Post Training Mixed Precision Quantization of Neural Networks using First-Order Information

Arun Chauhan*, Utsav Tiwari*, Vikram N R
Samsung Research Institute
Bangalore, India
{arun.c, u.tiwari, vikram.nr}@samsung.com

Abstract

Quantization is an efficient way of downsizing both memory footprints and inference time of large size Deep Neural Networks (DNNs) and makes their application feasible on resource-constrained devices. However, quantizing all layers uniformly with ultra-low precision bits results in significant degradation in performance. A promising approach to address this problem is mixed-precision quantization where higher bit precisions are assigned to layers that are more sensitive. In this study, we introduce the method that uses first-order information (i.e. gradient) only for determining the neural network layers' sensitivity for mixed-precision quantization and shows that the proposed method is equally effective in performance and better in computation complexity with its counterpart methods which use second order information (i.e. hessian). Finally, we formulate the mixed precision problem as an Integer linear programming problem which uses proposed sensitivity metric and allocate the number of bits for each layer efficiently for a given model size. Furthermore, we only use post training quantization techniques to achieve the state of the art results in comparison to the popular methods for mixed precision quantization which fine-tunes the model with large training data. Extensive experiments conducted on benchmark vision neural network architectures using ImageNet dataset demonstrates the superiority over existing mixed-precision approaches. Our proposed method achieves better or comparable results for ResNet18 (0.65% accuracy-drop, for 8× weight compression), ResNet50 (0.69% accuracy-drop, for 8× weight compression), MobileNet-V2 (0.49% accuracy-drop, for 8× weight compression) and Inception-V3 (1.30% accuracy-drop, for 8× weight compression), compared to other state-of-the-art methods which requires retraining or uses hessian as a sensitivity metric for mixed precision quantization.

*These authors contributed equally to this work.

1. Introduction

In recent years, the applications using deep neural networks are growing by a large number due to its impressive performance on various artificial intelligence tasks. Introducing large number of parameters has shown substantial gain in accuracy which motivate researchers to train the neural networks with large number of parameters. However, the world is moving towards extensive use of edge devices like- smartphone, smartwatch, many more wearable and non-wearable devices. To enable ML-based real-time-applications on edge devices there are mainly three constraints- power, memory and latency. To address this problem, researchers offer number of techniques to compress and accelerate DNNs, including knowledge distillation [12, 18], channel pruning [15, 16] and quantization [24, 26].

Among these methods, quantization shows promising results by representing 32-bit floating point weights and activation with fewer fixed-point bits without changing the original architecture of the model. Though, quantization is much favoured for compressing model multifolds with significant acceleration in inference, it is still the topic of research-interest because of the accuracy gap between original and quantized model when compressed significantly. In particular, many approaches have been tried for quantization which can be broadly categorized as uniform and non-uniform quantization, symmetric and asymmetric quantization, static and dynamic quantization, stochastic quantization and mixed-precision quantization [9]. Attempts have been made in each category to narrow-down the gap between accuracy of the original model and the quantized model, however, some of approaches (like non-uniform quantization) are generally difficult for efficient implementation on existing hardware.

To achieve large compression, some of the work attempts to quantize the models in very low precision-bits. However, due to low precision quantization their is a undesirable drop in accuracy which obstructs the objective to achieve the higher compression. One of the promising approach to achieve larger compression which shows encouraging results is mixed-precision quantization, where different layers are quantized with different bit-precision. The main challenge in mixed-precision quantization is the exponential search space which grows with number of layers in the neural network. For L layer neural network which may be quantized with D different bit-precision settings, the size of the search space is D^L. Computationally, it is infeasible to evaluate all the settings in the permissible time period for required accuracy and inference time. State-of-the-art try to address this problem by proposing different strategies either searching more efficiently through this exponential space or using some criteria to evaluate the performance of different bit-precision setting in minimal time. Searching methods are time consuming and requires huge amount of computational resources for evaluating the performance of models for every setting returned by these searching methods along the search trajectory. In addition, like other searching methods, quantization results are very much dependent on the initial value of the search parameters and hence unpredictable. Criteria based methods use some heuristics and try to evaluate the performance of the model for given bit-precision setting without actually training the model, hence they are less time consuming in comparison to the search-based methods.

