on Urban100 with different universal set settings.

The full-precision models and compared methods are implemented based on the official open-source code or the description of the original paper. Following [27,29], we quantize both weights and activations of the high-level feature extraction module of the quantized models. The low-level feature extraction and reconstruction modules remain the full-precision. All experiments are implemented with PyTorch [68]. The max iteration of K-means is set to 100 and the termination threshold is set to 1e−3.

We adopt the Min-Max linear uniform quantization for comparing with optimization-based BRECQ [32], QDROP [37], and the recent PTQ4SR [35]. Given the model difference and code unavailability, we re-implement PTQ4SR based on our full-precision models.

4.2 Ablation study

Table 1 shows the effect of different components. With both channel-wise activation and normalization, our DFSQ presents the best results. As shown in the first and the second rows, the quantized model collapses without normalization. Thus, the normalization that makes the activations range from −1 to 1 is very important for the performance. The third row provides the results without layer-wise activation quantization, which already presents an acceptable performance. Further, channel-wise activation brings significant performance improvements. On Urban100, the quantized model presents a 0.168 dB PSNR increase, demonstrating the efficacy of handling each channel independently.

We further construct different universal sets by fixing the size of each word set and varying the number of word sets. Table 2 provides four settings including 2×4, 3×4, 4×4, and 5×4. The performance comparison is presented in Figure 3. It can be seen that by increasing the number of word sets from 2 to 3 (2×4 vs. 3×4), the PSNR is improved by 0.34 dB. When the number of word sets is increased to 4, the PSNR is improved by 0.002 dB. While at the 5×4 setting, the PSNR drops by 0.051 dB, indicating

Table 3 PSNR/SSIM results of the compared methods and our DFSQ in quantizing EDSR [13] of scales ×2 and ×4. Results of the full-precision model are presented below the dataset name.

Model	Dataset	Bit	BRECQ	QDROP	Min-Max	PTQ4SR	DFSQ (ours)
EDSR ×2	Set5 37.931/0.9604	8	37.921/0.9603	37.926/0.9603	37.926/0.9603	37.687/0.9598	37.928/0.9603
		6	37.865/0.9597	37.882/0.9598	37.905/0.9601	37.670/0.9596	37.927/0.9603
		4	37.499/0.9564	37.370/0.9572	37.497/0.9559	37.652/0.9588	37.832/0.9599
		3	36.845/0.9500	36.603/0.9526	36.199/0.9383	37.126/0.9528	37.382/0.9567
	Set14 33.459/0.9164	8	33.457/0.9163	33.457/0.9163	33.451/0.9164	33.157/0.9152	33.459/0.9164
		6	33.419/0.9157	33.398/0.9159	33.436/0.9161	33.154/0.9149	33.455/0.9164
		4	33.138/0.9123	32.947/0.9123	33.229/0.9122	33.128/0.9142	33.399/0.9159
		3	32.714/0.9054	32.459/0.9074	32.477/0.8955	32.866/0.9073	33.068/0.9113
	BSD100 32.102/0.8987	8	32.098/0.8986	32.099/0.8986	32.100/0.8987	31.919/0.8972	32.101/0.8987
		6	32.066/0.8979	32.060/0.8980	32.089/0.8984	31.894/0.8970	32.099/0.8986
		4	31.829/0.8939	31.710/0.8939	31.911/0.8941	31.874/0.8965	32.060/0.8981
		3	31.475/0.8866	31.324/0.8885	31.263/0.8764	31.632/0.8897	31.824/0.8939
	Urban100 31.709/0.9248	8	31.698/0.9246	31.683/0.9245	31.663/0.9245	30.683/0.9163	31.707/0.9247
		6	31.588/0.9235	31.463/0.9228	31.642/0.9241	30.632/0.9160	31.698/0.9246
		4	30.874/0.9158	30.265/0.9111	31.367/0.9188	30.569/0.9157	31.609/0.9236
		3	30.106/0.9041	29.407/0.8989	30.395/0.8977	30.480/0.9098	30.972/0.9137
EDSR ×4	Set5 32.095/0.8938	8	32.088/0.8935	32.089/0.8936	32.087/0.8936	31.793/0.8901	32.090/0.8937
		6	32.018/0.8909	31.996/0.8911	32.056/0.8925	31.519/0.8843	32.079/0.8933
		4	31.287/0.8722	31.103/0.8715	31.364/0.8687	31.440/0.8808	31.755/0.8855
		3	30.164/0.8342	30.286/0.8478	29.150/0.7580	30.027/0.8083	30.757/0.8489
	Set14 28.576/0.7813	8	28.566/0.7809	28.566/0.7810	28.566/0.7809	28.361/0.7776	28.568/0.7810
		6	28.516/0.7788	28.501/0.7788	28.549/0.7801	28.184/0.7737	28.560/0.7807
		4	28.080/0.7635	27.922/0.7634	28.159/0.7617	28.133/0.7697	28.397/0.7749
		3	27.396/0.7330	27.392/0.7446	26.723/0.6681	27.313/0.7141	27.732/0.7431
	BSD100 27.562/0.7355	8	27.557/0.7352	27.557/0.7352	27.555/0.7351	27.464/0.7337	27.558/0.7354
		6	27.507/0.7326	27.509/0.7330	27.547/0.7344	27.347/0.7311	27.555/0.7351
		4	27.198/0.7184	27.153/0.7197	27.255/0.7168	27.343/0.7266	27.430/0.7307
		3	26.717/0.6903	26.811/0.7032	26.117/0.6253	26.699/0.6767	27.044/0.7074
	Urban100 26.035/0.7848	8	26.018/0.7843	26.002/0.7841	26.014/0.7844	25.626/0.7749	26.025/0.7845
		6	25.907/0.7801	25.849/0.7791	25.997/0.7831	25.419/0.7653	26.020/0.7840
		4	25.291/0.7543	25.044/0.7485	25.588/0.7595	25.318/0.7632	25.769/0.7736
		3	24.560/0.7124	24.460/0.7188	24.287/0.6520	24.664/0.6998	24.987/0.7249

the over-fitting issue. Thus, we use the 4×4 by default since the division of 4 can be achieved by shifting bit on the elements of each word set.

4.3 Main quantitative comparisons

In this subsection, we provide quantitative results of EDSR and RDN across various bit-widths. We re-implement all compared methods with the same full-precision model as the initialization for a fair comparison. Comparisons of using different full-precision models as the initialization are provided in Subsection 4.4.

4.3.1 *EDSR*

Table 3 presents results of EDSR×2 and EDSR×4. Our DFSQ obtains the best performance across different datasets and bit-widths. When performing high-bit PTQ (6 or 8), our DFSQ achieves comparable performance to the full-precision counterpart. For instance, on 8-bit and 6-bit EDSR×2, DFSQ obtains 32.101 and 32.099 dB PSNR on BSD100, which only gives a drop of 0.001 and 0.003 dB compared with the full-precision model, respectively. Also, results of 8-bit and 6-bit EDSR×4 on BSD100 demonstrate that DFSQ only incurs 0.004 and 0.007 dB PSNR drops, respectively. It is worth emphasizing that the performance superiority of DFSQ stands out as the bit-width goes down. Taking results of EDSR×2 on Urban100 as the example, compared with the best competitor, DFSQ obtains gains of 0.009 dB PSNR for 8-bit. For 6-, 4-, and 3-bit cases, DFSQ brings improvement of 0.056, 0.242, and 0.492 dB PSNR, respectively. Results of EDSR×4 also provide a similar conclusion. For example, on Urban100, our DFSQ

Table 4 PSNR/SSIM results of compared methods and our DFSQ in quantizing RDN [14] of scales ×2 and ×4. Results of the full-precision model are presented below the dataset name.

