Towards Mixed-Precision Quantization of Neural Networks via Constrained Optimization: Supplementary Material

Weihan Chen[1,2] Peisong Wang[1] Jian Cheng[1*]
[1]NLPR & AIRIA, Institute of Automation, Chinese Academy of Sciences
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences
chenweihan2018@ia.ac.cn, {peisong.wang, jcheng}@nlpr.ia.ac.cn

1. Experimental Setup

In this section, we introduce the implementation details of our experiments. First, we quantize standard ResNet-18, ResNet-50 [4], and MobileNet-V2 [6] architectures to justify the proposed method. Unless explicit mention of the contrary, these models are pre-trained on ImageNet dataset [2] and taken from the Pytorch model zoo.

As for mixed-precision quantization, the candidate bit-widths of weight are $\{1, 2, 3, 4, 5, 6, 7, 8\}$ for each architecture by default. We quantize the activation of the first and last layer with 8 bit, and the intermediate layers are quantized uniformly with lower bit-width.

To compensate for the accuracy degradation, we will perform fine-tuning through the standard Straight-Through Estimator (STE) [1] for each quantized network. Specifically, we adopt the standard data augmentation refers to [4], which resizes the image first such that the shorter side is 256 and then randomly samples a 224×224 crop from the image or its horizontal flip, with the per-channel mean subtracted. Besides, We use SGD with a momentum of 0.9 and a mini-batch size of 256. The learning rate starts from 0.005 and anneals with the cosine learning rate scheduler. As we expect that low-precision quantization will reduce the tendency to overfit, we train the models with a weight decay of $5e-5$ for 30 epochs.

2. Computational Complexity

In this section, we analyze the computational complexity of the proposed method. As shown in Algorithm 1, the computational cost of our method is mainly composed of two parts, namely the calculation of loss perturbation and the greedy-search to solve MCKP.

For the calculation of loss perturbation, we denote the number of training images as N and the computational complexity of network forward/backward propagation for a single image as T. Then its total computational complexity is $\mathcal{O}(NT)$. As shown in the Figure 1, the loss perturbation converges rapidly and a few hundred of images are enough for its approximation.

For greedy-search to solve MCKP, we denote the number of candidate bit-widths and layers as $|\mathbb{B}|$ and L, respectively. First, the computational complexity of picking the largest element from an unordered sequence is $\mathcal{O}(L)$. As the maximum number of iterations of the loop is $|\mathbb{B}|L$, its total computational complexity is $\mathcal{O}(|\mathbb{B}|L^2)$.

3. Theoretical Analysis on the Hessian Matrix Approximation

In this section, we provide a theoretical analysis of the Hessian matrix approximation. In summary, we first give the conditions under which the approximated Hessian matrix equals to the true one. Second, We give the upper bound of the difference between the approximated Hessian matrix and the true one.

Consider a supervised machine learning problem of predicting outputs $\mathbf{y} \in \mathbb{Y}$ from inputs $\mathbf{x} \in \mathbb{X}$. We assume a probabilistic model for the conditional distribution of the form $p_\theta(\mathbf{y}|\mathbf{x}) = p(\mathbf{y}|f(\mathbf{x}, \theta))$, where $p(\mathbf{y}|\cdot)$ is an exponential family

*Corresponding Author

with natural parameters in $\mathbb{F}$ and $f : \mathbb{X} \times \Omega \to \mathbb{F}$ is a prediction function parameterized by $\theta \in \Omega$. Given N i.i.d. training samples $D_N = \{\mathbf{x}^{(n)}, \mathbf{y}^{(n)}\}_{n=1}^N$, we want to minimize

$$\begin{aligned}\mathcal{L}(\theta) &= -\frac{1}{N}\sum_{n=1}^{N} \log p_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n)}) \\ &= -\frac{1}{N}\sum_{n=1}^{N} \log p(\mathbf{y}^{(n)}|f(\mathbf{x}^{(n)}, \theta)).\end{aligned} \tag{1}$$

