CADyQ: Content-Aware Dynamic Quantization for Image Super-Resolution
– *Supplementary Document* –

Cheeun Hong[1], Sungyong Baik[3], Heewon Kim[1],
Seungjun Nah[1,4], and Kyoung Mu Lee[1,2]

[1] Dept. of ECE & ASRI, {cheeun914, ghimhw, kyoungmu}@snu.ac.kr
[2] IPAI, Seoul National University
[3] Dept. of Data Science, Hanyang University dsybaik@hanyang.ac.kr
[4] NVIDIA seungjun.nah@gmail.com

This supplementary document presents **additional experimental quantitative results** to further demonstrate the effectiveness of our framework in Section A; **additional analyses** of our proposed framework in Section B; **implementation details** in Section C; **complexity analysis** in Section D; and **additional qualitative results** in Section E.

A Additional Experiments

A.1 Effectiveness on Other Quantization Function

In this section, we provide experimental results to demonstrate that the proposed quantization framework is applicable to other quantization functions. In the main text, each quantization function candidate is implemented with PAMS, a linear quantization function with a learnable scale parameter [31]. To demonstrate the generalizability of CADyQ with different quantization functions in this section, we employ another linear quantization function [52], which we refer to as LinQ. The major difference between PAMS and LinQ is that PAMS uses a learnable scale parameter to clip the outliers, whereas LinQ does not clip the outliers and thus takes the whole feature range as the quantization range. Table S1 shows the quantization results of CADyQ with LinQ (referred to as LinQ-CADyQ). Our framework successfully achieves the reduction in average precision while simultaneously achieving the performance similar to or better than baselines or its 8-bit counterpart quantized with LinQ. The results show that the proposed quantization framework is flexible in terms of quantization functions. Note that CADyQ achieves a larger average precision reduction with PAMS than with LinQ. The results corroborate the previous findings that learnable scale parameters are effective for SR networks, which can have variant outliers in feature distributions [31].

A.2 Experiments on SR Networks for Scale $\times$ 2

In addition to the scale $\times 4$ experiments presented in the main manuscript, we additionally evaluate our framework on SR models for scale $\times 2$: IDN [21], EDSR-baseline [34], SRResNet [29], and CARN [2]. As shown in Table S2, compared

Table S1: Quantitative comparisons with other quantization function. The results demonstrate the effectiveness of the proposed framework on SR networks (IDN [21], EDSR-baseline [34], SRResNet [29], and CARN [2]) regardless of the quantization function type

Model	Urban100			Test2K			Test4K		
	FQR$_\downarrow$	PSNR$_\uparrow$	SSIM$_\uparrow$	FQR$_\downarrow$	PSNR$_\uparrow$	SSIM$_\uparrow$	FQR$_\downarrow$	PSNR$_\uparrow$	SSIM$_\uparrow$
IDN [21]	32.00	25.42	0.763	32.00	27.48	0.774	32.00	28.54	0.806
IDN-LinQ [52]	8.00	25.47	0.764	8.00	27.50	0.774	8.00	28.56	0.806
IDN-LinQ-CADyQ (Ours)	6.85	25.60	0.769	6.65	25.72	0.775	6.57	28.58	0.807
EDSR-baseline [34]	32.00	26.04	0.784	32.00	27.71	0.782	32.00	28.80	0.814
EDSR-baseline-LinQ [52]	8.00	25.85	0.777	8.00	27.65	0.780	8.00	28.74	0.812
EDSR-baseline-LinQ-CADyQ (Ours)	7.26	25.96	0.782	7.20	27.68	0.781	7.20	28.78	0.813
SRResNet [29]	32.00	25.74	0.773	32.00	27.60	0.778	32.00	28.68	0.810
SRResNet-LinQ [52]	8.00	25.83	0.777	8.00	27.62	0.780	8.00	28.70	0.812
SRResNet-LinQ-CADyQ (Ours)	6.49	25.87	0.777	6.38	27.61	0.780	6.33	28.69	0.811
CARN [2]	32.00	26.07	0.784	32.00	27.69	0.782	32.00	28.79	0.814
CARN-LinQ [52]	8.00	25.82	0.776	8.00	27.60	0.779	8.00	28.68	0.811
CARN-LinQ-CADyQ (Ours)	6.24	25.88	0.778	6.00	27.63	0.779	5.83	28.71	0.811

