Auto Hessian Aware Channel-wise Quantization of Neural Networks

Xu Qian
Intel Asia Research Center
xu.qian@intel.com

Victor Li
Intel Asia Research Center
victor.li@intel.com

Crews Darren S
Intel IOTG
darren.s.crews@intel.com

Abstract

Second-order information has proven to be very effective in determining the redundancy of neural network weights and activations. Recent paper proposes to use the Hessian traces of weights and activations for mixed-precision quantization and achieves state-of-the-art results. However, prior works only focus on selecting bits for each layer while the redundancy of different channels within a layer also differ a lot. This is mainly because the complexity of determining bits for each channel is too high for original methods. Here, we introduce Auto Hessian Aware Channel-wise Quantization (AHCQ). AHCQ uses the Hessian traces to determine the relative sensitivity order of different channels of activations and weights. What's more, AHCQ proposes to use deep Reinforcement learning (DRL) Deep Deterministic Policy Gradient (DDPG)-based agent to find the optimal fractions of channels to be quantized to different quantization bits and assign bits to channels according to the order of Hessian traces. The agent training time of AHCQ is much shorter compared with traditional AutoML based mix-precision methods since we only need to search the fractions of different quantization bits. Comparing AHCQ with state-of-the-art shows that we can achieve better results for multiple networks.

1 Introduction

In order to increase inference speed and energy efficiency, weights and activations of neural networks need to be quantized to low precision[3]. Prior works use the same QBN(quantization bit number) for all layers[1][4][7], which would introduce a huge decrease in accuracy as bit goes down.

A possible solution here is to use mixed-precision quantization, where higher precision is used for more sensitive layers of the network, and lower precision for less sensitive layers. However, the search space of choosing a QBN for each layer is too large for exhaustive search.

HAWQ-V2[2] finds that second order information, like Hessian traces of different layers are great indicators of the layers' sensitivity to quantization error. HAWQ-V2 uses the Hessian traces as criterion for layers' sensitivity and is able to outperform most model quantization methods.

However, HAWQ-V2 only focuses on selecting QBNs for different layers, while the redundancy of channels within a layer also differs a lot[5]. A more fine-grained method would be using mixed-precision for different channels. But the search space is much larger for channel-wise mixed precision problem. Traditional methods like HAWQ-V2 are unable to solve this kind of problem.

A recent paper AutoQ[5], proposes to use a hierarchical-DRL-based agent to solve this problem. Based on the insights of HAQ[6], AutoQ automatically searches a QBN for each weight kernel and choose a QBN for each activation layer. However, the state number is quite huge for AutoQ and the agent requires a lot of training time to converge. Also, the author did not consider using mixed-precision for different channels of activations.

Submitted to 6th Workshop on Energy Efficient Machine Learning and Cognitive Computing (EMC2 2020). Do not distribute.

2 Method

Hessian trace is a great indicator of a neural network block's sensitivity to quantization error[2]. For channel-wise mixed-precision quantization, using the Hessian traces can significantly reduce the search space, since a channel with a higher Hessian trace cannot be assigned lower bits, as compared to another channel with a smaller Hessian trace. Based on that, we are able to sort all channels according to their average Hessian traces. Then the mixed-precision problem can be reduced to an integer partition problem, namely, to partition the sorted channels into different groups of QBNs. However, the search space is still huge after Hessian trace analysis because there are too many channels. For channel-wise mixed precision quantization on ResNet50 where QBNs are chosen in {2, 3, 4, 5, 6, 7, 8}, the search space is as large as $\binom{7}{27000} \approx 2.1 \times 10^{27}$.

We model this task as a DRL problem. We use the actor-critic model with DDPG agent to give the action: fractions of channels to be quantized to different QBNs. With the prior knowledge of the Hessian traces, the states of the agent only contain the fractions of different bits, in our experiments, fractions of {2, 3, 4, 5, 6, 7, 8}. This is much smaller compared with traditional AutoML based quantization methods where each layer or channel has an individual state like HAQ and AutoQ.

This section is organized as follows: first we will introduce the method to generate the traces of different channels and analyze the traces of ResNet18 as an example. Then we will discuss in details about the DRL method to determine the fractions of channels to be quantized to different QBNs.

