Trace Weighted Hessian-Aware Quantization

Zhen Dong, Zhewei Yao, Daiyaan Arfeen*, Yaohui Cai*,
Amir Gholami, Michael W. Mahoney, Kurt Keutzer
University of California at Berkeley
{zhendong, zheweiy, daiyaanarfeen, yaohuic, amirgh, mahoneymw, keutzer}@berkeley.edu

Abstract

Quantization can efficiently assist the deployment of neural networks on mobile systems with constrained resources. However, directly quantizing a model to ultra low precision could cause significant accuracy degradation. Most of the works addressing this problem use first order information, along with expensive AutoML search methods to find the bit precision for different layers. Here we introduce trace weighted Hessian-aware Quantization, a new second order based method which does not require any expensive search methods. We provide theoretical results to show that the trace of the Hessian, under certain assumption, could be used to determine sensitivity of different layers to quantization, and we use this information to perform Hessian aware fine-tuning. We test our second-order approach, and show that it exceeds industry-scale results which use expensive AutoML search methods. In particular, we present quantization results on ImageNet dataset for Inception-V3 (75.68% with 7.57MB model size) and ResNet50 (75.76% with 7.99MB model size). Both results are state-of-the-art for quantized models.

1 Introduction

One of the major challenges of deploying Neural Network (NN) models on embedded systems, is their prohibitive model size, energy usage, and in some cases unacceptable latency. For example many edge devices have limited memory size and battery capacity, and large models with high power consumption cannot be deployed on them. Moreover, applications such as autonomous driving have strict limitations on the acceptable inference latency. Quantization [10, 15, 16, 11, 3, 14, 4] is a very promising approach to address these problems. Quantization reduces the memory footprint of the model parameters and activations by using reduced precision storage instead of using 32-bit floating point. This leads to smaller memory movement volume which can significantly reduce power consumption [5]. Moreover, quantization allows use of reduced precision integer arithmetic instead of floating-point arithmetic which can reduce inference latency.

Recently, mixed-precision quantization and progressive fine-tuning have been used for ultra-low precision quantization to achieve negligible accuracy drop between quantized and original models [13]. In this approach, each layer is quantized with different bit precision. However, the search space for choosing this bit setting is exponential in the number of layers. Existing approaches use expensive AutoML methods to search this exponential space. However, these search methods require large amounts of computational resources and are time-consuming.

A very promising technique to address the above problem is to use second order information. The recent method [4] use the Hessian information as a sensitivity metric to determine the mixed-precision bit setting for different layers, as well as a new fine-tuning order to automatically choose a quantization sequence. This is achieved by computing the top eigenvalue of the Hessian of the loss with respect to parameters of each block. However, as we will discuss later, just using the top Hessian eigenvalue is

* Equal Contribution
33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

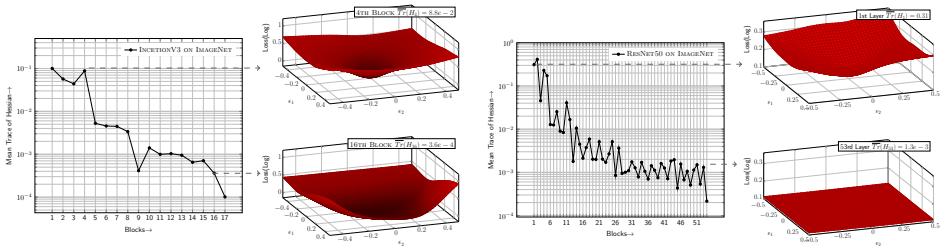

Figure 1: Mean trace of Hessian of different blocks in Inception-V3 and ResNet50 on ImageNet, along with the loss landscape of the block 4 and 16 in Inception-V3 (block 1 and 53 in ResNet50). As one can see, the mean trace of the Hessian is significantly different for different blocks. We use this information to determine the quantization precision setting, i.e. we assign higher bits for blocks with larger mean Hessian trace, and fewer bits for blocks with smaller mean Hessian trace.

not enough. One needs to consider the full Hessian spectrum and especially its trace. In particular, our contributions are as follows:

- We prove that under assumptions specified in Assumption 1, the mean Hessian trace can determine the relative sensitivity of different layers to quantization. In particular, we show that layers with smaller mean Hessian trace can achieve better loss value after fine-tuning with quantized weights as compared to layers with large mean Hessian trace.
- We compute the trace of Hessian using Huthinson algorithm, with a matrix free implementation in PyTorch. We test the proposed algorithm on various novel models on ImageNet dataset including ResNet50 and InceptionV3. We show that in all cases our new method achieves better results in both accuracy and model size, even when compared to AutoML based methods.