Model	Dataset	Bit	BRECQ	QDROP	Min-Max	PTQ4SR	DFSQ (ours)
RDN ×2	Set5 38.053/0.9607	8	38.019/0.9603	38.020/0.9604	38.049/0.9606	37.361/0.9590	38.053/0.9607
		6	37.884/0.9588	37.873/0.9591	37.975/0.9599	37.057/0.9580	38.042/0.9606
		4	37.143/0.9538	37.050/0.9540	37.172/0.9544	36.956/0.9564	37.786/0.9593
		3	36.135/0.9469	36.000/0.9469	35.872/0.9464	36.312/0.9499	37.125/0.9559
	Set14 33.594/0.9174	8	33.576/0.9172	33.566/0.9172	33.560/0.9175	32.933/0.9150	33.589/0.9174
		6	33.464/0.9158	33.417/0.9160	33.530/0.9168	32.713/0.9135	33.585/0.9173
		4	32.949/0.9106	32.825/0.9101	33.136/0.9108	32.696/0.9121	33.373/0.9154
		3	32.322/0.9032	32.158/0.9012	32.297/0.8996	32.376/0.9058	32.896/0.9105
	BSD100 32.197/0.8998	8	32.185/0.8995	32.187/0.8996	32.193/0.8998	31.789/0.8971	32.195/0.8998
		6	32.120/0.8982	32.115/0.8985	32.167/0.8991	31.613/0.8954	32.184/0.8996
		4	31.761/0.8932	31.689/0.8934	31.818/0.8913	31.608/0.8939	32.043/0.8973
		3	31.281/0.8859	31.191/0.8851	31.133/0.8782	31.329/0.8878	31.675/0.8919
	Urban100 32.125/0.9286	8	32.088/0.9282	32.051/0.9282	32.014/0.9281	30.254/0.9145	32.115/0.9285
		6	31.827/0.9260	31.713/0.9257	31.975/0.9274	29.870/0.9095	32.079/0.9281
		4	30.681/0.9150	30.367/0.9128	31.418/0.9193	29.834/0.9087	31.594/0.9217
		3	29.532/0.8989	29.214/0.8930	30.086/0.8998	29.444/0.8996	30.504/0.9080
RDN ×4	Set5 32.244/0.8959	8	32.233/0.8953	32.230/0.8954	32.238/0.8956	30.696/0.8726	32.244/0.8959
		6	32.148/0.8929	32.141/0.8930	32.191/0.8941	30.948/0.8773	32.228/0.8955
		4	31.498/0.8801	31.341/0.8794	31.619/0.8798	31.290/0.8785	31.932/0.8895
		3	30.509/0.8586	30.455/0.8602	30.430/0.8546	30.546/0.8489	31.077/0.8718
	Set14 28.669/0.7838	8	28.657/0.7834	28.650/0.7834	28.642/0.7835	27.596/0.7646	28.663/0.7837
		6	28.582/0.7811	28.563/0.7812	28.616/0.7822	27.753/0.7682	28.653/0.7834
		4	28.139/0.7701	28.032/0.7696	28.303/0.7701	28.011/0.7687	28.471/0.7774
		3	27.516/0.7528	27.468/0.7537	27.621/0.7478	27.549/0.7432	27.941/0.7616
	BSD100 27.627/0.7379	8	27.620/0.7375	27.621/0.7376	27.618/0.7377	27.080/0.7258	27.625/0.7378
		6	27.575/0.7354	27.572/0.7360	27.597/0.7365	27.167/0.7281	27.616/0.7375
		4	27.305/0.7261	27.249/0.7268	27.367/0.7245	27.272/0.7261	27.504/0.7326
		3	26.918/0.7123	26.916/0.7143	26.889/0.7037	26.914/0.7002	27.158/0.7195
	Urban100 26.293/0.7924	8	26.262/0.7916	26.245/0.7915	26.182/0.7904	24.699/0.7454	26.292/0.7924
		6	26.116/0.7875	26.061/0.7870	26.157/0.7891	24.873/0.7521	26.266/0.7914
		4	25.448/0.7662	25.292/0.7636	25.789/0.7720	25.211/0.7589	25.960/0.7780
		3	24.700/0.7351	24.590/0.7331	24.921/0.7340	24.776/0.7259	25.156/0.7442

improves the PSNR by 0.007, 0.023, 0.181, and 0.323 dB for 8-, 6-, 4-, and 3-bit cases, respectively. Despite fine-tuning the weights, optimization-based BRECQ, QDROP, and PTQ4SR exhibit lower performance than the simple Min-Max at most bit-widths, indicating a severe over-fitting issue. In contrast, our DFSQ does not need any fine-tuning, and still achieves stable performance.

4.3.2 *RDN*

RDN results are presented in Table 4. Similarly, DFSQ obtains the best performance over different bit-widths and datasets. For the high-bit cases, DFSQ provides comparable performance to the full-precision model. For example, on 8-bit and 6-bit RDN×2, DFSQ obtains 32.915 and 32.184 dB PSNR on BSD100, corresponding to 0.002 and 0.013 dB drops. While on 8-bit and 6-bit RDN×4, DFSQ only incurs decreases of 0.002 and 0.011 dB on BSD100, respectively. In contrast, all compared methods suffer from larger degradation than DFSQ. Also, the performance advantage of our DFSQ becomes increasingly apparent as the bit-widths decrease. In particular, for RDN×2 on Urban100, DFSQ improves the PSNR by 0.027, 0.104, 0.176, and 0.418 dB on 8-, 6-, 4-, and 3-bit, respectively. On BSD100, DFSQ's improvements are 0.002, 0.017, 0.225, and 0.346 dB PSNR for 8-, 6-, 4-, and 3-bit, respectively. For RDN×4 on Urban100, our DFSQ obtains performance gains by 0.03, 0.109, 0.171, and 0.235 dB PSNR on 8-, 6-, 4-, and 3-bit, respectively. On BSD100, DFSQ's gains are 0.004, 0.019, 0.137, and 0.240 dB PSNR for 8-, 6-, 4-, and 3-bit, respectively. Moreover, it can be observed that the optimization-based methods do not even give higher results than the Min-Max methods for 6-bit and 4-bit cases.

Table 5 PSNR/SSIM results of the compared PTQ4SR [35] and our DFSQ in quantizing EDSR [13] of scales ×2 and ×4. Results of the full-precision model are presented below the dataset name. The results of PTQ4SR are drawn from the original paper.