Recently, some of the notable work [3, 7, 8] for mixed-precision quantization used second order information i.e., hessian, to measure the sensitivity of the layers' parameters of the pre-trained neural network. **Sensitivity** can be understood as the change in loss function when the optimal parameters of the trained model are perturbed. Though, hessian provides a lot of information to measure the sensitivity, the computing cost of the hessian is quadratic in number of parameters and therefore very expensive in case of deep neural networks. On the other hand, the first-order information i.e., gradient is zero at the point of convergence (optimal point) and therefore provides no information about the curvature of the loss function. In this paper, we introduce a novel approach for measuring the sensitivity of the layers using first-order information in the ϵ-$neighbourhood$ of optimal point for mixed-precision quantization and show promising results. The mixed-precision quantization in general can be seen as an optimization problem where the objective is to minimize loss function over all possible bit-precision assignments for given constraints. In this work, we formulate the mixed-precision quantization problem as an instance of Integer Linear Programming Problem (ILPP) for given compression ratio. This simple formulation can also be used to generate pareto-front by solving the ILPP for given range of compression ratio. Finally extensive experiments conducted on benchmark vision neural networks architectures using ImageNet dataset demonstrates the superiority over existing mixed-precision approaches.

To summarize, we propose a gradient guided approach for mixed-precision quantization. In particular, our main contributions are as follows:

- We show that expected average norm of the gradients, taken in the close proximity of the point of convergence, can be considered as a sensitivity metric for mixed precision quantization. The computation complexity for estimating the expected average norm of gradients is better than the complexity to estimate the average hessian trace [7] or approximating hessian [3].
- We formulate the problem of assigning different bit-precision to different layers using proposed sensitivity metric (expected average norm of the gradients) as a constraint optimization problem.
- We perform extensive experiments to show that state of art accuracy can be achieved for mixed-precision post training quantization with very limited access of training dataset ($\approx 0.002\%$, in case of ImageNet).

2. Related Work

Deploying large size deep neural network on edge devices is challenging due to limited memory and computing resources. To overcome this problem many literature work proposed hardware friendly solutions which includes pruning, neural architecture search, knowledge distillation and quantization. Here we briefly review the work related to ours and suggest interested reader to refer the recent survey [9, 22] for a comprehensive overview.

Quantization is a promising approach for compressing deep neural networks using low precision bits. A major challenge in quantization is to retain the accuracy of the original model. To achieve this, several methods propose retraining of the model to recover the lost accuracy [3, 7, 8, 10, 13, 17, 23]. However to perform retraining, full dataset is required that was used to train the original model. Retraining not only takes long time but need access to the training data which may not always be possible. To address this, state of the art propose various methods which focus on quantization that do not require retraining the model. [6] proposed OMSE method that optimize L_2 distance between the original weights and quantized weights. ACIQ [1] analytically compute the clipping range and the channel-wise bit precision setting and shows good generalization performance. [25] propose outlier channel splitting method to overcome the severe impact of outliers on quantization which require access to the limited training data. One of the notable work is AdaRound [19] which perform adaptive rounding that is better optimal than round-to-nearest method. Similar to OCS, AdaRound also require limited training data for adaptive rounding. In this work, we use adaptive rounding after performing quantization and show promising results in Section 4. To the extreme, few notable work proposed methods which do require training or testing data during quantization [2, 11, 20].

Quantizing layers of neural network uniformly with ultra low precision bit is sub-optimal solution for large compression. To achieve large compression, state of the art show promising results for mixed precision quantization where different layers are quantized with different bit precision [3, 7, 8, 17, 23]. However, these approaches require retraining which is time-consuming and cannot be possible if full training data set is not available. We address this limitation in our work and show promising results by using approximately 0.002% of the training data with no retraining.

3. Methodology

Here our focus is on supervised framework where the objective is to minimize the empirical risk loss $\mathscr{L}(\theta)$,

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

where $\theta \in \mathbb{R}^d$ is the trainable model parameters, $l(\hat{y_i}, y_i)$ is the loss between the model prediction $\hat{y_i}$ and the ground truth y_i, $f(x_i,\theta)$ is the model that maps input sample x_i to a prediction $\hat{y_i}$ using parameters θ, N is the training set cardinality. We assume that the neural networks is partitioned in L blocks as $\{B_1, B_2, ..B_L\}$ which are parametrized by learnable parameters, $\{W_1, W_2, \ldots, W_L\}$ respectively.