We will analyze on the basis of this framework as it covers common scenarios such as least-square regression with fixed σ^2

$$\begin{aligned}\mathbb{Y} &= \mathbb{F} = \mathbb{R} \\ p(\mathbf{y}|f) &= N(\mathbf{y}; f, \sigma^2)\end{aligned} \tag{2}$$

or C-class classification

$$\begin{aligned}\mathbb{Y} &= \{1, \dots, C\}, \mathbb{F} = \mathbb{R}^C \\ p(\mathbf{y} = c|f) &= \exp(f_c)/\sum_i \exp(f_i)\end{aligned} \tag{3}$$

for an arbitrary prediction function f (e.g. CNNs). It should be noted that here $p_\theta(\mathbf{y}|\mathbf{x}^{(n)})$,instead of $f(\mathbf{x}^{(n)}, \theta)$, denotes the final output of the model, which is different from the main body of the paper. Besides, we define $p_{D^N}(\mathbf{y}|\mathbf{x}^{(n)})$ as the **data distribution** and $p_\theta(\mathbf{y}|\mathbf{x}^{(n)})$ as the **model distribution**.
Based on the above notations, we define the Hessian matrix as

$$\begin{aligned}H(\theta) &= \nabla_\theta^2 \mathcal{L}(\theta) \\ &= -\frac{1}{N}\sum_{n=1}^{N} \nabla_\theta^2 \log p_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n)}) \\ &= \frac{1}{N}\sum_{n=1}^{N} \mathbb{E}_{\mathbf{y}\sim p_{D^N}(\mathbf{y}|\mathbf{x}^{(n)})}[-\nabla_\theta^2 \log p_\theta(\mathbf{y}|\mathbf{x}^{(n)})].\end{aligned} \tag{4}$$

The Hessian matrix approximation can be rewritten as

$$\begin{aligned}\tilde{H}(\theta) &= \frac{1}{N}\sum_{n=1}^{N} \frac{1}{p_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n)})^2}[\nabla_\theta p_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n)})\, \nabla_\theta p_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n)})^T] \\ &= \frac{1}{N}\sum_{n=1}^{N} \Big[\frac{1}{p_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n)})}\nabla_\theta p_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n)})\Big]\Big[\frac{1}{p_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n)})}\nabla_\theta p_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n)})^T\Big] \\ &= \frac{1}{N}\sum_{n=1}^{N} \nabla_\theta \log p_\theta(\mathbf{y}^{(\mathbf{n})}|\mathbf{x}^{(n)})\, \nabla_\theta \log p_\theta(\mathbf{y}^{(\mathbf{n})}|\mathbf{x}^{(n)})^T \\ &= \frac{1}{N}\sum_{n=1}^{N} \mathbb{E}_{\mathbf{y}\sim p_{D^N}(\mathbf{y}|\mathbf{x}^{(n)})}[\nabla_\theta \log p_\theta(\mathbf{y}|\mathbf{x}^{(n)})\, \nabla_\theta \log p_\theta(\mathbf{y}|\mathbf{x}^{(n)})^T].\end{aligned} \tag{5}$$

Besides, we introduce the Fisher information matrix [5] as

$$\begin{aligned}F(\theta) &= \frac{1}{N}\sum_{n=1}^{N} \mathbb{E}_{\mathbf{y}\sim p_\theta(\mathbf{y}|\mathbf{x}^{(n)})}[\nabla_\theta \log p_\theta(\mathbf{y}|\mathbf{x}^{(n)})\, \nabla_\theta \log p_\theta(\mathbf{y}|\mathbf{x}^{(n)})^T] \\ &= \frac{1}{N}\sum_{n=1}^{N} \mathbb{E}_{\mathbf{y}\sim p_\theta(\mathbf{y}|\mathbf{x}^{(n)})}[-\nabla_\theta^2 \log p_\theta(\mathbf{y}|\mathbf{x}^{(n)})].\end{aligned} \tag{6}$$

The second equality of Eq. (6) may seem counterintuitive. To see why, apply the chain rule on it and then we have

$$\begin{aligned}\mathbb{E}_{\mathbf{y}\sim p_\theta(\mathbf{y}|\mathbf{x}^{(n)})}[-\nabla_\theta^2 \log p_\theta(\mathbf{y}|\mathbf{x}^{(n)})] = \mathbb{E}_{\mathbf{y}\sim p_\theta(\mathbf{y}|\mathbf{x}^{(n)})}[-\frac{1}{p_\theta(\mathbf{y}|\mathbf{x}^{(n)})}\nabla_\theta^2 p_\theta(\mathbf{y}|\mathbf{x}^{(n)})] \\ + \mathbb{E}_{\mathbf{y}\sim p_\theta(\mathbf{y}|\mathbf{x}^{(n)})}[\nabla_\theta \log p_\theta(\mathbf{y}|\mathbf{x}^{(n)})\, \nabla_\theta \log p_\theta(\mathbf{y}|\mathbf{x}^{(n)})^T].\end{aligned} \tag{7}$$