Model	Dataset	Bit	PTQ4SR	DFSQ (ours)
EDSR ×2	Set5 38.193/0.961	8	38.120/0.960	38.192/0.9610
		6	37.896/0.958	38.191/0.9610
		4	36.327/0.942	38.168/0.9610
	Set14 33.948/0.920	8	33.850/0.920	33.921/0.9202
		6	33.675/0.918	33.930/0.9203
		4	32.753/0.904	33.851/0.9197
	BSD100 32.352/0.902	8	32.313/0.901	32.342/0.9018
		6	32.186/0.899	32.339/0.9017
		4	31.477/0.884	32.322/0.9014
	Urban100 32.967/0.936	8	32.810/0.935	32.924/0.9356
		6	32.452/0.932	32.922/0.9357
		4	30.900/0.913	32.807/0.9346
EDSR ×4	Set5 32.485/0.899	8	32.460/0.898	32.485/0.8987
		6	32.300/0.894	32.481/0.8985
		4	31.203/0.867	32.365/0.8962
	Set14 28.815/0.788	8	28.763/0.787	28.789/0.7875
		6	28.653/0.784	28.760/0.7871
		4	27.977/0.760	28.734/0.7856
	BSD100 27.721/0.742	8	27.695/0.741	27.721/0.7420
		6	27.627/0.738	27.712/0.7418
		4	27.085/0.714	27.681/0.7404
	Urban100 26.646/0.804	8	26.567/0.802	26.639/0.8036
		6	26.382/0.797	26.618/0.8030
		4	25.556/0.764	26.545/0.7998

4.4 More quantitative comparisons

We provide more comparisons between our method and other methods in this subsection. All results of the compared methods are directly drawn from their paper. Note that the performance of the full-precision models is different from ours. For a fair comparison, we report the performance drop between the full-precision model and the quantized model.

4.4.1 *Comparisons with PTQ4SR*

EDSR. Note that the EDSR used in PTQ4SR's paper has 256 channels and 32 blocks, which is larger than the EDSR used in our main paper. To avoid ambiguity, we denote the EDSR that has a larger structure as EDSR-L in this paper. For a comprehensive comparison, we also implement our method on EDSR-L and provide the results in Table 5.

It can be seen that our DFSQ significantly improves performance. Specifically, DFSQ outperforms PTQ4SR by 0.029, 0.153, and 0.845 dB PSNR on 8-, 6-, and 4-bit EDSR-L×2 on BSD100, respectively. On Urban100, DFSQ exhibits 0.114, 0.470, and 1.907 dB PSNR gains compared with PTQ4SR, respectively. Also, DFSQ presents its superiority when quantizing EDSR-L×4 to various bit-widths. For instance, compared with PTQ4SR, our DFSQ achieves 0.026, 0.107, and 0.757 dB PSNR increases on Set14, respectively. While on Urban100, DFSQ presents 0.006, 0.085, and 0.569 dB PSNR increase, respectively. Results of other datasets also exhibit the same conclusion. We also emphasize that our DFSQ's performance gains are more evident as the bit-widths go down, while PTQ4SR suffers from severe performance degradation, demonstrating the adaptability of DFSQ on low-bit cases.

SRResNet. Table 6 presents the comparisons between PTQ4SR and DFSQ on SRResNet. The full-precision model of PTQ4SR has a higher performance than ours. A fair comparison would be considering the performance gaps between the full-precision model and the quantized model. Thus, we refer the readers to focus on the numbers in the subscript within the table, which presents the performance gaps.

As shown in Table 6, the proposed DFSQ achieves better results compared with PTQ4SR. Specifically, for SRResNet×2 on Set5, our DFSQ leads to 0.009, 0.037, and 0.503 dB PSNR drops for 8-, 6-, and 4- cases, respectively, compared with the full-precision model. In contrast, PTQ4SR suffers from more

Table 6 PSNR/SSIM results of the compared PTQ4SR [35] and our DFSQ in quantizing SRResNet [39] of scales ×2 and ×4. Results of the full-precision model are presented in the column of "FP". † denotes the results of the full-precision model of PTQ4SR and ‡ denotes the results of our full-precision model. The number in the subscript represents the performance loss compared to the full precision model.

Model	Dataset	FP	Bit	PTQ4SR	DFSQ (ours)
SRResNet ×2	Set5	38.091/0.961† 37.889/0.960‡	8	$38.032_{0.052}/0.960_{0.001}$	$37.880_{\mathbf{0.009}}/0.960_{\mathbf{0.000}}$
			6	$37.811_{0.280}/0.959_{0.002}$	$37.852_{\mathbf{0.037}}/0.959_{\mathbf{0.001}}$
			4	$36.487_{1.604}/0.951_{0.010}$	$37.386_{\mathbf{0.503}}/0.957_{\mathbf{0.003}}$
	Set14	33.752/0.919† 33.400/0.916‡	8	$33.648_{0.104}/0.919_{0.000}$	$33.398_{\mathbf{0.002}}/0.916_{0.000}$
			6	$33.295_{0.457}/0.916_{0.003}$	$33.373_{\mathbf{0.027}}/0.915_{\mathbf{0.001}}$
			4	$32.404_{1.348}/0.904_{0.015}$	$32.952_{\mathbf{0.448}}/0.909_{\mathbf{0.007}}$
	BSD100	32.241/0.900† 32.077/0.898‡	8	$32.212_{0.029}/0.900_{0.000}$	$32.071_{\mathbf{0.006}}/0.898_{0.000}$
			6	$32.068_{0.173}/0.898_{0.002}$	$32.037_{\mathbf{0.040}}/0.897_{\mathbf{0.001}}$
			4	$31.357_{0.884}/0.885_{0.005}$	$31.655_{\mathbf{0.422}}/0.894_{\mathbf{0.004}}$
	Urban100	32.367/0.931† 31.602/0.923‡	8	$32.210_{0.427}/0.930_{0.001}$	$31.583_{\mathbf{0.019}}/0.923_{\mathbf{0.000}}$
			6	$31.719_{0.918}/0.926_{0.005}$	$31.485_{\mathbf{0.117}}/0.922_{\mathbf{0.001}}$
			4	$29.896_{2.741}/0.904_{0.027}$	$30.708_{\mathbf{0.894}}/0.911_{\mathbf{0.012}}$
SRResNet ×4	Set5	32.234/0.896† 32.066/0.893‡	8	$32.207_{0.027}/0.895_{0.001}$	$32.063_{\mathbf{0.003}}/0.893_{\mathbf{0.000}}$
			6	$32.089_{0.145}/0.892_{0.004}$	$32.016_{\mathbf{0.006}}/0.892_{\mathbf{0.001}}$
			4	$31.146_{1.008}/0.878_{0.018}$	$31.644_{\mathbf{0.422}}/0.886_{\mathbf{0.007}}$
	Set14	28.656/0.784† 28.497/0.780‡	8	$28.619_{0.037}/0.783_{0.001}$	$28.491_{\mathbf{0.006}}/0.780_{\mathbf{0.000}}$
			6	$28.504_{0.152}/0.779_{0.005}$	$28.477_{\mathbf{0.020}}/0.779_{\mathbf{0.001}}$
			4	$27.889_{0.767}/0.763_{0.021}$	$28.185_{\mathbf{0.312}}/0.770_{\mathbf{0.010}}$
	BSD100	27.630/0.738† 27.516/0.735‡	8	$27.618_{0.012}/0.738_{0.000}$	$27.513_{\mathbf{0.003}}/0.735_{0.000}$
			6	$27.561_{0.114}/0.733_{0.005}$	$27.491_{\mathbf{0.025}}/0.734_{\mathbf{0.001}}$
			4	$27.152_{0.478}/0.718_{0.020}$	$27.241_{\mathbf{0.275}}/0.724_{\mathbf{0.011}}$
	Urban100	26.229/0.791† 25.858/0.779‡	8	$26.191_{0.038}/0.790_{0.001}$	$25.855_{\mathbf{0.003}}/0.779_{\mathbf{0.000}}$
			6	$26.011_{0.218}/0.783_{0.008}$	$25.808_{\mathbf{0.050}}/0.777_{\mathbf{0.002}}$
			4	$25.133_{1.096}/0.753_{0.038}$	$25.300_{\mathbf{0.558}}/0.756_{\mathbf{0.023}}$

