DAQ: Channel-Wise Distribution-Aware Quantization for Deep Image Super-Resolution Networks
– *Supplementary Document* –

Cheeun Hong* Heewon Kim* Sungyong Baik Junghun Oh Kyoung Mu Lee
Department of ECE, ASRI, Seoul National University
{cheeun914, ghimhw, dsybaik, dh6dh, kyoungmu}@snu.ac.kr

A. Implementation details on computation cost

In the main paper, computational resources (BOPs and estimated energy consumption) are measured with respect to quantization bit-width, considering the overflow of low-bit arithmetic operations. Overflow occurs when an arithmetic operation attempts to create a numeric value that is outside the range that can be possibly represented. Taking integer overflow into account is especially essential in low-bit networks, since the output ranges of low-bit multiplication and addition are strictly limited. Various techniques are used to avoid the integer overflow, such as using the overflow checker or value sanity testing. For ultra-low precision operations where the integer overflow is highly likely to occur, we design an appropriate large bit-width for each operation, under the assumption of an integer overflow. For instance, when the sum is accumulated over vector of size C with each n-bit element, the output buffer is $n{+}log_2(C{-}1)$-bit. Also, the output buffer for multiplication of two n-bit elements is $(2n{-}1)$-bit.

B. Derivation of convolution operations for DAQ

Section 3.3 in the main paper claims that quantization can step forward to hardware efficiency simply by postponing the de-transformation process as late as possible (See Equation (B), (D), and (G)). As more operations are done before de-transformation, in other words, in the state of real integer, the more efficient the quantization becomes.

Section 3.2 of the main paper presents a convolution operation with a n-bit *channel-wise* quantized feature map and a n-bit layer-wise quantized weight tensor. Given a feature map $\boldsymbol{x} \in \mathbb{R}^{C\times H\times W}$, c-th channel, j, k-th element of the feature map is denoted as $x_c[j, k]$ with the indexing operator $[\cdot, \cdot]$. Given a weight $\boldsymbol{w} \in \mathbb{R}^{C\times C_{out}\times K\times K}$, c-th input channel and i, u, v-th element of a part of weight tensor $\boldsymbol{w} \in \mathbb{R}^{C\times C_{out}\times K\times K}$ is denoted as $w_c[i, u, v]$ with indexing operator $[\cdot, \cdot, \cdot]$. Then, the output response $\boldsymbol{y} \in \mathbb{R}^{C_{out}\times H\times W}$ is the output of convolution between the given feature map and the weight in a sliding window manner. The i, j, k-th element of the output response is formulated as follows:

$$y[i, j, k] = \sum_{c=1}^{C}\sum_{u=1}^{K}\sum_{v=1}^{K} x_c[u{+}j, v{+}k] \cdot w_c[i, u, v]. \tag{A}$$

For simplicity, we drop the index subscript in the following equations to denote $x_c[u{+}j, v{+}k]$ as x_c and $w_c[i, u, v]$ as w_c. From our proposed quantization method, convolution with floating-point values can be approximated with low-precision values, as follows:

$$y[i, j, k] \approx \sum_{c=1}^{C}\sum_{u=1}^{K}\sum_{v=1}^{K} x_c^q \cdot w_c^q \tag{B}$$

$$= \sum_{c=1}^{C}\sum_{u=1}^{K}\sum_{v=1}^{K} (\sigma_c s(n) \cdot \hat{x}_c^q + \mu_c) \cdot (\sigma_w s(n) \cdot \hat{w}_c^q) \tag{C}$$

*equal contribution

$$= \sigma_w s(n)^2 \sum_{c=1}^{C} \sigma_c \cdot \sum_{u=1}^{K} \sum_{v=1}^{K} \hat{x}_c^q \cdot \hat{w}_c^q + \sigma_w s(n) \sum_{c=1}^{C} \mu_c \sum_{u=1}^{K} \sum_{v=1}^{K} \hat{w}_c^q. \tag{D}$$

Likewise overview Figure 3 in the main paper, the channel-wise de-transformation (see Equation (D)) derived from the procedure with element-wise de-transformation (see Equation (B)) can reduce costly operations with floating-point values. Although the computation costly operation of element-wise de-transformation is largely reduced in Equation (D), it still bears computational overhead, by operating the channel-wise summation in floating-point values (due to floating-point de-transformation parameters μ_c and σ_c). To alleviate this issue, main paper presents a scheme of quantizing quantization transformation parameters of $\boldsymbol{\mu} \in \mathbb{R}^C$ and $\boldsymbol{\sigma} \in \mathbb{R}^C$, to approximate μ_c and σ_c by using the distribution statistics of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$. From Equation 4, 5, 6 of the main paper, Equation (D) can be approximated as follows:

$$\approx \sigma_w s(n)^2 \sum_{c=1}^{C} \sigma_c^q \cdot \sum_{u=1}^{K} \sum_{v=1}^{K} \hat{x}_c^q \cdot \hat{w}_c^q + \sigma_w s(n) \sum_{c=1}^{C} \mu_c^q \sum_{u=1}^{K} \sum_{v=1}^{K} \hat{w}_c^q \tag{E}$$