The first term on the right side of Eq. (7) is zero, since

$$\begin{aligned}\mathbb{E}_{\mathbf{y}\sim p_\theta(\mathbf{y}|\mathbf{x}^{(n)})}[-\frac{1}{p_\theta(\mathbf{y}|\mathbf{x}^{(n)})}\nabla_\theta^2 p_\theta(\mathbf{y}|\mathbf{x}^{(n)})] &= -\int \frac{1}{p_\theta(\mathbf{y}|\mathbf{x}^{(n)})}\nabla_\theta^2 p_\theta(\mathbf{y}|\mathbf{x}^{(n)}) p_\theta(\mathbf{y}|\mathbf{x}^{(n)})\mathrm{d}\mathbf{y} \\ &= \int \nabla_\theta^2 p_\theta(\mathbf{y}|\mathbf{x}^{(n)})\mathrm{d}\mathbf{y} = \nabla_\theta^2 \int p_\theta(\mathbf{y}|\mathbf{x}^{(n)})\mathrm{d}\mathbf{y} \\ &= \nabla_\theta^2[1] = 0.\end{aligned} \tag{8}$$

And the reserved second term is exactly what we expect. To sum up, we have the following three matrices:

$$H(\theta) = \frac{1}{N}\sum_{n=1}^{N} \mathbb{E}_{\mathbf{y}\sim p_{D^N}(\mathbf{y}|\mathbf{x}^{(n)})}[-\nabla_\theta^2 \log p_\theta(\mathbf{y}|\mathbf{x}^{(n)})] \tag{9}$$

$$\tilde{H}(\theta) = \frac{1}{N}\sum_{n=1}^{N} \mathbb{E}_{\mathbf{y}\sim p_{D^N}(\mathbf{y}|\mathbf{x}^{(n)})}[\nabla_\theta \log p_\theta(\mathbf{y}|\mathbf{x}^{(n)})\, \nabla_\theta \log p_\theta(\mathbf{y}|\mathbf{x}^{(n)})^T] \tag{10}$$

$$\begin{aligned}F(\theta) &= \frac{1}{N}\sum_{n=1}^{N} \mathbb{E}_{\mathbf{y}\sim p_\theta(\mathbf{y}|\mathbf{x}^{(n)})}[\nabla_\theta \log p_\theta(\mathbf{y}|\mathbf{x}^{(n)})\, \nabla_\theta \log p_\theta(\mathbf{y}|\mathbf{x}^{(n)})^T] \\ &= \frac{1}{N}\sum_{n=1}^{N} \mathbb{E}_{\mathbf{y}\sim p_\theta(\mathbf{y}|\mathbf{x}^{(n)})}[-\nabla_\theta^2 \log p_\theta(\mathbf{y}|\mathbf{x}^{(n)})]\end{aligned} \tag{11}$$

Then it's easy to get the following conclusion.

Proposition 1. *If 1) the number of observed data is large enough $N \to \infty$, and 2) the data distribution equals to the model distribution (i.e., $p_{D^N}(\mathbf{y}|\mathbf{x}^{(n)}) = p_\theta(\mathbf{y}|\mathbf{x}^{(n)})$), the Fisher equals to the approximated and true Hessian, namely*

$$\tilde{H}(\theta) = F(\theta), H(\theta) = F(\theta) \Longrightarrow \tilde{H}(\theta) = H(\theta) \tag{12}$$

As the observed data and model capacity is limited, the conditions in Proposition 1 are not strictly met usually. Furthermore, we show that the difference between the approxiamted and true Hessian can be bounded by the residuals and the smoothness constant of the model f.

Proposition 2. *Let $\mathcal{L}(\theta)$ be defined as in Eq. (1) with $\mathbb{F} = \mathbb{R}^M$. Denote by $f_m^{(n)}$ the m-th component of $f(\mathbf{x}^{(n)}, \cdot) : \Omega \to \mathbb{R}^M$ and assume each $f_m^{(n)}$ is β-smooth. Let $\tilde{H}(\theta)$ and $H(\theta)$ be the approximated and true Hessian, respectively. Then,*

$$\|H(\theta) - \tilde{H}(\theta)\|_2^2 \le r(\theta)\beta \tag{13}$$

where $r(\theta) = \sum_{n=1}^{N} \|\nabla_f \log p(\mathbf{y}^{(n)}|f(\mathbf{x}^{(n)}, \theta))\|_1$ and $\|\cdot\|_2$ denotes the spectral norm.

The approximation improves as the residuals in $r(\theta)$ diminish, and is exact if the data is perfectly fit, which is consistent with Proposition 1.

