Supplementary Material for Solving Oscillation Problem in Post-Training Quantization Through a Theoretical Perspective

1. Proof of Theorem 1

Theorem 1. *Given a pre-trained model and input data. If two adjacent modules are equivalent, we have,*

$$\mathcal{L}(W_i, X_i) \leq \mathcal{L}(W_{i+1}, X_{i+1}). \tag{1}$$

Proof. When two adjacent equivalent modules contain only one convolutional layer, that is,

$$\begin{aligned} & f_{i+1}^{(n_{i+1})}(W_i, X_i) \\ =& f_i^{(n_i)}(W_i, X_i) \\ =& f_i^{(1)}(W_i, X_i) \\ =& W_i X_i. \end{aligned} \tag{2}$$

Where we transform the tensor convolution into a matrix multiplication for simplicity. This transform is common in the practical implementation of convolution and is usually accompanied by the General Matrix Multiplication (GEMM) for the practical speedup [1]. Therefore,

$$\begin{aligned} & \mathcal{L}(W_{i+1}, X_{i+1}) \\ =& \mathbb{E}\left[\|W_{i+1}X_{i+1} - \widetilde{W}_{i+1}\widetilde{X}_{i+1}\|_F^2\right] \\ =& \mathbb{E}\left[\|W_{i+1}X_{i+1} - (W_{i+1} + \Delta W_{i+1})(X_{i+1} + \Delta X_{i+1})\|_F^2\right] \\ =& \mathbb{E}\left[\|W_{i+1}\Delta X_{i+1} + \Delta W_{i+1}X_{i+1} + \Delta W_{i+1}\Delta X_{i+1}\|_F^2\right] \\ \approx& \mathbb{E}\left[\|W_{i+1}\Delta X_{i+1} + \Delta W_{i+1}X_{i+1}\|_F^2\right], \end{aligned} \tag{3}$$

where ΔW and ΔX are the quantization errors of W and X. We ignore the higher order term $\Delta W \Delta X$ of the quantization error. Then,

$$\begin{aligned} & \mathbb{E}\left[\|W_{i+1}\Delta X_{i+1} + \Delta W_{i+1}X_{i+1}\|_F^2\right] \\ =& \mathbb{E}\left[\sum_m \sum_n \left(w_{i+1}^{(m)}\Delta x_{i+1}^{(n)} + \Delta w_{i+1}^{(m)} x_{i+1}^{(n)}\right)^2\right] \\ =& \mathbb{E}\left[\sum_m \sum_n \left(w_{i+1}^{(m)}\Delta x_{i+1}^{(n)}\right)^2\right] + \mathbb{E}\left[\sum_m \sum_n \left(\Delta w_{i+1}^{(m)} x_{i+1}^{(n)}\right)^2\right] \\ & + 2\mathbb{E}\left[\sum_m \sum_n w_{i+1}^{(m)}\Delta x_{i+1}^{(n)}\Delta w_{i+1}^{(m)} x_{i+1}^{(n)}\right] \\ =& \sum_m \sum_n \mathbb{E}\left[\left(w_{i+1}^{(m)}\Delta x_{i+1}^{(n)}\right)^2\right] + \sum_m \sum_n \mathbb{E}\left[\left(\Delta w_{i+1}^{(m)} x_{i+1}^{(n)}\right)^2\right] \\ & + 2\sum_m \sum_n \mathbb{E}\left[w_{i+1}^{(m)}\Delta x_{i+1}^{(n)}\Delta w_{i+1}^{(m)} x_{i+1}^{(n)}\right], \end{aligned} \tag{4}$$

where $w_{i+1}^{(m)}$ is the m-th row vector of W, $x_{i+1}^{(n)}$ is the n-th column vector of X, and others as well. Since the pre-trained model and the input are given, $w_{i+1}^{(m)}$ and $x_{i+1}^{(n)}$ are constant vectors in the PTQ optimization process, so that,

$$\begin{aligned} \mathbb{E}\left[\left(w_{i+1}^{(m)}\Delta x_{i+1}^{(n)}\right)^2\right] &= \left(w_{i+1}^{(m)}\mathbb{E}\left[\Delta x_{i+1}^{(n)}\right]\right)^2, \\ \mathbb{E}\left[\left(\Delta w_{i+1}^{(m)} x_{i+1}^{(n)}\right)^2\right] &= \left(\mathbb{E}\left[\Delta w_{i+1}^{(m)}\right] x_{i+1}^{(n)}\right)^2, \\ \mathbb{E}\left[w_{i+1}^{(m)}\Delta x_{i+1}^{(n)}\Delta w_{i+1}^{(m)} x_{i+1}^{(n)}\right] &= w_{i+1}^{(m)}\mathbb{E}\left[\Delta x_{i+1}^{(n)}\Delta w_{i+1}^{(m)}\right] x_{i+1}^{(n)}. \end{aligned} \tag{5}$$