For quantization, we assume that neural network is already trained by minimizing the $\mathscr{L}(W)$ using backpropagation algorithm and converged to a point which satisfy first-order optimal condition (i.e., $\nabla_W \mathscr{L}(W) = 0$). All weights and activations of the model are stored in single precision (32-bit floating point numbers). While quantizing, our aim is to reduce the inference time and memory footprints by representing the floating point weights, W with quantized weights, $\widetilde{W}$ as:

$$Q_b(z) = q_j,\ for\, z \in (t_j, t_{j+1}] \tag{2}$$

where z denotes either weight and activation, $(t_j, t_{j+1}]$ is an interval in the real numbers $(j = 0, 1, \ldots, 2^b - 1)$ and b is the quantization precision. This means that all the values in the range of $(t_j, t_{j+1}]$ are mapped to q_j which is equivalent to dequantized value of z.

One much favoured practice is to use uniform bit precision for all the blocks. However, quantizing all the blocks uniformly with ultra low bit precision may results into significant drop in accuracy. It has been argued that different layers of the network have different sensitivity and therefore bit precision used for quantizing different layers should be chosen accordingly. However, as mentioned above the search space for assigning bit precision to different layers is exponential in number of layers. In the following section, we do the theoretical analysis to obtain the sensitive metric and propose a method to assign bit precision for each layer by avoiding exponential search space.

3.1. Quantization Sensitive Metric

The model is more sensitive to the perturbation if the curvature of the loss function is large (imply large eigen values of hessian) at point of convergence. Larger curvature implies a substantial change in loss function for the same amount of perturbation in comparison to the smaller curvature of the loss function. Attempts [7, 8] have been made to measure the curvature either using top eigen value and average hessian trace or approximating hessian [3]. In this work, we hypothesize that expected value of the average norm of the gradients, taken in the close proximity of the point of convergence, can be considered as a quantization sensitive metric. We also hypothesize that the proposed metric may consider the possibility of saddle point at the point of convergence, hence estimate sensitivity better than average hessian trace [7]. As a simple example, this can be illustrated by considering the function $F(x, y) = 10x^2 - 10y^2$. The trace of the hessian for $F(x, y)$ is zero at the origin ($i.e., x = y = 0$) and therefore represents zero sensitivity for perturbation. In contrast, computing the expected value of the average of the norm of the gradients taken at $(x = 0, y = 0.1)$, $(x = 0.1, y = 0)$, $(x = 0, y = -0.1)$, and $(x = -0.1, y = 0)$ is 2 which is more accurate proxy then the trace of the hessian.

3.1.1 First-Order Information

The first-order information is zero at the point of convergence. However, it is possible to compute the gradient in ϵ-$neighbourhood$ of the convergence point which may provide the first-order information of the curvature in the ϵ-$neighbourhood$. We hypothesize that the expected average gradient norm in the ϵ-$neighbourhood$ of the convergence point can be considered as a quantization sensitive metric. To illustrate that how this can be done for neural network, let us denote W_i^* as the converging point of i^{th} block of the network and δW_i^* is very small perturbation to the W_i^* such that $0 < \|\delta W_i^*\|_2 < \epsilon$. Then the average gradient norm denoted by g_i^{av} at the perturbed point, $W_i^* + \delta W_i^*$ can be computed as

$$g_i^{av} = \frac{1}{n_i}\left\|\nabla_{\delta W_i^*}\mathscr{L}(W_i^* + \delta W_i^*)^*\right\|_1 \tag{3}$$

where n_i is the dimension of W_i^* and $\|.\|_1$ refers to the norm one of the vector. To compute the expection of g_i^{av} denoted by $\mathscr{G}_i$, we draw δW_i^* several times from Gaussian distribution and scale them accordingly.