Outline: In § 2, we discuss theoretical results related to the mean trace of Hessian. Then we test our proposed method for various models on image classification task in § 3, followed by conclusions.

2 Methodology

For a supervised learning framework, the goal is to minimize the empirical risk loss,

$$L(\theta) = \frac{1}{N} \sum_{i=1}^{N} l(x_i, y_i, \theta), \tag{1}$$

where θ is the parameter, $l(x, y, \theta)$ is the loss for the input datum (x, y), and N is the cardinality of the training set. Here we assume that the model is already trained by solving the above optimization problem. That is we assume that all blocks of the model have converged to the local optimal point, i.e. $\nabla_{W_i} L(\theta) = 0$, [1]. The NN is partitioned into b blocks as $\{B_1, B_2, \cdots, B_b\}$, with corresponding learnable parameters $\{W_1, W_2, \cdots, W_b\}$. The goal is to quantize the parameters and activations of these blocks to lower bit precision.

2.1 Quantization and Hessian Spectrum

A Hessian based method was proposed in [4] to achieve mixed-precision quantization. However, the approach proposed there was only based on the top Hessian eigenvalue, and did not consider the rest of Hessian spectrum. In [4], a higher bit precision is used for layers with larger top Hessian eigenvalue, and vice versa. Since the direction of quantization perturbation is not necessarily the same as the direction of top eigenvector during quantization-aware fine-tuning, we show that under the assumptions outlined below, a better metric is to measure mean Hessian trace instead of just the top Hessian eigenvalue.

Assumptions 1 *Assume that:*

- *The model is twice differentiable, and has converged to a local minima such that the first and second order optimality conditions are satisfied, i.e. the gradient is zero and the Hessian is positive semi-definite.*

[1] Here, we assume that the training process terminates until the model converges.

- *Let H_i be the Hessian of i-th block and $v_1^i,\ v_2^i, ..., v_{n_i}^i$ be the orthonormal eigenvectors of H_i. Then we assume the fine-tuning perturbation, $\Delta W_i^* = \arg\min_{W_i^* + \Delta W_i^* \in Q(\cdot)} L(W_i^* + \Delta W_i^*)$, satisfies*

$$\Delta W_i^* = \alpha_{bit} v_1^i + \alpha_{bit} v_2^i + ... + \alpha_{bit} v_{n_i}^i. \tag{2}$$

  Here n_i is the dimension of W_i, W_i^* is the converging point of i-th block, $Q(\cdot)$ is the quantization function, which maps floating point values to reduced precision values, and α_{bit} is a constant number based on quantization bit setting.
- *After fine-tuning, the best solution is within the convex vicinity of the solution before quantization.*

We could prove that,

Lemma 1 *Suppose we quantize two blocks (for simplicity, assume they are B_1 and B_2) with same amount of perturbation, namely $\|\Delta W_1^*\|_2^2 = \|\Delta W_2^*\|_2^2$. Under Assumption 1, we will have*

$$L(W_1^* + \Delta W_1^*; W_2^*, \cdots, W_b^*) \leq L(W_1^*, W_2^* + \Delta W_2^*, W_3^*, \cdots, W_b^*),^2 \tag{3}$$

if

$$\frac{1}{n_1} \nabla_{W_1}^2 L(W_1^*) \leq \frac{1}{n_2} \nabla_{W_2}^2 L(W_2^*). \tag{4}$$

Proof Sketch: Denote g_1 and H_1 are the corresponding gradient and Hessian of first block. By Taylor's expansion, we have:

$$L(W_1^* + \Delta W_1^*) = L(W_1^*) + g_1^T \Delta W_1^* + \frac{1}{2} \Delta {W_1^*}^T H_1 \Delta W_1^* = L(W_1^*) + \frac{1}{2} \Delta {W_1^*}^T H_1 \Delta W_1^*.$$