Since the two adjacent modules are equivalent and each module contains a batchnorm layer [3], we thus consider the full precision weights and activations of the two modules to be identically distributed, i.e.,

$$\begin{aligned} \mathbb{E}_{W_i \sim \mathcal{D}_w}\left[w_i^{(m)}\right] &= \mathbb{E}_{W_{i+1} \sim \mathcal{D}_w}\left[w_{i+1}^{(m)}\right], \\ \mathbb{E}_{X_i \sim \mathcal{D}_x}\left[x_i^{(n)}\right] &= \mathbb{E}_{X_{i+1} \sim \mathcal{D}_x}\left[x_{i+1}^{(n)}\right]. \end{aligned} \tag{6}$$

Meanwhile, due to the accumulative effect of quantification errors, we have,

$$\mathbb{E}\left[\Delta x_{i+1}^{(n)}\right] > \mathbb{E}\left[\Delta x_{i}^{(n)}\right], \mathbb{E}\left[\Delta w_{i+1}^{(m)}\right] > \mathbb{E}\left[\Delta w_{i}^{(m)}\right]. \tag{7}$$

Therefore, Eq. 1 holds when the two modules contain one convolutional layer. Subsequently, suppose that Eq. 1 holds when the module contains n convolutional layers, we have,

$$\begin{aligned}&\mathbb{E}\left[\left\|f_{i+1}^{(n)}(W_{i+1}, X_{i+1}) - f_{i+1}^{(n)}(\widetilde{W}_{i+1}, \widetilde{X}_{i+1})\right\|_F^2\right] \\ &> \mathbb{E}\left[\left\|f_{i}^{(n)}(W_{i}, X_{i}) - f_{i}^{(n)}(\widetilde{W}_{i}, \widetilde{X}_{i})\right\|_F^2\right].\end{aligned} \tag{8}$$

We set,

$$\begin{aligned}f_{i+1}^{(n)}(W_{i+1}, X_{i+1}) &= X_{i+1}^{(n+1)}, \\ f_{i+1}^{(n)}(\widetilde{W}_{i+1}, \widetilde{X}_{i+1}) &= \widetilde{X}_{i+1}^{(n+1)}.\end{aligned} \tag{9}$$

Our supposition is equivalently converted to,

$$\begin{aligned}\mathbb{E}\left[\left\|X_{i+1}^{(n+1)} - \widetilde{X}_{i+1}^{(n+1)}\right\|_F^2\right] &> \mathbb{E}\left[\left\|X_{i}^{(n+1)} - \widetilde{X}_{i}^{(n+1)}\right\|_F^2\right] \\ \mathbb{E}\left[\left\|\Delta X_{i+1}^{(n+1)}\right\|_F^2\right] &> \mathbb{E}\left[\left\|\Delta X_{i}^{(n+1)}\right\|_F^2\right].\end{aligned} \tag{10}$$

When the module contains $n+1$ convolutional layers,

$$\begin{aligned}&\mathbb{E}\left[\left\|f_{i+1}^{(n+1)}(W_{i+1}, X_{i+1}) - f_{i+1}^{(n+1)}(\widetilde{W}_{i+1}, \widetilde{X}_{i+1})\right\|_F^2\right] \\ =&\mathbb{E}\left[\left\|W_{i+1}^{(n+1)} f_{i+1}^{(n)}(W_{i+1}, X_{i+1}) - \widetilde{W}_{i+1}^{(n+1)} f_{i+1}^{(n)}(\widetilde{W}_{i+1}, \widetilde{X}_{i+1})\right\|_F^2\right] \\ =&\mathbb{E}\left[\left\|W_{i+1}^{(n+1)} X_{i+1}^{(n+1)} - \widetilde{W}_{i+1}^{(n+1)} \widetilde{X}_{i+1}^{(n+1)}\right\|_F^2\right].\end{aligned} \tag{11}$$

Similar to the derivation of Eqs. 4-7, we can obtain that Eq. 1 also holds when the module contains $n+1$ convolutional layers. Therefore, by inductive reasoning, the Theorem 1 is proved.