$$= \sigma_w s(n)^2 \sum_{c=1}^{C} (\sigma_\sigma s(m) \cdot \hat{\sigma}_c^q + \mu_\sigma) \cdot \sum_{u=1}^{K} \sum_{v=1}^{K} \hat{x}_c^q \cdot \hat{w}_c^q + \sigma_w s(n) \sum_{c=1}^{C} (\sigma_\mu s(m) \cdot \hat{\mu}_c^q + \mu_\mu) \sum_{u=1}^{K} \sum_{v=1}^{K} \hat{w}_c^q \tag{F}$$

$$\begin{aligned} &= \sigma_w s(n)^2 \sigma_\sigma s(m) \sum_{c=1}^{C} \hat{\sigma}_c^q \sum_{u=1}^{K} \sum_{v=1}^{K} \hat{x}_c^q \cdot \hat{w}_c^q + \sigma_w s(n)^2 \mu_\sigma \sum_{c=1}^{C} \sum_{u=1}^{K} \sum_{v=1}^{K} \hat{x}_c^q \cdot \hat{w}_c^q \\ &\quad + \sigma_w s(n) \sigma_\mu s(m) \sum_{c=1}^{C} \hat{\mu}_c^q \sum_{u=1}^{K} \sum_{v=1}^{K} \hat{w}_c^q + \sigma_w s(n) \mu_\mu \sum_{c=1}^{C} \sum_{u=1}^{K} \sum_{v=1}^{K} \hat{w}_c^q. \end{aligned} \tag{G}$$

The floating-point channel-wise summation (see Equation (D)) is replaced with lower-precision channel-wise summation (see Equation (G)). As shown in Table A, simply changing the operation order from Equation (A) to Equation (D) reduces the BOPs largely from 174T to 10T. Furthermore, **q**uantizing **q**uantization transformation parameters (QQ) in Equation (D) results in Equation (G), which further reduces the BOPs to 3T with 4-bit for QQ.

Table A: Computational cost comparison of a 2-bit (w2a2) channel-wise quantized convolution. C=C_{out}=256, K=3, (H, W)=(480, 270)

De-transformation		Eqn.	Number of (n-bit, m-bit) operations								BOPs
Type	QQ		(2, 2)	(3, 3)	(6, 4)	(6, 6)	(9, 9)	(14, 32)	(17, 32)	(32, 32)	
Element-wise	✗	Eqn. (B)	-	-	-	-	-	-	-	169937M	174015G
Channel-wise	✗	Eqn. (D)	76441M	67948M	-	8493M	-	8493M	-	8493M	10108G
Channel-wise	✓	Eqn. (G)	152882M	135895M	8493M	8493M	8460M	33M	33M	66M	3046G

C. Additional experiments

C.1. Comparison with SotA methods

Existing state-of-the-art quantized SR networks [23, 41] involve a specialized architecture or an ad-hoc training scheme for low precision SR networks, mostly concentrated on binary precision. BTM [23] exploits a new training scheme like knowledge distillation and specialized gradient update rule instead of the traditional straight-through estimator [46]. BAM [41] designs a new binarized SR network, namely BSRN, utilizing a bit accumulation module.

Our proposed quantization method is orthogonal to these techniques. EDSR-BTM and BSRN-BAM are re-implemented according to the respective paper, and we replace the typical quantization (binarization) function with our distribution-aware channel-wise quantization (DAQ) function. The re-implemented architectures and the DAQ-applied architectures are respectively trained with batch size 4 and other settings same as the baseline in each paper. Despite the channel-wise overhead, our proposed method DAQ gives clear auxiliary gain in performance, for about 0.3 dB in Set5, as shown in Table B.

Table B: Comparisons on existing low-precision SR networks, EDSR-BTM and BSRN-BAM of scale 4.

Method	Precision		BOPs	Energy	Parameters	PSNR (dB)			
	w	a				Set5	Set14	B100	Urban100
EDSR-BTM [23]	1	1	23.1 T	52.8 mJ	43.1M	31.30	28.05	27.22	25.08
EDSR-BTM [23] - DAQ	1	1	75.1 T	138.9 mJ	43.1M	31.60	28.19	27.34	25.30
BSRN-BAM [41]	1	1	2.9 T	8.5 mJ	1.2M	31.17	27.94	27.15	25.01
BSRN-BAM [41] - DAQ	1	1	7.2 T	23.7 mJ	1.2M	31.44	28.03	27.21	25.05

C.2. SR Networks with batch normalization layers

In the main paper, we made a comparison with quantization methods without retraining. Among the compared methods, DFQ [37] utilizes batch normalization (BN) parameters to further improve the quantization accuracy. However, the comparison backbone of Table 3, EDSR [31] removed the BN layers for improved performance, followed by several other SR networks. For further fair comparison with DFQ, we compare the quantization methods without retraining, on EDSR *with BN*. Table C shows that our method outperforms DFQ regardless of BN layers.

Table C: Comparison of quantization methods *without retraining* on pre-trained EDSR *with BN* of scale 4.

Method	Precision		BOPs (HD image)	Energy (HD image)	PSNR (Urban100)
	w	a			
EDSR w/ BN	32	32	10025.8 T	22516.0 mJ	26.04 dB
EDSR w/ BN - LinQ	4	4	357.8 T	378.4 mJ	22.79 dB
EDSR w/ BN - DFQ	4	4	364.3 T	390.1 mJ	23.07 dB
EDSR w/ BN - DAQ	2	2	213.4 T	333.5 mJ	**24.53 dB**