Supplementary Material of "Distribution-aware Adaptive Multi-bit Quantization"

Sijie Zhao[1,2], Tao Yue[1], Xuemei Hu[1]
[1]School of Electronic Science and Engineering, Nanjing University, Nanjing, China
[2]Shenzhen Institute of Future Media Technology, Shenzhen 518071, China
sjzhao@smail.nju.edu.cn, yuetao@nju.edu.cn, xuemeihu@nju.edu.cn

In this supplementary material, we provide additional analysis and experimental results. In Sec. 1, we present the details for optimizing the expected mean square error under assumption of Gaussian distribution. In Sec. 2, we show the details of the brute force search algorithm. In Sec. 3, we explain the reason for choosing the Laplace distribution in the main paper. In Sec. 4, we analysis the theoretical computation cost of the proposed DMBQ and LBA.

1. DMBQ for Gaussian Distribution

As illustrated in Sec. 3.1, the proposed DMBQ method can be easily migrated to other distribution type. In this section, we will depict the details for Gaussian case which is a widely used distribution. Under the Gaussian distribution, we first normalize the weights, that is rewriting Eq. (5) as,

$$\hat{\boldsymbol{W}} = \Pi_{\mathcal{Q}(\boldsymbol{\alpha})}\big(\frac{\boldsymbol{W}-\hat{\mu}}{\hat{\sigma}}\big)\cdot\hat{\sigma}+\hat{\mu}, \tag{1}$$

where $\hat{\mu}=\mathbb{E}(\boldsymbol{W})$ and $\hat{\sigma}^2=\mathbb{E}\big((\boldsymbol{W}-\hat{\mu})^2\big)$. Then we only consider the standard case of Gaussian, i.e. $\mathcal{N}(0,1)$. We take $f(x)=\frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}}$ into Eq. (4) and we can get,

$$\min_{\boldsymbol{\alpha}} \sum_{i=1}^{2^{M-1}} 2\Big(\Psi\big(q_i(\boldsymbol{\alpha}),s_{i+1}(\boldsymbol{\alpha})\big)-\Psi\big(q_i(\boldsymbol{\alpha}),s_i(\boldsymbol{\alpha})\big)\Big),$$
$$\Psi(q,s)=\frac{(2q-s)}{\sqrt{2\pi}}e^{-\frac{s^2}{2}}+\frac{1+q^2}{2}erf(\frac{s}{\sqrt{2}}), \tag{2}$$

where $\Psi(q,s)$ is the primitive function of $f(x)(x-q)^2$ in the negative x-axis and $erf(\cdot)$ is Gauss Error Function. $q_i(\boldsymbol{\alpha})$ and $s_i(\boldsymbol{\alpha})$ are not continuous w.r.t $\boldsymbol{\alpha}$ which can not be derived as close form solution, thus we apply the same brute force searching.

2. The Brute Force Searching Algorithm

As introduced in Sec. 3.1, we use brute force searching to find the optimal coordinates for MBQ under a certain distribution. Here we present the details of this algorithm.

We generate the search space for coordinate $\boldsymbol{\alpha}$ to calculate the quantization levels $\mathcal{Q}(\boldsymbol{\alpha})$, i.e., α_i $(i=1,2,...,M)$ from 0 to 10 with resolution of 0.001. Through computing the objective (Eq. (6)) with respect to different $\boldsymbol{\alpha}$, we could find the optimal $\boldsymbol{\alpha}^{\text{best}}$ which corresponds to the minimal quantization error. The details of the brute force searching algorithm are concluded in Alg. 1.