Here we have used the fact that gradient at the optimum point is zero, and that the loss function is locally convex. Also note that $L(W_1^*) = L(W_2^*)$ since the model has the same loss before we quantize any block. Based on assumption, ΔW_1^* can be decomposed by the eigenvectors of the Hessian. As a result we have:

$$\Delta {W_1^*}^T H_1 \Delta W_1^* = \sum_{i=1}^{n_1} \alpha_{bit,1}^2 {v_i^1}^T H_1 v_i^1 = \alpha_{bit,1}^2 \sum_{i=1}^{n_1} \lambda_i^1,$$

where λ_i^1 is the corresponding eigenvalue of v_i^1. Similarly, for the second layer we will have: $\Delta {W_2^*}^T H_2 \Delta W_2^* = \alpha_{bit,2}^2 \sum_{i=1}^{n_2} \lambda_i^2$, where λ_i^2 is the i-th eigenvalues of H_2. Since $\|\Delta W_1^*\|_2 = \|\Delta W_2^*\|_2$, we have $\sqrt{n_1}\alpha_{bit,1} = \sqrt{n_2}\alpha_{bit,2}$. Therefore, we have:

$$L(W_2^* + \Delta W_2^*) - L(W_1^* + \Delta W_1^*) = \alpha_{bit,2}^2 n_2 (\frac{1}{n_2} \sum_{i=1}^{n_2} \lambda_i^2 - \frac{1}{n_1} \sum_{i=1}^{n_1} \lambda_i^1) \geq 0.$$

It is easy to see that the lemma holds since the sum of eigenvalues equals to the trace of the matrix. □

It is possible to compute the mean trace of the Hessian using matrix free methods to avoid the prohibitive cost of forming it. Several works [1, 2] have proposed randomized algorithm to quickly estimate the trace of matrix by transforming an algebra problem into a statistical problem. In particular, we are interested in the trace of a symmetric matrix $H \in R^{d \times d}$. Then, given a random vector $z \in R^d$ whose component is iid sampled Gaussian distribution ($N(0, 1)$) (or Rademacher distribution), we have:

$$Trace(H) = Trace(HI) = Trace(H\, \mathbb{E}[zz^T]) = \mathbb{E}[Trace(Hzz^T)] = \mathbb{E}[z^T H z], \tag{5}$$

where I is the identity matrix. Based on this, the Hutchinson algorithm [1] can be used to estimate the trace of the Hessian:

$$Trace(H) \approx \frac{1}{m} \sum_{i=1}^{m} z_i^T H z_i = Trace_{Est}(H). \tag{6}$$

Using the above method, we have computed the mean trace of the Hessian for different blocks of Inception-V3 and ResNet50, as shown in Figure 1. As one can see, there is a significant difference between the mean Hessian trace of different blocks of the model. To better illustrate this, we have also plotted the loss landscape of Inception-V3 and ResNet50 by perturbing the pre-trained model along the first and second eigenvectors of the Hessian for each block. It is clear that different layers have significantly different "sharpness". For instance, the 4th block in Inception-V3 is very sensitive, and thus needs to be kept at higher bit precision, whereas the 16th block exhibits a very "flat" loss landscape and can be quantized more aggressively.

[2] We will leave $L(W_i^*; W_1^*, \cdots W_{i-1}^*, W_{i+1}^* \cdots, W_b^*)$ as $L(W_i^*)$ without confusion.

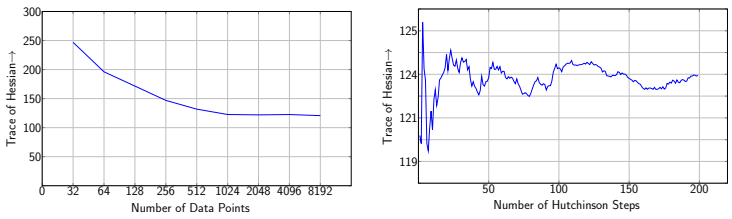

Figure 2: Relationship between the convergence of Hutchinson and the number of data points (Left) as well as the number of steps (Right) used for trace estimation on block 21 in ResNet50.

3 Results

In Figure 2, we show how the convergence of Hutchinson algorithm related to the number of data points and the number of Hutchinson steps used for trace estimation. It can be clearly seen that 4096 data points with 50 Hutchinson steps can already give a very accurate approximation.