severe performance loss. In particular, for 8-, 6-, and 4- cases, PTQ4SR respectively suffers from 0.052, 0.280, and 1.604 dB PSNR drops as compared to the full-precision model. For SRResNet×2 on BSD100, DFSQ incurs 0.006, 0.040, and 0.422 dB PSNR drops for 8-, 6-, and 4-bit cases, respectively. While PTQ4SR leads to 0.029, 0.173, and 0.884 dB PSNR drops. The results of SRResNet×4 also give the same conclusion. Specifically, compared with the full-precision model, DFSQ respectively obtains 0.003, 0.025, and 0.275 dB PSNR drops on BSD100 for 8-, 6-, and 4-bit cases. In contrast, PTQ4SR leads to 0.012, 0.114, and 0.478 dB PSNR drops, which are more severe than DFSQ. In summary, the performance degradation of the proposed DFSQ is much smaller than PTQ4SR, clearly demonstrating the advantage of DFSQ. Moreover, compared with PTQ4SR, the proposed DFSQ typically has small performance degradation when it comes to low-bit cases such as 4-bit and 6-bit. Such results prove the ability of DFSQ to fully utilize the limited representability of low-bit numbers. Moreover, SRResNet is a BN-inserted SR model. The performance gains of our method prove DFSQ's generalizability in handling the activation distributions with the existence of the BN layer.

4.4.2 *Comparisons with HPTQ*

HPTQ [56] is a hybrid post-training method that consists of piecewise quantization, mix-precision quantization, and clustered quantization. We emphasize that the full-precision model used in HPTQ has a different performance from ours. Thus, for a fair comparison, we report the performance gap between the full-precision model and the quantized model and remind readers of that. Note that HPTQ only reports the results of mix-precision quantization, while DFSQ adopts uniform bit-width for each layer of the SR models. Despite such difference, our DFSQ still outperforms HPTQ by a large margin as shown in Table 7.

For instance, on EDSR×4, our DFSQ only incurs 0.000 and 0.200 dB PSNR drops for 8-bit and 4-bit cases when tested on Set5, respectively. In contrast, HPTQ suffers from significant performance degradation, i.e., 0.16 dB PSNR drop for 8-bit and 3.99 dB PSNR drop for 4-bit. On Urban100, HPTQ incurs 0.36 dB PSNR and 2.97 dB PSNR losses for 8-bit and 4-bit cases, respectively. Our DFSQ instead

Table 7 PSNR/SSIM results of the compared HPTQ [56] and our DFSQ in quantizing EDSR [13] and RDN [14] of scales ×4. Note that HPTQ is a mix-precision method, which is denoted as "MP". Results of the full-precision model are presented in the column of "FP". † denotes the results of the full-precision model of HPTQ and ‡ denotes the results of our full-precision model. The number in the subscript represents the performance loss compared to the full precision model.

Model	Dataset	FP	Bit	HPTQ_{MP}	DFSQ (ours)
EDSR ×4	Set5	32.14/–†	8	$31.98_{0.16}$/–	$32.485_{\mathbf{0.000}}$/$0.899_{\mathbf{0.000}}$
		32.485/0.899‡	4	$28.15_{3.99}$/–	$32.365_{\mathbf{0.200}}$/$0.896_{\mathbf{0.003}}$
	Set14	28.75/–†	8	$28.65_{0.10}$/–	$28.789_{\mathbf{0.026}}$/$0.788_{\mathbf{0.000}}$
		28.815/0.788‡	4	$25.91_{2.84}$/–	$28.734_{\mathbf{0.081}}$/$0.786_{\mathbf{0.002}}$
	BSD100	28.51/–†	8	$28.48_{0.03}$/–	$27.721_{\mathbf{0.000}}$/$0.742_{\mathbf{0.000}}$
		27.721/0.742‡	4	$26.10_{2.41}$/–	$27.681_{\mathbf{0.040}}$/$0.740_{\mathbf{0.002}}$
	Urban100	26.07/–†	8	$25.71_{0.36}$/–	$26.639_{\mathbf{0.007}}$/$0.804_{\mathbf{0.000}}$
		26.646/0.804‡	4	$23.10_{2.97}$/–	$26.545_{\mathbf{0.101}}$/$0.800_{\mathbf{0.004}}$
RDN ×4	Set5	32.40/–†	8	$32.27_{0.13}$/–	$32.244_{\mathbf{0.000}}$/$0.896_{\mathbf{0.000}}$
		32.244/0.896‡			
	Set14	28.91/–†	8	$28.81_{0.10}$/–	$28.663_{\mathbf{0.006}}$/$0.784_{\mathbf{0.000}}$
		28.669/0.784‡			
	BSD100	28.62/–†	8	$28.64_{0.08}$/–	$27.625_{\mathbf{0.002}}$/$0.738_{\mathbf{0.000}}$
		27.627/0.738‡			
	Urban100	26.40/–†	8	$26.26_{0.11}$/–	$26.292_{\mathbf{0.001}}$/$0.792_{\mathbf{0.000}}$
		26.293/0.792‡			

Table 8 Complexity analysis. The number in brackets indicates the parameter amount. BOPs are computed by generating a 1920×1080 image (upscaling factor ×4).

Model	Bit	Baselines	DFSQ (ours)
EDSR×4	4	204T (0.49M)	222T (0.49M)
RDN×4	4	271T (3.13M)	307T (3.13M)
SRResNet×4	4	278T (0.51M)	282T (0.51M)

only presents 0.007 dB PSNR and 0.101 dB PSNR losses. On RDN×4, our DFSQ also demonstrates its effectiveness compared with HPTQ. For 8-bit RDN×4 on Set5, DFSQ only incurs 0.10 dB PSNR drop while HPTQ suffers from 0.006 dB PSNR drop. On Urban100, the PSNR drop of DFSQ is 0.001 dB while HPTQ is 0.11 dB. In summary, DFSQ achieves less performance degradation than HPTQ even though the latter adopts mix-precision quantization.