to DAQ [17] (4-bit) that suffers from severe performance degradation, CADyQ achieves minimal or no performance degradation from the unquantized baseline model. Compared to PAMS [31] (8-bit), the average feature quantization rate (FQR) is reduced while achieving a similar or better performance, demonstrating the effectiveness of our framework on scale $\times 2$.

Table S2: Quantitative comparisons on various SR networks: IDN [21], EDSR-baseline [34], SRResNet [29], and CARN [2] of **scale $\times$ 2**

Model	Urban100			Test2K			Test4K		
	FQR$_\downarrow$	PSNR$_\uparrow$	SSIM$_\uparrow$	FQR$_\downarrow$	PSNR$_\uparrow$	SSIM$_\uparrow$	FQR$_\downarrow$	PSNR$_\uparrow$	SSIM$_\uparrow$
IDN [21]	32.00	31.29	0.920	32.00	32.42	0.924	32.00	34.02	0.940
IDN-PAMS [31]	8.00	31.39	0.921	8.00	32.46	0.925	8.00	34.05	0.941
IDN-DAQ [17]	4.00	29.15	0.886	4.00	31.33	0.904	4.00	32.70	0.921
IDN-CADyQ (Ours)	5.22	31.54	0.923	4.67	32.51	0.925	4.57	34.10	0.941
EDSR-baseline [34]	32.00	31.97	0.927	32.00	32.75	0.928	32.00	34.37	0.943
EDSR-baseline-PAMS [31]	8.00	31.96	0.927	8.00	32.72	0.928	8.00	34.33	0.943
EDSR-baseline-DAQ [17]	4.00	31.63	0.923	4.00	32.58	0.926	4.00	34.15	0.942
EDSR-baseline-CADyQ (Ours)	6.15	31.95	0.927	5.68	32.70	0.928	5.59	34.30	0.943
SRResNet [29]	32.00	31.44	0.921	32.00	32.51	0.925	32.00	34.11	0.941
SRResNet-PAMS [31]	8.00	31.44	0.922	8.00	32.52	0.925	8.00	34.11	0.941
SRResNet-DAQ [17]	4.00	31.25	0.919	4.00	32.42	0.924	4.00	33.98	0.940
SRResNet-CADyQ (Ours)	6.46	31.58	0.923	6.10	32.61	0.926	6.02	34.19	0.942
CARN [2]	32.00	31.92	0.926	32.00	32.75	0.928	32.00	34.32	0.943
CARN-PAMS [31]	8.00	31.74	0.924	8.00	32.62	0.927	8.00	34.22	0.942
CARN-DAQ [17]	4.00	30.74	0.912	4.00	31.89	0.918	8.00	33.51	0.936
CARN-CADyQ (Ours)	4.32	31.87	0.925	4.04	32.66	0.927	4.03	34.25	0.942

A.3 Fully Quantized Super-Resolution Networks

In most SR quantization works [31, 46], not all layers were quantized. Specifically, the shortcut connections in ResBlocks, the first and the last convolutional layers, were not quantized as they are known to have high quantization sensitivity. FQSR [41] is the first work to quantize all layers and the shortcut connections in SR networks. In Table S3, we show that CADyQ substantially reduces the average precision while maintaining the performance even when quantizing with FQSR on layers, including the ones with high quantization sensitivity. The results outline the effectiveness of the proposed quantization sensitivity estimation measures (*i.e.*, the average gradient magnitude of the given patch and the standard deviation of the layer feature) and the dynamic bit-width selection in the CADyQ framework.