$$\mathscr{G}_i = \mathbb{E}_{\delta W_i^*}\left[g_i^{av}\right] \tag{4}$$

From now we denote $\mathscr{G}_i$ as expected average gradient norm. We hypothesize that when block B_i and B_j with expected average gradient norm $\mathscr{G}_i$ and $\mathscr{G}_j$ respectively, is quantized with same amount of perturbation such that $\|\Delta W_i^*\|_2^2 = \left\|\Delta W_j^*\right\|_2^2$. Then we will have:

$$\|\mathscr{L}(W_i^* + \Delta W_i^*) - \mathscr{L}(W_i^*)\|_2 < \left\|\mathscr{L}(W_j^* + \Delta W_j^*) - \mathscr{L}(W_j^*)\right\|_2 \tag{5}$$

if

$$\mathscr{G}_i < \mathscr{G}_j \tag{6}$$

The intution is that if expected average gradient norm in the ϵ-$neighbourhood$ of j^{th} block is greater than i^{th} block, then for the same amount of perturbation (i.e. $\|\Delta W_i^*\|_2 = \left\|\Delta W_j^*\right\|_2$) the relative change (i.e. $\|\Delta W_i^*\|_2 \mathscr{G}_i < \left\|\Delta W_j^*\right\|_2 \mathscr{G}_j$) in loss is also greater. This implies that j^{th} block is more sensitive than i^{th} block for perturbation.

3.2. Algorithm

Algorithm 1 Expected Average Gradient Norm of $i^{th} block$

Input: Block Parameters W_i^*
Output: Expected Average Gradient Norm, $\mathscr{G}_i$

```
1: Let $g_i^{av} = 0$.
2: for $t = 1, 2, 3, .., p$ do
3:     $d_t \sim \mathcal{N}(0, \mathrm{I})$
4:     Compute $\delta W_t^* = k \frac{d_t}{\|d_t\|_2}$
5:     Compute $g_t = \nabla_{\delta W_t^*} \mathscr{L}(W_i^* + \delta W_t^*)$
6:     $g_i^{av} = g_i^{av} + \frac{\|g_t\|_1}{n_i}$
7: end for
8: return $\mathscr{G}_i = \frac{g_i^{av}}{p}$
```

Computing expected average gradient norm of i^{th} block is shown in Algorithm 1. The expected average gradient norm measures the quantization sensitivity which is used to assign the bit-precision to different model layers accordingly. To measure the sensitivity of the i^{th} block, we keep the remaining blocks except i^{th} block to the original weights. To define the close proximity, δW_i^*, of convergence point, W_i^*, first we obtain the random direction vector, d, sampled from the random Gaussian distribution with dimension compatible with W_i^*. Then we substitute $\delta W_i^* \leftarrow k \frac{d}{\|d\|_2}$ to remove the scaling effect [14]. Here we get random directions with constant $\|\delta W_l^*\|_2$, k is scalar value to tune the perturbation across the layers. Please see Appendix for more details on value of k. We compute the average of norm of the gradient, $g_i^{av} = \frac{1}{n_i} \nabla_{\delta W_i^*} \mathscr{L}(W_i^* + \delta W_i)$, where n_i is the dimension of W_i^*. Finally, we compute the expected average gradient norm in the ϵ-$neighbourhood$. In Section 4 we show empirically that the algorithm has good convergence properties with much smaller computation complexity in comparison to training the model itself.

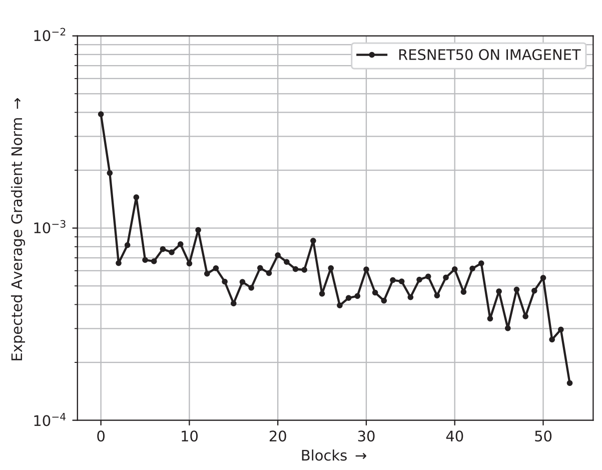
We use the Algorithm 1 to compute the expected average gradient norm of different layers of ResNet50 as shown in Figure 1 *(Right)*. We can clearly observe the comparable difference between the expected average gradient norm for different layers of the model. To further support the observation, we have plotted the 1D loss landscape by perturbing the pretrained weights in ten random directions for ResNet50 along which we computed average gradient norms. From Figure 1 *(Left)*, it is evident that the Block 1 and Block 49 of ResNet50 on ImageNet has very significant difference in expected average gradient norm which is also visible in Figure 1(Right) for 1D loss landscape. We reproduce the same plots for InceptionV3, MobileNetV2 and ResNet18 which are shown in Appendix. The blocks having higher sensitivity needs to be kept at higher bit precision in comparison to the blocks having lower sensitivity.