Algorithm 1: Brute Force Searching for Optimal $\boldsymbol{\alpha}$

```
Input: bit-width $M$
Output: $\boldsymbol{\alpha}^{\text{best}}$
Generate search space for $\alpha_i$ $(i=1,2,...,M)$, i.e.
  from 0 to 10 with resolution of 0.001.
Initialize minimal expected mean square error
  $e^{\min}=\infty$.
for $\boldsymbol{\alpha}$ in search space do
    Calculate the quantization levels $\mathcal{Q}(\boldsymbol{\alpha})$.
    Calculate the rounding edges with Eq. (3).
    Calculate the expected mean square error $e$ with
      Eq. (6).
    if $e < e^{\min}$ then
        $e^{\min} = e$.
        $\boldsymbol{\alpha}^{\text{best}} = \boldsymbol{\alpha}$.
    end
end
```

With the brute force searching, in Laplace case, we can get the optimal $\boldsymbol{\alpha}^{\text{best}}$ = [1.0], [1.009,1.591], [0.832,1.514,1.897], [0.838,1.324,1.619,1.879] for M = 1, 2, 3, 4 respectively, in Gaussian case, we can get the optimal $\boldsymbol{\alpha}^{\text{best}}$ = [0.8], [0.521,0.981], [0.445,0.748,0.985], [0.3,0.547,0.71,1.12]. Then the $\boldsymbol{\alpha}^{\text{best}}$ can be used to construct lookup table for $\mathcal{Q}(\boldsymbol{\alpha})$ w.r.t bit-width for once.

3. Distribution Problem

The most weights of neural network distributed in a central symmetric bell-like shape, which is usually modeled by the Laplace and Gaussian distributions [1]. In our study,

Table 1. Comparison between Laplace and Gaussian distribution (ResNet20 on CIFAR10 Dataset).

Assumption	Prec (W/A)	Top-1
Gaussian	2/2	89.6
	3/3	92.5
	4/4	92.8
Laplace	2/2	90.7
	3/3	92.5
	4/4	93.0

Table 2. Computational cost analysis of the proposed DMBQ and LBA (ResNet18 on ILSVRC12).

Method	Prec (W/A)	FixOPS	Top-1
FP	32/32	1.81G	70.3
DMBQ	2/2	224M	65.1
	3/3	357M	69.2
	4/4	542M	70.2
DMBQ+LBA	2.0/2.0	298M	67.8
	3.0/3.0	413M	70.0

we test several widely used networks, e.g. ResNet, VGG, ShuffleNet-V2, SqueezeNet, DenseNet-V2, Inception-V3, and fitting the probability distribution functions of the their weights with both Laplace and Gaussian, then use KL divergence to measure the similarity between the actual distribution and fitted distribution. The result shows that up to 85.3% convolution layers are more similar (i.e. with lower KL divergence) to Laplace than Gaussian. We also make a comparison between Gaussian and Laplace. Specifically, we use the above two lookup tables for training-aware quantization with ResNet20 on CIFAR10 Dataset. The results are shown as Table 1, and the accuracy in Laplace case is always not inferior to the accuracy in Gaussian, especially in 2/2 bit. Thus we choose Laplace distribution in the main paper. However, the distribution of weights is complicated for some very few layers in which cases the choice of distribution is still an open problem.

4. Computation Cost

In this section, we analyze the theoretical computational cost of the proposed DMBQ and LBA. In order to compare the operations with different bit-width, we use FixOP computation scheme introduced by Li *et al.* [2] where one FixOP is defined as one operation between 8-bit activation and 8-bit weight which is equivalent to 64 binary operations. We implement our experiments on ResNet18 with ILSVRC12 dataset and the implementation details are included in the main paper.

As shown in Table 2, Our DMBQ with 4/4-bit quantization scheme has only 0.1% degradation on Top-1 accuracy with 3.34$\times$ acceleration, and Our DMBQ+LBA with 3.0/3.0-bit quantization scheme has only 0.3% degradation on Top-1 accuracy with 4.38$\times$ accelerate. The promising results exhibit the potential of deployment on hardwares with limited memory and computing power.

References

[1] Ron Banner, Yury Nahshan, and Daniel Soudry. Post training 4-bit quantization of convolutional networks for rapid-deployment. In *Proceedings of Advances in Neural Information Processing Systems*, pages 7950–7958, 2019.

[2] Yuhang Li, Xin Dong, and Wei Wang. Additive powers-of-two quantization: A non-uniform discretization for neural networks. In *Proceedings of International Conference on Learning Representations*, 2020.