Table S3: Quantitative comparison with fully quantized SR network. Evaluation is conducted with EDSR-baseline [34] as a backbone, on Urban100. The weights of the fully quantized networks are quantized to 8-bit. Computational cost is measured w.r.t FQR which denotes the average feature quantization rate of the quantized layers and BitOPs which is calculated over the whole network

Model	FQR$_\downarrow$	BitOPs$_\downarrow$	PSNR$_\uparrow$	SSIM$_\uparrow$
EDSR-baseline [34]	32.00	317.5G	26.04	0.784
EDSR-baseline-FQSR [41]	8.00	19.8G	25.92	0.781
EDSR-baseline-FQSR-CADyQ	5.90	17.6G	25.90	0.779

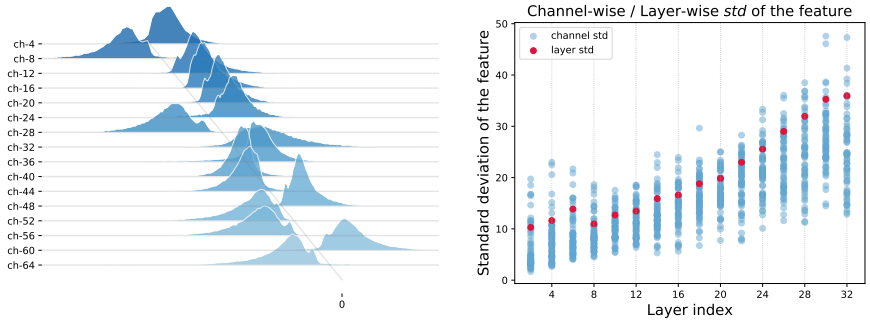
B Additional Analyses

B.1 Analysis on Quantization Sensitivity Estimation

The results reported in Table 3 of the main paper justify the use of the channel-wise standard deviation to estimate the quantization sensitivity of each layer feature. Fig. S1a shows that SR network layers have distinct channel distributions, where the 16-th layer feature distributions of EDSR are illustrated as an example. As each channel has a distinct distribution, the channels have diverse standard deviations, as demonstrated in Fig. S1b. Thus, using layer-wise standard deviation would result in loss of such information on distinct channel distributions, thereby obtaining less accurate quantization sensitivity estimation.

B.2 The Variation of Quantization Bit-Width Candidates

In this section, we investigate the impact of using different combinations of candidate bit-widths, as displayed in Table S4. The results demonstrate that



(a) Channel-wise feature of 16-th layer in EDSR **(b)** Channel-wise vs layer-wise standard deviation of each layer

Fig. S1: Visualization of feature distributions in SR networks

using different combinations of candidates results in different trade-offs between the computational complexity (FQR) and the restoration performance (PSNR and SSIM). Such model behaviors can provide us the flexibility in selecting a model with the desired computational complexity and performance to target for specific applications, ranging from mobile devices to 4K displays. For instance, adding a candidate bit-width of 6 (CADyQ, **(S5b)**) to the candidate bit-widths $\{4, 8\}$ **(S4a)** leads to higher restoration performance but also higher computational complexity. On the other hand, adding a candidate bit-width of 2 **(S5c)** to the current candidate bit-widths $\{4, 6, 8\}$ results in lower computational complexity but also lower restoration performance.

Table S4: Ablation study on variation of quantization bit-width candidates. Evaluation is conducted by utilizing CARN as a backbone, on Urban100

	Candidate bit-widths	FQR$_\downarrow$	PSNR$_\uparrow$	SSIM$_\uparrow$
(S5a)	{4, 8}	5.02	25.90	0.779
(S5b)	{4, 6, 8}	5.32	25.94	0.780
(S5c)	{2, 4, 6, 8}	3.45	25.61	0.770

B.3 Ablation on Different Inference Patch Size

Larger patches have overall similar average magnitude of image gradient (*i.e.*, smaller variance). As CADyQ is conditioned on the average gradient magnitude of each patch, smaller variance prevents CADyQ from allocating distinct bit-widths. Thus, BitOPs are less reduced for large patches, though still low compared to the existing methods, as in Table S5. By contrast, smaller patches further reduce BitOPs and maintain comparable PSNR.