□

2. Proof of Corollary 1

Corollary 1. *Suppose two adjacent modules be topologically homogeneous. If the module capacity of the later module is large enough, the loss will decrease. Conversely, if the latter module capacity is smaller than the former, then the accumulation effect of quantization error is exacerbated.*

Proof. We consider an extreme case, where the module is equivalent to no quantization if the bit-width b_i in the ModCap is taken to be 32 bits and the activation value is not quantized. In this case, the quantization error of this module is 0. Therefore, when the module capacity is large enough, the quantization error of the module will converge to 0, i.e.,

$$\lim_{ModCap \to m} \Delta W = \lim_{ModCap \to m} \Delta X = 0, \tag{12}$$

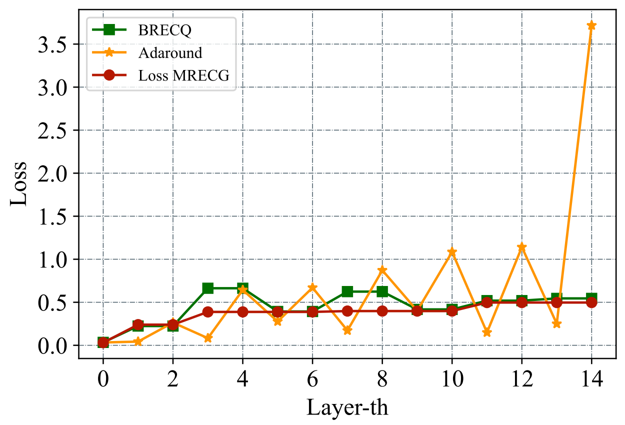

Figure 1. Loss distribution of different algorithms on ResNet-18. We ignore the reconstruction loss of the final block and the fully connected layer. Convolutional layers within a block of the block reconstruction algorithm share the reconstruction loss of that block.

where m is the module capacity of the full precision module. $\Delta W, \Delta X$ is the quantization error of the module weights and activation. Therefore, for arbitrary $\epsilon > 0$, there exists a module capacity n such that $\mathcal{L}(W_{i+1}, X_{i+1}) < \epsilon$ when $ModCap > n$. Since ϵ is arbitrary, we take $\epsilon = \epsilon_1 < \mathcal{L}(W_i, X_i)$. Consequently, we have

$$\mathcal{L}(W_{i+1}, X_{i+1}) < \epsilon_1 < \mathcal{L}(W_i, X_i). \tag{13}$$

Conversely, by Theorem 1, when the capacity of the later module is the same as the earlier one, due to the accumulative effect of the quantization error, we have

$$\mathcal{L}(W_i, X_i) \leq \mathcal{L}(W_{i+1}, X_{i+1}). \tag{14}$$

If the capacity of the later module is smaller than the earlier one, the quantization error of the module will increase, i.e.,

$$\lim_{ModCap \to 0} \Delta W = \lim_{ModCap \to 0} \Delta X = \zeta, \tag{15}$$

where ζ is the upper bound on the quantization error of the module. Therefore, for arbitrary $\epsilon > 0$, there exists a module capacity n such that $\mathcal{L}^{'}(W_{i+1}, X_{i+1}) > \zeta - \epsilon$ when $ModCap < n$. Since ϵ is arbitrary, we take $\epsilon = \epsilon_1 = \zeta - \mathcal{L}(W_{i+1}, X_{i+1})$. Consequently, we have

$$\mathcal{L}^{'}(W_{i+1}, X_{i+1}) > \zeta - \epsilon_1 = \mathcal{L}(W_{i+1}, X_{i+1}) > \mathcal{L}(W_i, X_i). \tag{16}$$

Therefore, the corollary is proved.

□

3. Loss distribution of ResNet-18

In Fig. 1 we present the reconstruction loss distribution of ResNet-18 quantized to 4/4 bit. AdaRound shows the

(a) ResNet-18

Methods	Searching Time	Top-1 Acc(%)
BRECQ	**0**	64.83
QDROP	**0**	65.56
ModCap MRECG	**0**	66.07
Loss MRECG	19.1 mins	**66.30**

(b) ResNet-50

Methods	Searching Time	Top-1 Acc(%)
BRECQ	**0**	70.06
QDROP	**0**	71.07
ModCap MRECG	**0**	71.65
Loss MRECG	60.9 mins	**71.92**

(c) MobileNetV2×0.5

Methods	Searching Time	Top-1 Acc(%)
BRECQ	**0**	29.79
QDROP	**0**	35.14
ModCap MRECG	**0**	38.02
Loss MRECG	39.5 mins	**38.43**

Table 1. Searching time of reconstruction granularity and Top-1 accuracy for different algorithms on ResNet-18, ResNet-50 and MobileNetV2. All weights and activations are quantized to 3 bit.

most drastic loss oscillations. As a result, this causes a sharp increase in reconstruction error due to irretrievable information loss. The block reconstruction strategy of BRECQ mitigates the oscillations in AdaRound, but still shows small oscillations within some layers. For example, oscillations occur between the 4-th layer and 5-th layer. MRECG completely smoothes out the oscillations, allowing the reconstruction loss of ResNet-18 to be smoothly incremented by the accumulation of quantization errors.