4.5 Model complexity

Table 8 provides the complexity analyses of 4-bit SR models including EDSR×4, RDN×4, and SRResNet×4. We present the bit operations (BOPs) [69] as the metrics to assess the complexity of DFSQ. Due to the bit-shifting operation being extremely efficient [70,71], we regard one bit-shifting operation as a 1-bit operation. We compute BOPs by generating a 1920×1080 image (upscaling factor ×4). The baselines are the methods adopting uniform quantization such as BRECQ, QDROP, Min-Max, and PTQ4SR. It can be seen that DFSQ exhibits complexity advantages.

4.6 Qualitative results

Figures 4 and 5 exhibit the qualitative results of the 4-bit EDSR×4, 4-bit RDN×4, 4-bit EDSR×2, and 4-bit RDN×2, respectively. The reported PSNR/SSIM are measured by the displayed image. Here, we select representative images for the purpose of better meeting human perceptual quality. It can be seen that our method obtains the best visualization results compared with other methods, which demonstrates the superiority of our DFSQ.

4.7 Intermediate results of the fast selection strategy

Figure 6 displays the selected quantization points of the fast selection strategy. It can be seen that as the iteration increases, the selected quantization points gradually approximate the input distribution. Finally, the fast selection strategy produces non-uniform cluster clusters, effectively preserving the activation distribution.

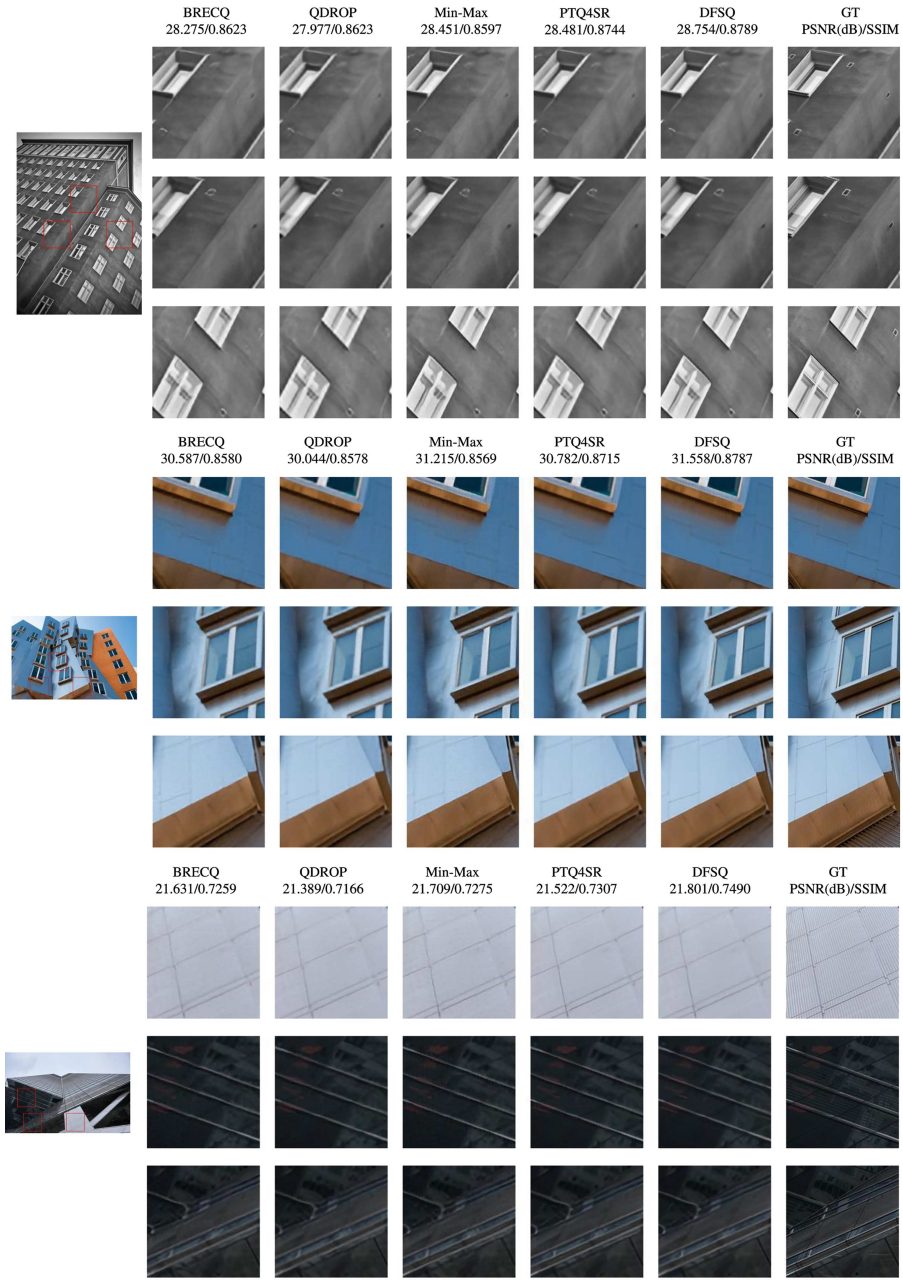

Figure 4 (Color online) Illustration of the qualitative results of 4-bit EDSR×4.

5 Results on image restoration

In this section, we further apply the proposed DFSQ to the model designed for image restoration tasks. We chose to quantize the famous AirNet [72]. We adopt the model that is trained on all degradations in an all-in-one manner. Following the setting of quantizing SR models, we remain the encoder and the first/last layer of the AirNet full-precision. The calibration dataset contains 32 images random sampled from the training set of AirNet. The models are tested on three challenging datasets including BSD68 [65] for denoising, Rain100L [73] for deraining, and synthetic objective testing set (SOTS) [74] for dehazing. All PSNR and SSIM are computed by using the default code of the official code.

Results are presented in the Table 9. It can be seen that the proposed DFSQ achieves the best performance across different bit-widths and datasets. Specifically, for the high-bit cases, DFSQ provides comparable performance to the full-precision model. For example, on 8-bit and 6-bit AirNet, DFSQ obtains 33.91 and 33.90 dB PSNR on BSD68 ($\sigma = 15$). In addition, as the bit-width decreases, the