Table S5: Ablation on patch size on CARN-CADyQ, Urban100

Patch size	Variance of $\|\nabla I_i\|$	PSNR (dB)	BitOPs (G)
192×192	1.41	25.95	3.36
96×96	1.79	25.94	3.23
48×48	2.53	25.92	3.12

B.4 Details on w_{reg}

w_{reg} is the hyperparameter to control the trade-off between accuracy and efficiency. We empirically chose the scalar value of w_{reg} with the DIV2K validation PSNR, as in Fig. S2. A larger w_{reg} gives further efficient network but at the cost of performance degradation. Total BitOPs are manually controlled by w_{reg}, and how to allocate total BitOPs to each layer is controlled by L_{wb} based on each layer's impact on overall performance.

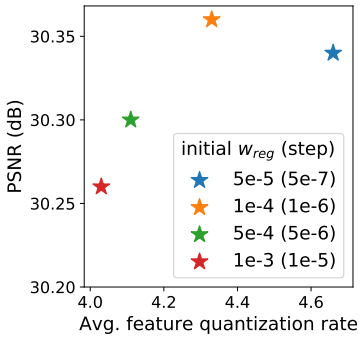


Fig. S2: w_{reg} of CARN-CADyQ

C Implementation Details

Training is done with DIV2K dataset, which consists of eight hundred 2K-resolution images (index 0001-0800). Hundred images are selected (index 0801-0900) from the DIV2K validation set for validation. The proposed framework is optimized using ADAM optimizer [51] with β_1 = 0.9, β_2 = 0.999, and ϵ = 10^{-8}. The mini-batch size is 16, and the input patch size during training is 48×48. The other training settings, such as the learning rate schedule, number of iterations, and data augmentation strategies, follow the settings of each baseline model. The initial learning rate is set as 0.01 for the learnable scale parameter a while the learning rate for the bit selector is the same as the learning rate for the SR network. Table S6 covers the detailed settings for the implementation

of the model. Specifically, each epoch consists of 100 iterations, and the bit loss weight is initialized and then increased with *step* amount for every epoch. Also, the training data is augmented with random horizontal flips and 90° rotations, and all the images are pre-processed by subtracting the mean RGB value of the DIV2K dataset.

Table S6: Implementation details of CADyQ applied SR networks

	IDN -CADyQ	EDSR-baseline -CADyQ	SRResNet -CADyQ	CARN -CADyQ
Epochs	300	300	300	600
Initial learning rate	10^{-4}	10^{-4}	10^{-4}	10^{-4}
Learning rate scheduling (γ, decay)	0.5 at 150	0.5 at 150	0.5 at 150	0.5 at 400
Bit selector initial learning rate	10^{-4}	10^{-4}	10^{-4}	10^{-3}
Bit selector learning rate scheduling (γ, decay)	0.5 at 150	0.5 at 150	0.5 at 150	0.5 at 400
Bit loss initial weight	10^{-4}	10^{-4}	10^{-4}	10^{-4}
Bit loss weight step	10^{-6}	10^{-6}	10^{-6}	10^{-6}