2.1 Channel-wise trace weighted quantization

Computing the Hessian traces may seem a prohibitive task, as we do not have direct access to the elements of the Hessian matrix. Hence in HAWQ-V2, the author uses Hutchinson algorithm[2] to estimate the Hessian trace of a neural network layer.

Based on that, we introduce the masked Hutchinson algorithm to calculate the traces for different weight channels:

$$\begin{gathered} Tr(H_w^j) \approx \frac{1}{m}\sum_{i=1}^{m} {z_i^j}^T H_w z_i^j = Tr_{Est}(H_w^j) \\ z_i^j = Mask_j * z_i, j \in [0, output\ channels] \end{gathered} \tag{1}$$

where z_i^j is the masked random vector for the j^{th} channel of a given layer. H_w^j is the weight Hessian matrix of j^{th} channel. The j^{th} channel of $Mask_j$ is set to 1 while the others set to 0.

Similarly, we can get the average activation trace of the j^{th} channel using:

$$\begin{gathered} Tr(H_a^j) \approx \frac{1}{N}\sum_{i=1}^{N} {z_i^j}^T H_a(x_i) z_i^j \\ z_i^j = Mask_j * z_i, j \in [0, output\ channels] \end{gathered} \tag{2}$$

where H_a^j is the activation Hessian matrix of j^{th} channel.

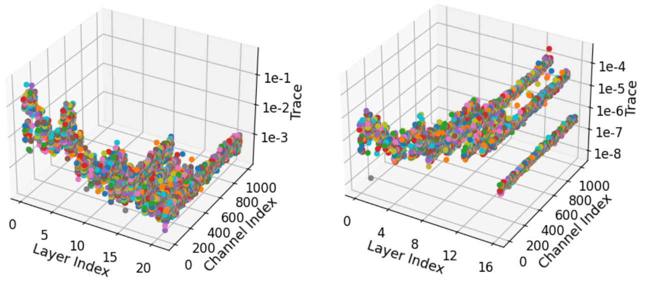

Figure 1: Average Hessian traces of different blocks in ResNet18 on ImageNet (a) Average trace of different weight channels; (b) Average trace of different activation channels;

We incorporate the above approach and compute the average Hessian traces for different layers and channels of ResNet18. As shown in Figure 1, there is a significant difference between average Hessian traces for different channels. This conclusion works both for weights and activations.

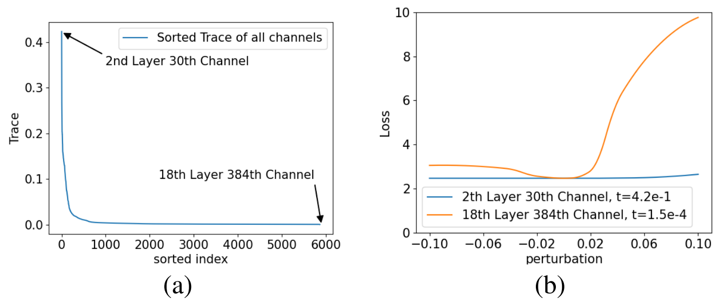

Figure 2: Hessian trace analysis (a) Sorted average Hessian traces in descending order; (b) The loss landscape of channels with min and max trace

We sort the weight Hessian traces of different channels of ResNet18 in descending order as Figure 2(a) shows. The difference of different channels can be as large as 10^3.

To better illustrate this, we also plot the loss landscape of ResNet18 by adding perturbation to the pre-trained model as Figure 2(b). It is clear that different channels have significantly different "sharpness". For the 30^{th} channel of the second layer, the average trace is about 4.2×10^{-1}. The loss changes significantly as we add perturbation to this channel, so we need to use higher bits for this channel. And for the 384^{th} channel of the 18^{th} layer, the average trace is about 1.5×10^{-4}. The loss landscape is relatively flat so we can quantize this channel more aggressively.

2.2 Quantization bit allocation

However, a major drawback for Hessian trace analysis is that it does not provide the specific bit precision setting for different channels. For example, in Figure 1, it is clear that some channels have significant higher average traces than the others. Although it is obvious that we should allocate higher QBNs for these channels to increase accuracy, we still cannot get a specific bit precision setting.