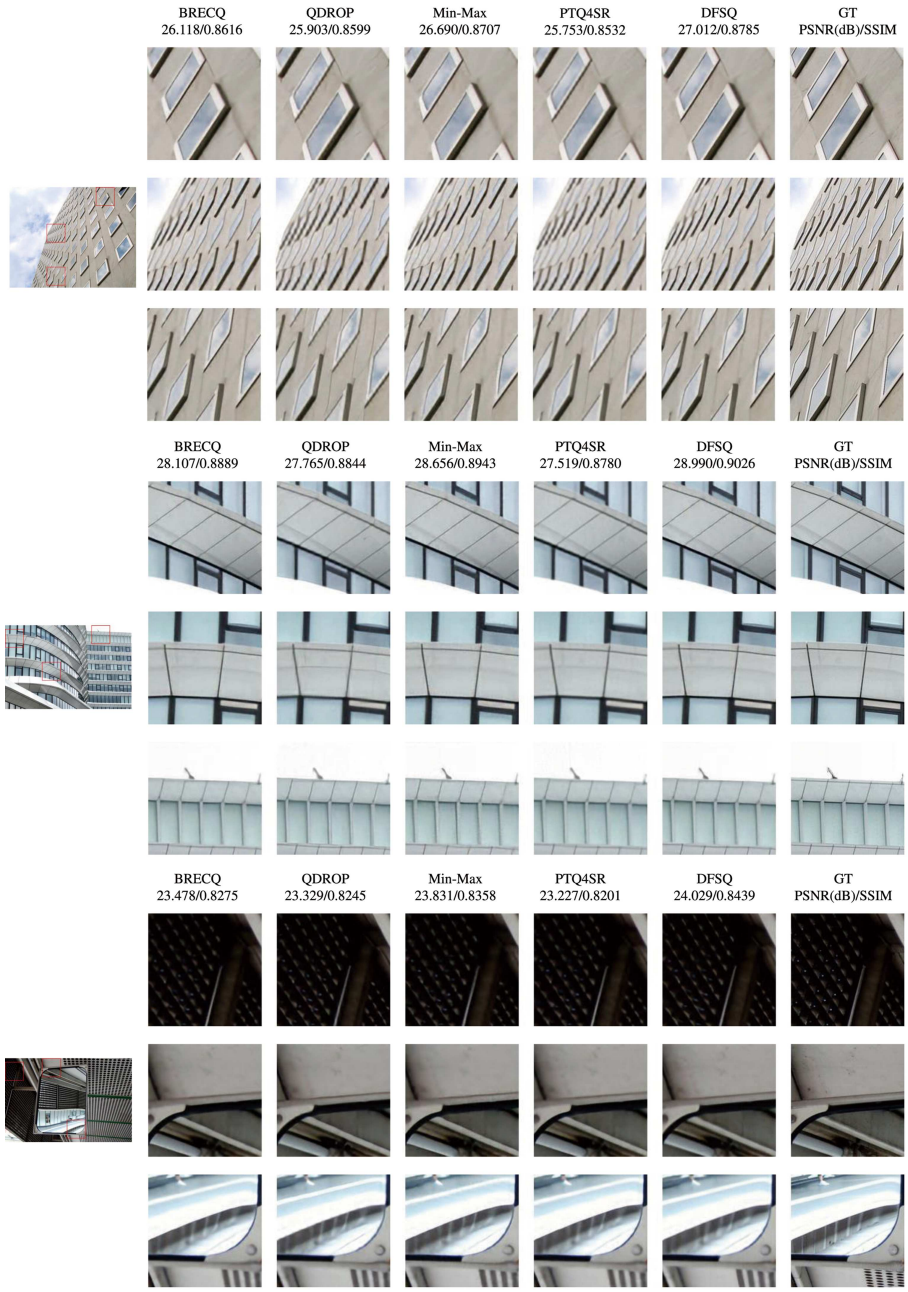

Figure 5 (Color online) Illustration of the qualitative results of 4-bit RDN×4.

performance advantage of our DFSQ becomes increasingly obvious. For example, DFSQ improves the 4-bit AirNet by 1.94, 3.11, and 4.11 dB PSNR on BSD68 (σ15), Rain100L, and SOTS, respectively. For the 3-bit case, DFSQ improves the 4-bit AirNet by 3.72, 4.5, and 3.53 dB PSNR on BSD68 (σ15), Rain100L, and SOTS, respectively. The above results demonstrate that the proposed DFSQ is also applicable to image restoration tasks and showcases the best performance compared with the baseline methods.

6 Conclusion

We have presented a novel quantization method, termed DFSQ for PTQ on SR networks. We discover that the activation distribution of SR models exhibits significant variance between samples and channels. Thus, DFSQ suggests conducting a channel-wise normalization for activation at first, then applying the

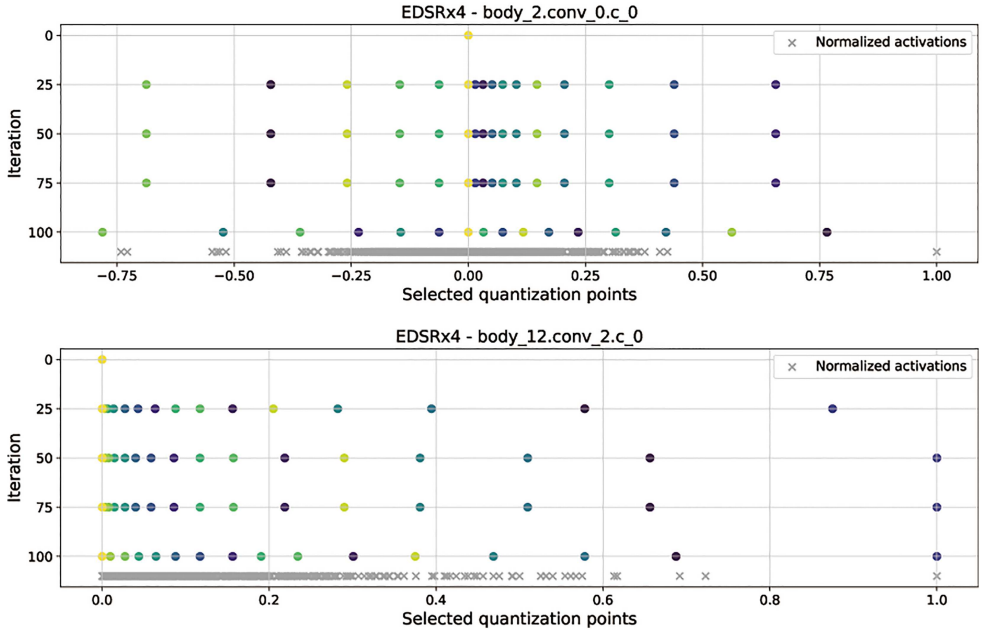

Figure 6 (Color online) Intermediate results of the K-means. Results are drawn from 4-bit EDSR×4.

Table 9 PSNR/SSIM results of the compared methods and our DFSQ in quantizing AirNet [72] on the image restoration tasks. BSD68, Rain100L, and SOTS are the datasets for denoising, deraining, and dehazing, respectively. σ denotes the different degradation levels. Results of the full-precision model are presented below the dataset name.