D Complexity Analysis

D.1 Additional Complexity Analysis

In addition to the CARN backbone models analyzed in the main manuscript, this section further analyzes the computational complexity of our framework applied to various SR networks such as IDN, EDSR-baseline, and SRResNet. The computational complexity is measured with BitOPs, which denotes the number of operations weighted with the bit-width of the operands, and estimated energy. Conventionally, BitOPs of a quantized convolution layer with weight $\boldsymbol{w} \in \mathbb{R}^{C \times C_{out} \times F \times F}$ of b_w-bit and input feature $\boldsymbol{x} \in \mathbb{R}^{N \times C \times H \times W}$ of b-bit, is calculated as $\frac{b_w}{32} \cdot \frac{b}{32} \cdot 2CC_{out}F^2NHW$. However, since the bit-width of the feature varies for each patch in our framework, the BitOPs of a quantized convolution layer is calculated as $\sum_i^N \frac{b_w}{32} \cdot \frac{b_i}{32} \cdot 2CC_{out}F^2NHW$, where b_i denotes the bit-width of i-th patch feature. As stated in Section 4.5 of the main manuscript, our framework can process either the full input test image at once or process smaller patches in parallel, which are combined to construct the full image. For both cases, our framework effectively reduces the computational resources while preserving the performance, as shown in Table S7.

D.2 Overhead Analysis

There exists computational overhead of our framework, which is from 1) overlapping patch-wise inference and 2) additional bit selector. Generally, the overhead of patch-wise inference can be reduced with parallel processors as [28] and the overhead of overlapping patches is minimized by using small overlap regions (6

Table S7: Complexity analysis. Computational costs of image-wise and patch-wise inference are respectively analyzed for models with IDN, EDSR-baseline, SRResNet backbone. Computational complexity is measured w.r.t. BitOPs of the feature extraction stage required for generating a 720p (1820 × 720) image. PSNR is measured on Urban100

Model	Params.	Inference Patch Size			
		Full Image		96×96	
		PSNR$_\uparrow$	BitOPs$_\downarrow$	PSNR$_\uparrow$	BitOPs$_\downarrow$
IDN	590.9K	25.42	59.12G	25.42	60.23G
IDN-PAMS	590.9K	25.56	3.70G	25.56	3.76G
IDN-CADyQ	594.9K	25.66	2.70G	25.65	2.70G
EDSR-baseline	1517.6K	26.04	135.90G	26.04	138.45G
EDSR-baseline-PAMS	1517.6K	25.94	8.49G	25.94	8.65G
EDSR-baseline-CADyQ	1520.9K	25.94	6.55G	25.94	6.59G
SRResNet	1546.8K	25.74	136.13G	25.74	138.68G
SRResNet-PAMS	1546.8K	25.85	8.51G	25.85	8.67G
SRResNet-CADyQ	1555.4K	25.92	6.22G	25.92	6.21G

pixels from the boundary, in our case). Nevertheless, as shown in Table S8, overheads of patch-wise inference and bit selector are non-trivial. However, CADyQ manages to overcome these overheads and successfully reduces latency/BitOPs, owing to its dynamic bit allocation.

Table S8: Overhead analysis for 4K image on CARN

	Inference Type	Bit Selector	FQR	Latency (diff w/ (a))	BitOPs (diff w/ (a))
(a) PAMS	Image	✗	8.0	216.0 ms	43.0 G
(b) PAMS (patch)	Patch	✗	8.0	230.8 ms (+14.8)	43.8 G (+0.8)
(c) PAMS (patch+bit selector)†	Patch	✓	8.0	237.3 ms (+20.9)	45.7 G (+2.7)
(d) CADyQ	Patch	✓	4.5	202.9 ms (-13.4)	29.2 G (-13.8)

† (c) includes bit selector module, but unlike CADyQ, bit selector is forced to select 8-bit.

E Qualitative Results

Fig. S3: Qualitative results from models of IDN, EDSR-baseline, and SRResNet backbone

License of the Used Assets

- DIV2K dataset [1] is made available for academic research purposes.
- Urban100 dataset [19] is made available at https://github.com/jbhuang0604/SelfExSR
- Test2K and Test4K dataset [28] is made available at https://github.com/Xiangtaokong/ClassSR

References

51. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015) 5
52. Krishnamoorthi, R.: Quantizing deep convolutional networks for efficient inference: A whitepaper. arXiv preprint arXiv:1806.08342 (2018) 1, 2