Channels with higher traces should be assigned higher bits. The only parameters missing are the fractions of channels to be quantized to different QBNs. In this paper, we model this task as a DRL problem as Figure 3 shows. We use the actor-critic model with DDPG agent to give the action: fractions of channels to be quantized to different QBNs. Then we allocate the bits to the channels according to the sorted Hessian traces. In our experiments, we use compression ratio as constraint and accuracy as target to search for the optimal fractions. Below describes the details of our implementation.

2.2.1 DDPG agent

In the previous section, we have calculated the Hessian traces of different channels. Leveraging the sorted Hessian traces, we iteratively search the fractions of channels to be quantized to $\{2, 3, 4, 5, 6, 7, 8\}$ bits.

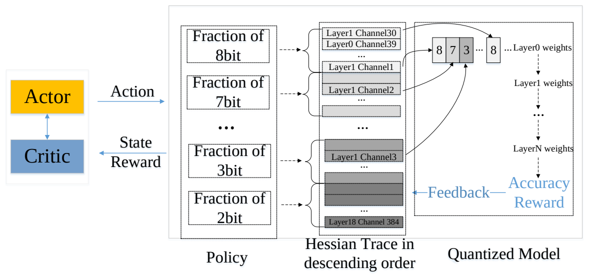

Figure 3: An overview of our framework. We leverage DRL to automatically search the fractions of different bits and allocate the bits to different channels according to the sorted Hessian traces

Method	w-bits	a-bits	Top-1	Top-1 drop	W-Comp	Size(MB)
Baseline	NA	NA	76.45	NA	1.00x	97.8
HAQ	3.03	MP	75.30	0.85	10.57x	9.22
HAWQ-V2	2.61	4_{MP}	75.76	1.63	12.24x	7.99
AutoQ	2.21	3.07	72.47	2.33	14.48x	6.75
AHCQ	2.61	4	76.65	-0.2	12.24x	7.99
AHCQ	2.21	3.07	75.57	0.88	14.48x	6.75

Table 1: Results of ResNet50 on ImageNet. We abbreviate average QBN used for weights as "w-bits," activations as "a-bits," top-1 accuracy as "Top-1," and weight compression ratio as "W-Comp." Here "MP" refers to mixed-precision quantization. "Top-1 drop," column is added for a fair comparison because different methods have different baseline.

Method	w-bits	a-bits	Top-1	Top-1 drop	W-Comp	Size(MB)
Baseline	NA	NA	70.47	NA	1.00x	44.65
HAQ	3.37	3.65	67.5	2.4	9.49x	4.71
AutoQ	2.19	3.02	67.4	2.5	14.61x	3.06
AHCQ	2.19	3.02	69.02	1.45	14.61x	3.06

Table 2: Quantization results of ResNet18 on ImageNet.

Observation (State Space). In this process, there is no need for extra information from the network because it is implied in the sorted Hessian trace list. So the state space is relatively simple, in this paper we introduce a 5 dimension feature vector O_k as our observation:

$$[k, n_{remains}, s_i, e_i, a_{k-1}]$$

where k is the channel index, s_i and e_i are the starting index and ending index of the last bit in the sorted Hessian trace list, $n_{remains}$ is the number of remaining parameters, and a_{k-1} is the action from the last step. For each dimension in the observation vector O_k, we normalize it into $[0, 1]$ to make them in the same scale.

Action Space. Due to the fact that all fractions need to sum up to exact 100%, the Markov Decision Process in this paper is as follows: we first search the fraction of 2-bit and quantize corresponding channels according to the sorted Hessian traces. Then search the fraction of 3-bit in the remaining channels and quantize these channels accordingly. Then we repeat this operation for 4-bit to 7-bit. Finally we quantize the remaining channels to 8-bit.

Reward Function. After quantization, we retrain the quantized model for one more epoch to recover the accuracy. As we have already imposed the resource constraints by limiting the action space, we define our reward function to be exactly the retraining accuracy top-1.

Quantization and fine-tuning. We follow a two step quantization method. In the first step, we use DDPG agent to search for the optimal policy for activation quantization and quantize the activations accordingly. In the second step, we use the fine-tuned model from step1 to search for the optimal policy for weight quantization. Then we use the best policy to add weights quantization to the fine-tuned model from step1.