Model	Dataset	Bit	BRECQ	QDROP	Min-Max	PTQ4SR	DFSQ (ours)
AirNet	BSD68 (σ = 15) 33.92/0.9329	8	32.97/0.9290	32.95/0.9286	32.99/0.9291	32.97/0.9290	33.91/0.9328
		6	32.79/0.9261	32.81/0.9257	32.82/0.9260	32.74/0.9254	33.90/0.9327
		4	31.24/0.8752	31.37/0.8809	31.26/0.8762	30.89/0.8704	33.31/0.9244
		3	26.72/0.7519	27.60/0.7357	27.28/0.7371	28.09/0.7560	31.81/0.8902
	BSD68 (σ = 25) 31.26/0.8884	8	30.47/0.8797	30.49/0.8799	30.49/0.8798	30.47/0.8798	31.25/0.8880
		6	30.46/0.8765	30.50/0.8760	30.49/0.8767	30.44/0.8762	31.23/0.8869
		4	29.36/0.8090	29.35/0.8098	29.42/0.8170	29.20/0.8033	30.76/0.8745
		3	25.74/0.6309	25.26/0.6480	25.66/0.6324	26.47/0.6544	29.32/0.8139
	BSD68 (σ = 50) 28.00/0.7974	8	27.80/0.7918	27.81/0.7921	27.82/0.7924	27.82/0.7922	28.00/0.7972
		6	27.76/0.7868	27.76/0.7866	27.75/0.7866	27.74/0.7858	27.98/0.7946
		4	26.13/0.6752	26.01/0.6801	26.19/0.6817	26.03/0.6707	27.48/0.7615
		3	22.17/0.4529	21.77/0.4497	21.80/0.4420	22.97/0.4818	25.85/0.6474
	Rain100L 34.90/0.9675	8	33.26/0.9507	33.28/0.9510	33.29/0.9511	33.26/0.9507	34.87/0.9672
		6	33.06/0.9463	33.02/0.9456	33.07/0.9463	33.04/0.9459	34.78/0.9661
		4	30.18/0.8708	30.69/0.8843	30.40/0.8746	30.06/0.8679	33.80/0.9520
		3	25.35/0.7119	24.93/0.7306	25.07/0.7165	25.93/0.7301	30.43/0.8971
	SOTS 27.94/0.9615	8	27.54/0.9551	27.55/0.9551	27.56/0.9552	27.54/0.9551	27.90/0.9600
		6	27.18/0.9488	27.15/0.9480	27.19/0.9490	27.17/0.9483	27.70/0.9574
		4	22.98/0.8708	22.97/0.8701	23.04/0.8660	22.83/0.8599	27.15/0.9468
		3	19.69/0.7457	19.83/0.7381	20.06/0.7371	20.73/0.7562	24.26/0.9040

hardware-friendly and DFSQ, in which the quantization points are selected from a universal set consisting of multi-word additive log-scale values. To select quantization points, we propose a fast quantization points selection strategy with linear time complexity. We perform K-means clustering to identify the closest quantization points to centroids from the universal set. Our superiority over many competitors is demonstrated on different quantized SR models across various bit-widths and benchmarks. First, despite DFSQ obtaining decent performance, there remains a performance gap between quantized models and full-precision models. To mitigate this issue, optimizing network weight carefully and applying mixed-precision quantization [75] offers a promising avenue. Second, the overhead of channel-wise normalization can be reduced by using the low-bit format as in [23, 76]. Finally, DFSQ is very suitable for erratic

distribution and can be further applied in many other models such as large language models, which is left in the future.

Acknowledgements This work was supported by National Key R&D Program of China (Grant No. 2022ZD0118202), National Science Fund for Distinguished Young Scholars (Grant No. 62025603), National Natural Science Foundation of China (Grant Nos. U21B2037, U22B2051, 62176222, 62176223, 62176226, 62072386, 62072387, 62072389, 62002305, 62272401), and Natural Science Foundation of Fujian Province of China (Grant Nos. 2021J01002, 2022J06001).

References

1 Greenspan H. Super-resolution in medical imaging. Comput J, 2008, 52: 43–63
2 Isaac J S, Kulkarni R. Super resolution techniques for medical image processing. In: Proceedings of the International Conference on Technologies for Sustainable Development (ICTSD), 2015. 1–6
3 Georgescu M I, Ionescu R T, Miron A I, et al. Multimodal multi-head convolutional attention with various kernel sizes for medical image super-resolution. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2023. 2195–2205
4 Yang H, Wang Z, Liu X, et al. Deep learning in medical image super resolution: a review. Appl Intell, 2023, 53: 20891–20916
5 Kappeler A, Yoo S, Dai Q, et al. Video super-resolution with convolutional neural networks. IEEE Trans Comput Imag, 2016, 2: 109–122
6 Liu H, Ruan Z, Zhao P, et al. Video super-resolution based on deep learning: a comprehensive survey. Artif Intell Rev, 2022, 55: 5981–6035
7 Xiao Y, Yuan Q, Zhang Q, et al. Deep blind super-resolution for satellite video. IEEE Trans Geosci Remote Sens, 2023, 61: 1–16
8 Xiao Y, Yuan Q, He J, et al. Space-time super-resolution for satellite video: a joint framework based on multi-scale spatial-temporal transformer. Int J Appl Earth Observation GeoInf, 2022, 108: 102731
9 Xiao Y, Yuan Q, Jiang K, et al. Local-global temporal difference learning for satellite video super-resolution. IEEE Trans Circ Syst Video Technol, 2024, 34: 2789–2802
10 Xiao Y, Su X, Yuan Q, et al. Satellite video super-resolution via multiscale deformable convolution alignment and temporal grouping projection. IEEE Trans Geosci Remote Sens, 2022, 60: 1–19
11 Xiao Y, Yuan Q, Jiang K, et al. EDiffSR: an efficient diffusion probabilistic model for remote sensing image super-resolution. IEEE Trans Geosci Remote Sens, 2024, 62: 1–14
12 Xiao Y, Yuan Q, Jiang K, et al. From degrade to upgrade: learning a self-supervised degradation guided adaptive network for blind remote sensing image super-resolution. Inf Fusion, 2023, 96: 297–311
13 Lim B, Son S, Kim H, et al. Enhanced deep residual networks for single image super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2017. 136–144
14 Zhang Y, Tian Y, Kong Y, et al. Residual dense network for image super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018. 2472–2481
15 Dong C, Loy C C, He K, et al. Learning a deep convolutional network for image super-resolution. In: Proceedings of the European Conference on Computer Vision (ECCV), 2014. 184–199
16 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), 2016. 1646–1654
17 Zhang Y, Li K, Li K, et al. Image super-resolution using very deep residual channel attention networks. In: Proceedings of the European Conference on Computer Vision (ECCV), 2018. 286–301
18 Lin M, Ji R, Wang Y, et al. Hrank: filter pruning using high-rank feature map. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020. 1529–1538
19 Krishnamoorthi R. Quantizing deep convolutional networks for efficient inference: a whitepaper. 2018. ArXiv:1806.08342
20 Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. 2015. ArXiv:1503.02531
21 Han S, Pool J, Tran J, et al. Learning both weights and connections for efficient neural network. In: Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), 2015. 1135–1143
22 Wang H, Chen P, Zhuang B, et al. Fully quantized image super-resolution networks. In: Proceedings of the 29th ACM International Conference on Multimedia (ACMMM), 2021. 639–647
23 Hong C, Kim H, Baik S, et al. 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), 2022. 2675–2684
24 Jiang X, Wang N, Xin J, et al. Training binary neural network without batch normalization for image super-resolution. In: Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), 2021. 1700–1707
25 Xin J, Wang N, Jiang X, et al. Binarized neural network for single image super resolution. In: Proceedings of the European Conference on Computer Vision (ECCV), 2020. 91–107
26 Ma Y, Xiong H, Hu Z, et al. Efficient super resolution using binarized neural network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2019. 694–703
27 Li H, Yan C, Lin S, et al. PAMS: quantized super-resolution via parameterized max scale. In: Proceedings of the European Conference on Computer Vision (ECCV), 2020. 564–580
28 Hong C, Baik S, Kim H, et al. CADyQ: content-aware dynamic quantization for image super-resolution. In: Proceedings of the European Conference on Computer Vision (ECCV), 2022. 367–383
29 Zhong Y, Lin M, Li X, et al. Dynamic dual trainable bounds for ultra-low precision super-resolution networks. In: Proceedings of the European Conference on Computer Vision (ECCV), 2022. 1–18
30 Jeon Y, Lee C, Cho E, et al. Mr.BiQ: post-training non-uniform quantization based on minimizing the reconstruction error. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022. 12319–12328