We first start with ResNet50 [7] quantization, which is a common benchmark problem. The results are shown in Table 1, along with other quantization methods proposed in the literature [16, 3, 14, 6, 13, 4]. Noted that [16, 3, 14, 6] followed traditional rules which set the precision for the first and last layer to 8-bit, and quantize other layers to an identical precision. Furthermore, both [13, 4] use mixed-precision quantization methods, and [13] uses reinforcement learning to search for a good precision setting, while [4] uses second-order information to guide the precision selection as well as the block-wise fine-tuning. Although [4] uses second-order information to obtain a relative order of quantization precision for each block, it is, as mentioned before, limited to top eigenvalue computation. In contrast, our method uses the Hessian trace information for each block. We can clearly see that our method exceeds the performance of all the other approaches. To the best of our knowledge, this is the state-of-the-art quantization result for ResNet50. Moreover, we also apply our method on InceptionV3 [12]. Direct quantization of InceptionV3 (*i.e.*, without use of second-order information), results in 7.69% accuracy degradation. Using the approach proposed in [8] results in more than 2% accuracy drop, even though it uses higher bit precision. [4] results in a 2% accuracy gap with a compression ratio of $12.04\times$, both of which are better than previous work [8, 9]. As we can see, our method can achieve 75.68% accuracy with the same model size as [4].

Table 1: Quantization results of ResNet50 and Inception-V3 on ImageNet. We show results of state-of-the-art methods [16, 3, 14, 6, 8, 9]. In particular, we also compare with the recent AutoML approach of [13]. Compared to [13], we achieve higher compression ratio with higher testing accuracy. Also note that [16, 3, 14, 6] use 8-bit for first and last layers.

(a) ResNet50

Method	w-bits	a-bits	Top-1	W-Comp	Size
Baseline	32	32	77.39	1.00×	97.8
Dorefa [16]	3	3	69.90	10.67×	9.17
PACT [3]	3	3	75.30	10.67×	9.17
LQ-Nets [14]	3	3	74.20	10.67×	9.17
DeepComp. [6]	3	MP	75.10	10.41×	9.36
HAQ [13]	MP	MP	75.30	10.57×	9.22
HAWQ [4]	2 MP	4 MP	**75.48**	12.28×	7.96
OURS	2 MP	4 MP	**75.76**	12.24×	**7.99**

(b) Inception-V3

Method	w-bits	a-bits	Top-1	W-Comp	Size
Baseline	32	32	77.45	1.00×	91.2
IntegerOnly [8]	8	8	75.40	4.00×	22.8
IntegerOnly [8]	7	7	75.00	4.57×	20.0
RVQuant [9]	3 MP	3 MP	74.14	10.67×	8.55
Direct	2 MP	4 MP	69.76	15.88×	5.74
HAWQ [4]	2 MP	4 MP	75.52	12.04×	**7.57**
OURS	2 MP	4 MP	**75.68**	12.04×	**7.57**

4 Conclusions

In this work, we proposed a new framework for quantizing NNs. Despite existing approaches which use zeroth order or first order methods, we use second order information. We proved that under certain assumptions a good metric to measure quantization sensitivity of different blocks is the mean trace of Hessian matrix. Despite the fact that the theoretical assumptions are very strong, the empirical results showed that our second order method can exceed the state-of-the-art (even when compared to AutoML based methods) for various model architectures including ResNet50 and Inception-V3.

References

[1] Haim Avron and Sivan Toledo. Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix. *Journal of the ACM (JACM)*, 58(2):8, 2011.

[2] Zhaojun Bai, Gark Fahey, and Gene Golub. Some large-scale matrix computation problems. *Journal of Computational and Applied Mathematics*, 74(1-2):71–89, 1996.

[3] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018.

[4] Zhen Dong, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. HAWQ: Hessian aware quantization of neural networks with mixed-precision. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[5] Song Han, Xingyu Liu, Huizi Mao, Jing Pu, Ardavan Pedram, Mark A Horowitz, and William J Dally. Eie: efficient inference engine on compressed deep neural network. In *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, pages 243–254. IEEE, 2016.

[6] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *International Conference on Learning Representations*, 2016.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[8] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2704–2713, 2018.

[9] Eunhyeok Park, Sungjoo Yoo, and Peter Vajda. Value-aware quantization for training and inference of neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 580–595, 2018.

[10] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*, pages 525–542. Springer, 2016.

[11] Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Q-bert: Hessian based ultra low precision quantization of bert. *arXiv preprint arXiv:1909.05840*, 2019.

[12] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[13] Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. HAQ: Hardware-aware automated quantization. *In Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019.

[14] Dongqing Zhang, Jiaolong Yang, Dongqiangzi Ye, and Gang Hua. LQ-Nets: Learned quantization for highly accurate and compact deep neural networks. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[15] Aojun Zhou, Anbang Yao, Yiwen Guo, Lin Xu, and Yurong Chen. Incremental network quantization: Towards lossless cnns with low-precision weights. *International Conference on Learning Representations*, 2017.

[16] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016.