3.3. Constraint Optimization Problem Formulation

The major challenging problem in mixed-precision quantization is to assign precision bits to different blocks based on the block's sensitivity without loosing accuracy of the model. We use expected average gradient norm to measure the sensitivity of different blocks as shown in Figure 1 *(Left)*. However, the problem of assigning the bit precision to different block still persist. For example, Block 1 and Block 4 of ResNet50 are having much larger expected average gradient norm in comparison to Block 48 and Block 49 which suggest that Block 1 and Block 4 should be assigned higher bit precision than Block 48 and Block 49 bit precision. We still may not be able to get the precise precision bit settings for each block of the model using Algorithm 1. We formulate mixed-precision quantization problem as a constrained optimization problem, as our solution space contains all boolean decision based parameters, so we chose Integer Linear Programming to solve this problem. We hypothesize that the expected average gradient norm of the block can be considered as an average slope of the loss landscape in the close proximity of the original weights of the block of the model. We use this average slope to measure the effect of perturbation on model's loss. The change in loss will be directly proportional to the product of the L_2 norm of the quantization perturbation with the average slope. This is illustrated in Figure 2 *(Left)* and defined as follows:

$$\mathrm{M}_{l,b} \propto \mathscr{G}_l \left\|Q_b(W_l^*) - W_l^*\right\|_2^2 \tag{7}$$

where $Q_b(W_l^*)$ is the quantized weights of l^{th} block



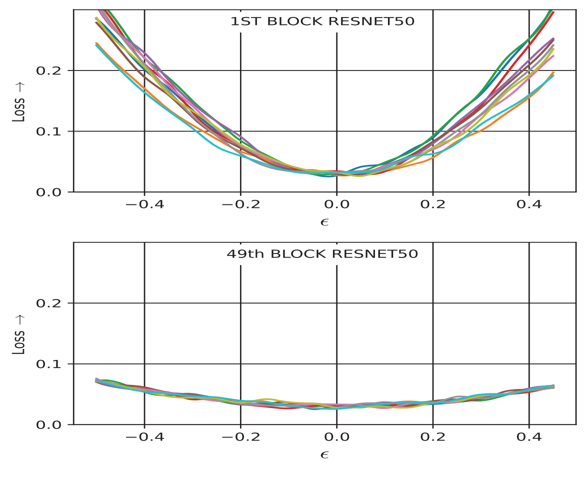
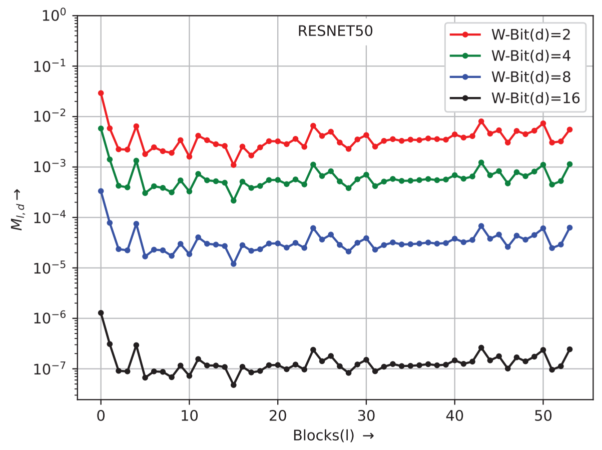


Figure 1: *(Left)* Expected Average Gradient Norm of different blocks in ResNet50 on ImageNet. *(Right)* The plot of 1D loss landscape for Block 1 and Block 49 of ReNet50 on ImageNet. The landscape is plotted by perturbing the pretrained weights of the model in ten random directions along which we computed average gradient norm ($\epsilon = 0$ corresponds to no perturbation). Higher curvature *(top)* shows higher sensitivity and lower curvature *(bottom)* shows lower sensitivity.