31 Fang J, Shafiee A, Abdel-Aziz H, et al. Post-training piecewise linear quantization for deep neural networks. In: Proceedings of the European Conference on Computer Vision (ECCV), 2020. 69–86

32 Li Y, Gong R, Tan X, et al. BRECQ: pushing the limit of post-training quantization by block reconstruction. In: Proceedings of the International Conference on Learning Representations (ICLR), 2021

33 Nagel M, Amjad R A, van Baalen M, et al. Up or down? Adaptive rounding for post-training quantization. In: Proceedings of the International Conference on Machine Learning (ICML), 2020. 7197–7206

34 Wang P, Chen Q, He X, et al. Towards accurate post-training network quantization via bit-split and stitching. In: Proceedings of the International Conference on Machine Learning (ICML), 2020. 9847–9856

35 Tu Z, Hu J, Chen H, et al. Toward accurate post-training quantization for image super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023. 5856–5865

36 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), 2018. 1664–1673

37 Wei X, Gong R, Li Y, et al. QDROP: randomly dropping quantization for extremely low-bit post-training quantization. In: Proceedings of the International Conference on Learning Representations (ICLR), 2022

38 Oh S, Sim H, Kim J, et al. Non-uniform step size quantization for accurate post-training quantization. In: Proceedings of the European Conference on Computer Vision (ECCV), 2022. 658–673

39 Ledig C, Theis L, Huszár F, 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), 2017. 4681–4690

40 Tong T, Li G, Liu X, et al. Image super-resolution using dense skip connections. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2017. 4799–4807

41 Magid S A, Zhang Y, Wei D, et al. Dynamic high-pass filtering and multi-spectral attention for image super-resolution. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021. 4288–4297

42 Mei Y, Fan Y, Zhou Y, et al. 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), 2020. 5690–5699

43 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), 2021. 3517–3526

44 Liang J, Cao J, Sun G, et al. SwinIR: image restoration using swin transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021. 1833–1844

45 Zamir S W, Arora A, Khan S, et al. Restormer: efficient transformer for high-resolution image restoration. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022. 5728–5739

46 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), 2016. 1637–1645

47 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), 2017. 3147–3155

48 Dong C, Loy C C, Tang X. Accelerating the super-resolution convolutional neural network. In: Proceedings of the European Conference on Computer Vision (ECCV), 2016. 391–407

49 Shi W, Caballero J, Huszár F, et al. 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), 2016. 1874–1883

50 Xiao Y, Yuan Q, Jiang K, et al. TTST: a top-k token selective transformer for remote sensing image super-resolution. IEEE Trans Image Process, 2024, 33: 738–752

51 Lai W S, Huang J B, Ahuja N, et al. Deep laplacian pyramid networks for fast and accurate super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2017. 624–632

52 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), 2018. 252–268

53 Hui Z, Gao X, Yang Y, et al. Lightweight image super-resolution with information multi-distillation network. In: Proceedings of the 27th ACM International Conference on Multimedia (ACM MM), 2019. 2024–2032

54 Luo X, Xie Y, Zhang Y, et al. LatticeNet: towards lightweight image super-resolution with lattice block. In: Proceedings of the European Conference on Computer Vision (ECCV), 2020. 272–289

55 Oh J, Kim H, Nah S, et al. Attentive fine-grained structured sparsity for image restoration. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022. 17673–17682

56 Xu N, Chen X, Cao Y, et al. Hybrid post-training quantization for super-resolution neural network compression. IEEE Signal Process Lett, 2023, 30: 379–383

57 Esser S K, McKinstry J L, Bablani D, et al. Learned step size quantization. In: Proceedings of the International Conference on Learning Representations (ICLR), 2020

58 Li Y, Dong X, Wang W. Additive powers-of-two quantization: an efficient non-uniform discretization for neural networks. In: Proceedings of the International Conference on Learning Representations (ICLR), 2020

59 Oh S, Sim H, Lee S, et al. Automated log-scale quantization for low-cost deep neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021. 742–751

60 Lee S, Sim H, Choi J, et al. Successive log quantization for cost-efficient neural networks using stochastic computing. In: Proceedings of the ACM/IEEE Design Automation Conference (DAC), 2019. 1–6

61 Wikipedia. Quantization (signal processing). 2021. https://en.wikipedia.org/wiki/Quantization_(signal_processing)#Neglecting_the_entropy_constraint:_Lloyd%E2%80%93Max_quantization

62 Hartigan J A, Wong M A. Algorithm AS 136: a K-means clustering algorithm. Appl Stat, 1979, 28: 100

63 Timofte R, Agustsson E, van Gool L, et al. 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), 2017. 114–125

64 Bevilacqua M, Roumy A, Guillemot C, et al. Low-complexity single-image super-resolution based on nonnegative neighbor

embedding. In: Proceedings of the British Machine Vision Conference (BMVC), 2012. 1–10
65 Martin D, Fowlkes C, Tal D, et al. 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), 2001. 416–423
66 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), 2015. 5197–5206
67 Wang Z, Bovik A C, Sheikh H R, et al. Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process, 2004, 13: 600–612
68 Paszke A, Gross S, Massa F, et al. PyTorch: an imperative style, high-performance deep learning library. In: Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), 2019. 8026–8037
69 van Baalen M, Louizos C, Nagel M, et al. Bayesian bits: unifying quantization and pruning. In: Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), 2020. 5741–5752
70 Vogel S, Liang M, Guntoro A, et al. Efficient hardware acceleration of CNNs using logarithmic data representation with arbitrary log-base. In: Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2018. 1–8
71 Patterson D A, Hennessy J L. Computer Organization and Design: The Hardware/Software Interface. Amsterdam: Elsevier, 2005
72 Li B, Liu X, Hu P, et al. All-in-one image restoration for unknown corruption. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022. 17431–17441
73 Yang W, Tan R T, Feng J, et al. Joint rain detection and removal from a single image with contextualized deep networks. IEEE Trans Pattern Anal Mach Intell, 2019, 42: 1377–1393
74 Li B, Ren W, Fu D, et al. Benchmarking single-image dehazing and beyond. IEEE Trans Image Process, 2018, 28: 492–505
75 Wang K, Liu Z, Lin Y, et al. HAQ: hardware-aware automated quantization with mixed precision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019. 8612–8620
76 Dai S, Venkatesan R, Ren M, et al. VS-Quant: per-vector scaled quantization for accurate low-precision neural network inference. 2021. ArXiv:2102.04503