3 Experiments

To evaluate AHCQ, we select several CNN models including ResNet-18, ResNet-50 and MobileNetV2. The CNN models are trained on ImageNet including 1.26M training images and tested on 50K test images spanning 1K categories of objects.

Method	w-bits	a-bits	Top-1	Top-1 drop	W-Comp	Size(MB)
Baseline	NA	NA	71.8	NA	1.00x	13.51
HAQ	3.21	3.92	68.66	2.44	9.97x	1.36
AutoQ	2.26	3.13	68.68	2.42	14.16x	0.95
AHCQ	2.26	3.13	69.85	1.95	14.16x	0.95

Table 3: Quantization results of MobilenetV2 on ImageNet.

3.1 Compare with State Of The Art

As shown in Table 1, we apply AHCQ on ResNet50 and compare the accuracy to state-of-the-art mixed precision methods. It is clear that AHCQ achieves state-of-the-art results compared with multiple methods. At 2.61 average weight QBN and 4 average activation QBN, AHCQ is even able to increase the baseline accuracy by 0.2%.

We also experiment with ResNet18. As in Table 2, we compared AHCQ with AutoQ and HAQ. AHCQ is able to achieve about 1.6% higher accuracy than AutoQ at the same compression ratio. Our Top-1 drop is about 1% smaller considering AutoQ uses a lower baseline.

We also tried smaller networks like MobilenetV2. As in Table 3, AHCQ is able to achieve about 1.2% better accuracy than AutoQ at the same compression ratio. The Top-1 drop is also about 0.5% smaller.

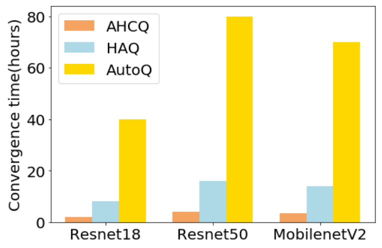

Figure 4: Agent training wall time for AHCQ vs AutoQ and HAWQ-V2

3.2 AutoML training efficiency.

One major drawback for AutoML based bit selection is that it takes a lot of time to train the agents. This is because the state space of previous methods (HAQ, AutoQ) is quite huge. While in AHCQ, the states only consist of the fractions of channels to be quantized to different QBNs ($\{2, 3, 4, 5, 6, 7, 8\}$). As a result, our agent is able to converge in a relative short time. As Figure 4 shows, the training time for AHCQ is much less compared with AutoQ and HAQ. For Resnet18, AHCQ converges after 2 hours' training with 4 Nvidia V100 GPUs, which is about 1/4 of HAQ and 1/20 of AutoQ.

4 Conclusion

In this paper, we propose Reinforcement learning based Hessian aware Channel-Wise quantization (AHCQ). AHCQ first use the masked Huntchison algorithm to calculate the average Hessian traces of weight and activation channels. Then AHCQ sorts the channels and uses DDPG agent to automatically select the fractions of different QBNs. Particularly, the training efficiency of AHCQ improves a lot compared with traditional AutoML based quantization methods. Experiments on multiple networks show the effectiveness of AHCQ.

References

[1] J. Choi, Z. Wang, S. Venkataramani, P. I.-J. Chuang, V. Srinivasan, and K. Gopalakrishnan. Pact: Parameterized clipping activation for quantized neural networks, 2018.

[2] Z. Dong, Z. Yao, Y. Cai, D. Arfeen, A. Gholami, M. W. Mahoney, and K. Keutzer. Hawq-v2: Hessian aware trace-weighted quantization of neural networks, 2019.

[3] S. Han, H. Mao, and W. J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding, 2015.

[4] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference, 2017.

[5] Q. Lou, F. Guo, L. Liu, M. Kim, and L. Jiang. Autoq: Automated kernel-wise neural network quantization, 2019.

[6] K. Wang, Z. Liu, Y. Lin, J. Lin, and S. Han. Haq: Hardware-aware automated quantization with mixed precision, 2018.

[7] S. Zhou, Y. Wu, Z. Ni, X. Zhou, H. Wen, and Y. Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients, 2016.