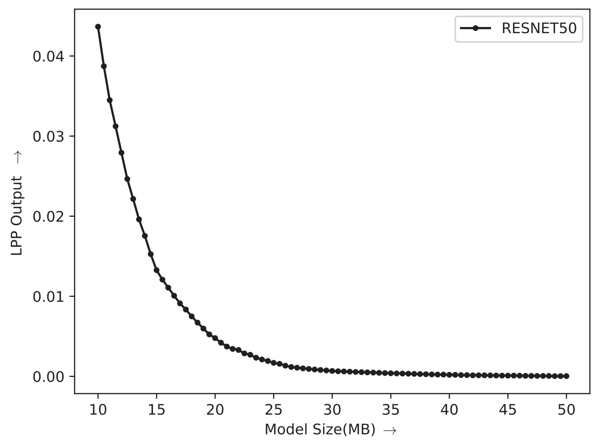


Figure 2: *(Left)* Sensitivity of ResNet50 when quantized to 2/4/8/16-bit weight precision. *(Right)* The trade-off between model size and the ILPP formulation (Equation 8) output for ResNet50.

of the model using b precision bits, W_l^* is the original weights of the block of the model, $\mathscr{G}_l$ is the expected average gradient norm of l^{th} block. Please note that total change in loss due to total perturbation using b precision bits in all L blocks is directly proportional to $\sum_{l=1}^{l=L} \mathscr{G}_l \, \|Q_b(W_l^*) - W_l^*\|_2^2$.

In ILPP formulation, we find the precise bit settings of each block for the given target model size with minimum change in loss due to perturbation. We denote the set of D admissible bit precision settings used for mixed-precision quantization by set $\mathrm{K} = \{b_1, b_2, ..., b_D\}$. In particular, the objective function is to minimize the total change in loss of the model due to total perturbation. The ILPP formulation for mixed-precision quantization is shown in equation 8, where $\mathrm{M}_{l,d}$ is the change in loss due to perturbation of l^{th} block using b_d precision bits, $\mathrm{C}_{l,d} \in \{0, 1\}$ is the binary variable which represents the precision bit setting, b_d is used for quantizing l^{th} block or not, $\mathrm{S}_{l,d}$ is the size of the l^{th} block when b_d bits are used for quantizing the l^{th} block, S_{target} is the desired target model size after mixed-precision quantization. Please note that each block is quantized by any one of the admissible precision bit set-

Algorithm 2 Assigning bit precision using ILPP

Input: Block-wise expected average gradient norm ($\mathscr{G}_l$) in the proximity of W_l (from Algorithm 1), $\|\Delta W_l^*\|_2$, number of parameters in l^{th} layer, n_l for $l = 1, 2, ..., L$, bit-precision set K, target model size S_{target}

Output: bit-width assignment of each layer $\{a^l\}_{l=1}^L$

```
1:  Intialize $\{\{\mathrm{C}_{l,d}\}_{d=1}^D\}_{l=1}^L$ with zero
2:  Compute $\{\{\mathrm{P}_{l,d} = \|Q_{b_d}(W_l) - W_l\|_2^2\}_{d=1}^D\}_{l=1}^L$
3:  Compute $\{\{\mathrm{M}_{l,d} = \mathscr{G}_l \mathrm{P}_{l,d}\}_{d=1}^D\}_{l=1}^L$
4:  Compute $\{\{\mathrm{S}_{l,d} = n_l b_d\}_{d=1}^D\}_{l=1}^L$
5:  Solve equation 8
6:  for $l = 1, 2, 3, ..., L$ do
7:      for $d = 1, 2, 3, ..., D$ do
8:          if $\mathrm{C}_{l,d}$ is equal to one then
9:              $a^l = b_d$ {bit–precision assign for each layer}
10:         end if
11:     end for
12: end for
13: return $\{a^l\}_{l=1}^L$
```

tings from the set K which we enforce by the constraint $\sum_{d=1}^D \mathrm{C}_{l,d} = 1; \quad \forall l \in \{1, 2, .., L\}$. This simple formulation can be used to generate pareto-front by solving the ILPP for given range of model size as shown in Figure 2 *(Right)*.