To prove the proposition, we drop θ from the notation for brevity, then the Hessian can be expressed as

$$H = \tilde{H} + \sum_{n=1}^{N}\sum_{m=1}^{M} r_m^{(n)} \nabla_\theta^2 f_m^{(n)}, \tag{14}$$

where $r_m^{(n)} = -\frac{\partial \log p_\theta(\mathbf{y}^{(n)}|f^{(n)})}{\partial f_m^{(n)}}$. If all $f_m^{(n)}$ are β-smooth, their Hessians are bounded by $-\beta\mathbf{I} \preceq \nabla_\theta^2 f_m^{(n)} \preceq \beta\mathbf{I}$ and

$$-\left|\sum_{n,m} r_m^{(n)}\right|\beta\mathbf{I} \preceq H - \tilde{H} \preceq \left|\sum_{n,m} r_m^{(n)}\right|\beta\mathbf{I} \tag{15}$$

Pulling the absolute value inside the double sum gives the upper bound

$$\left|\sum_{n,m} r_m^{(n)}\right| \leq \sum_{n,m} |r_m^{(n)}| = \sum_n \sum_m \left|\frac{\partial \log p_\theta(\mathbf{y}^{(n)}|f^{(n)})}{\partial f_m^{(n)}}\right| = \sum_{n=1}^{N} \|\nabla_f \log p(\mathbf{y}^{(n)}|f(\mathbf{x}^{(n)},\theta))\|_1 \tag{16}$$

Finally, substitute Eq (16) into Eq (15) and the proof is completed.

4. Empirical Analysis on the Hessian Matrix Approximation

As the approximation of Hessian matrix is the core of our method, except for the theoretical analysis, here we provide some additional empirical analysis. We denote the real Hessian by H_w and make a two-step approximation of it. First in eq (6), we ignore the first term in the rhs and denoted the approximation by $\tilde{H}_w$. Second in eq (12), we assume the Hessian is block-diagonal and denote the approximation by $\tilde{H}_w^{block}$. We denote the loss perturbations obtained from above Hessian as $\Delta\mathcal{L}$, $\Delta\tilde{\mathcal{L}}$ and $\Delta\tilde{\mathcal{L}}_{block}$. As the cost of analyzing Hessian matrix is too huge, we instead compare the corresponding loss perturbations. We randomly sample 100 different bitwidth allocations for ResNet18 with the target compression ratio of 8x and compute the above three loss perturbations for each one. The results are summarized in the first two subfigures of Figure 1. The x-axis and y-axis of the left and middle subfigure are $\Delta\mathcal{L}$ v.s. $\Delta\tilde{\mathcal{L}}$ and $\Delta\tilde{\mathcal{L}}$ v.s. $\Delta\tilde{\mathcal{L}}_{block}$ respectively. As these points are roughly distributed on a straight line passing through the origin with a slope of one, we justify the effectiveness of our approximation empirically.

Besides, we compare our approximation with HAWQ-V2[3]. Similarily, we randomly sample 50 bitwidth allocations and compare the difference between the true loss perturbation($\Delta\tilde{\mathcal{L}}_{true}$) and the two approximated ones(e.g. $|\frac{\Delta\mathcal{L}_{true}-\Delta\tilde{\mathcal{L}}}{\Delta\mathcal{L}_{true}}| \times 100$). The results in the right subfigure of Figure 1 show that our approximation is superior to HAWQ-V2 in the estimation of loss perturbation.

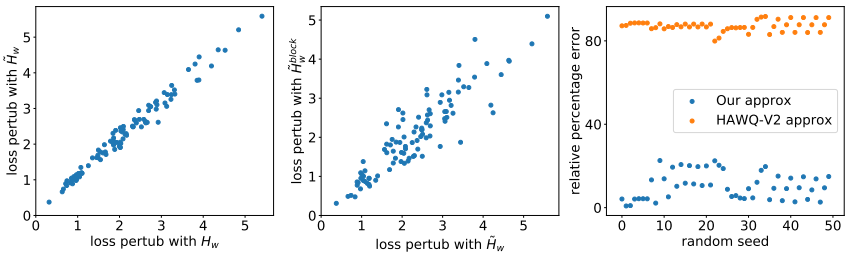

Figure 1. Analysis for Hessian Approximation

References

[1] Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *CoRR*, abs/1308.3432, 2013. 1

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255, 2009. 1

[3] Zhen Dong, Zhewei Yao, Yaohui Cai, Daiyaan Arfeen, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. HAWQ-V2: hessian aware trace-weighted quantization of neural networks. *CoRR*, abs/1911.03852, 2019. 4

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778, 2016. 1

[5] NICHOLAS T. LONGFORD. A fast scoring algorithm for maximum likelihood estimation in unbalanced mixed models with nested random effects. *Biometrika*, 74(4):817–827, 12 1987. 2

[6] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4510–4520, 2018. 1