4. MRECG searching time

The mixed reconstruction granularity based on the loss metric requires a small portion of the data for model reconstruction to obtain the loss distribution. As shown in Tab. 1, the searching time of Loss MRECG on ResNet-18, ResNet-50 and MobileNetV2 are 19.1 mins, 60.9 mins and 39.5 mins, respectively. Loss MRECG achieves the global optimum with a small time overhead. ModCap MRECG does not require the PTQ reconstruction process and thus it is more efficient. Meanwhile, the locally optimal ModCap MRECG also outperforms the previous SOTA method and has a small performance degradation compared to the global optimum.

Methods	W/A	Reg600M
Full Prec.	32/32	73.52
ZeroQ [2]	4/4	28.54
LAPQ [6]	4/4	57.71
AdaRound [5]	4/4	68.20
BRECQ* [4]	4/4	70.44
QDROP [9]	4/4	70.62
Ours+QDROP	4/4	**71.22** **(+0.60)**
LAPQ [6]	2/4	0.17
AdaRound [5]	2/4	57.00
BRECQ* [4]	2/4	61.77
QDROP [9]	2/4	63.10
Ours+QDROP	2/4	**65.16** **(+2.06)**
AdaRound* [5]	3/3	58.29
BRECQ* [4]	3/3	62.61
QDROP [9]	3/3	64.53
Ours+QDROP	3/3	**66.08** **(+1.55)**
BRECQ* [4]	2/2	28.89
QDROP [9]	2/2	38.90
Ours+QDROP	2/2	**43.67** **(+4.77)**

Table 2. A comparison of RegNet with the State-Of-The-Art method. "*" indicates that we reproduce the algorithm in a uniform experimental setup based on open-source code. W/A represents the weights and activations bit width, respectively. Under different bit configurations, we show the comparison of our algorithm with a wide range of PTQ methods on RegNet.

5. Classification accuracy of RegNet

We complement the performance of MRECG on RegNet [8]. We quantize the Reg600M model to different bit configurations. As shown in Tab. 2, MRECG achieves optimality for different bit configurations. Specifically, we obtain 65.16% Top-1 accuracy on Reg600M with 2/4 bit, which is 2.06% higher than QDROP.

6. Implementation details

For the hyper-parameters of the reconstruction, we keep the same as in the previous approaches [4, 9]. Specifically, the number of reconstruction iterations in each module is 20,000, and we set a consistent linearly decreasing temperature b, which ranges from 20 to 2. We apply a loss ratio of 0.01 and 0.1 in ResNet and MobileNetV2, respectively, to balance the reconstruction loss and rounding loss. In the combination of MRECG and QDROP, we adopt 0.5 probability for each element to decide whether to quantize or retain full precision as described in QDROP. Our pre-trained models for ResNet-18, ResNet-50 and MobileNetV2 are from the PyTorch library [7]. Our different scaled MobileNetV2 are obtained by our own training. The number of joint optimization modules k is set to 2,4,7 in ResNet-18, ResNet-50 and MobileNetV2, respectively.

References

[1] Afzal Ahmad and Muhammad Adeel Pasha. Optimizing hardware accelerated general matrix-matrix multiplication for cnns on fpgas. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 67(11):2692–2696, 2020. 1

[2] Yaohui Cai, Zhewei Yao, Zhen Dong, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Zeroq: A novel zero shot quantization framework. In *Computer Vision and Pattern Recognition (CVPR)*, pages 13169–13178, 2020. 3

[3] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, pages 448–456, 2015. 1

[4] Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. {BRECQ}: Pushing the limit of post-training quantization by block reconstruction. In *International Conference on Learning Representations (ICLR)*, 2021. 3

[5] Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. In *International Conference on Machine Learning*, pages 7197–7206. PMLR, 2020. 3

[6] Yury Nahshan, Brian Chmiel, Chaim Baskin, Evgenii Zheltonozhskii, Ron Banner, Alex M Bronstein, and Avi Mendelson. Loss aware post-training quantization. *Machine Learning*, 110(11):3245–3262, 2021. 3

[7] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 3

[8] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10428–10436, 2020. 3

[9] Xiuying Wei, Ruihao Gong, Yuhang Li, Xianglong Liu, and Fengwei Yu. Qdrop: Randomly dropping quantization for extremely low-bit post-training quantization. *arXiv preprint arXiv:2203.05740*, 2022. 3