$$\begin{aligned} \min_{\mathrm{C}_{l,d}} \quad & \sum_{l=1}^{L}\sum_{d=1}^{D} \mathrm{M}_{l,d}\mathrm{C}_{l,d} \\ s.t. \quad \sum_{d=1}^{D} \mathrm{C}_{l,d} = 1; \quad & \forall l \in \{1, 2, .., L\} \\ \sum_{l=1}^{L}\sum_{d=1}^{D} \mathrm{S}_{l,d}\mathrm{C}_{l,d} & \leq \mathrm{S}_{target} \\ \mathrm{C}_{l,d} & \in \{0, 1\} \end{aligned} \tag{8}$$

We show the steps of choosing exact bit precision in Algorithm 2. Expected average gradient norm is computed using Algorithm 1 for each block of the model which are then passed as the input to Algorithm 2 along with the K which denotes set of all admissible bit precision settings, and the target model size S_{target}. At first, the binary matrix $\mathrm{C}_{L\times D}$, is initialized to zero. In second step, we compute perturbation matrix $\mathrm{P}_{L\times D}$, where each entry constitutes the Euclidean norm of the perturbation to the l^{th} block when quantized using b_d precision bits that is $\|Q_{b_d}(W_l) - W_l\|_2^2$. In third step, we compute the estimated loss-change matrix $\mathrm{M}_{L\times D}$ as explained in Equation 7. At last, we compute the size matrix $\mathrm{S}_{L\times D}$, where each entry constitutes the size of each block having n_l weights when quantized by b_d precision bits. Finally, we solve Equation 8 using all matrices computed above to get the exact bit precision settings of the model for the given target size S_{target}. As a solution, we get modified configuration matrix $\mathrm{C}_{L\times D}$ where each row corresponds to the specific block contains only one entry set to one which corresponds to the column that represent the number of precision bits used for mixed-precision quantization.

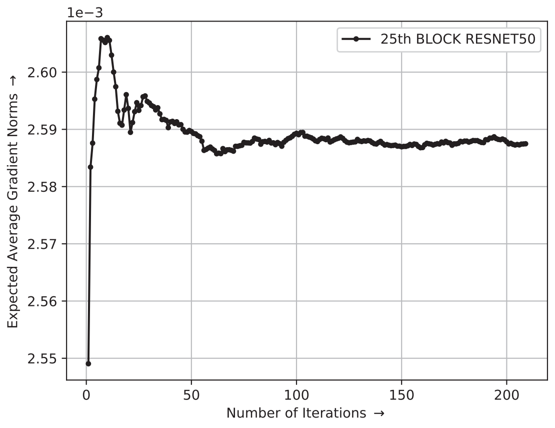
4. Experiments and Results

We use variant of adaptive rounding [19] for post training quantization which is computationally efficient, does not require retraining of the network, and only uses a small amount of unlabelled data. We minimize the following loss function for adaptive rounding:

$$\arg\max_{\mathbf{V}} \alpha \left\| f_a(\mathbf{Wx}) - f_a(\widetilde{\mathbf{W}}\hat{\mathbf{x}}) \right\|_F^2 + \eta \left\| \mathbf{W} - \widetilde{\mathbf{W}} \right\|_2^2 + \lambda f_{reg}(\mathbf{V}) \tag{9}$$

where we set α equals to 2, η equals to 0.25 and rest all the parameters are set to default. Please refer [19] for more details. In all experiments, we use asymmetric per-tensor weight and activation quantization scheme where all logics are implemented in Pytorch framework. We use pretrained models from torchvision-models library for our experiments. In this section, we present the comparative analysis of our method with state of the art quantization methods. We demonstrate that our proposed methodology is fair competitor to methods in both categories which require retraining and which do not. We want to emphasis that for adaptive rounding we use approx 0.002% (i.e. 2048 samples) of the training dataset and same data is used to compute expected average gradient norm. This minimal requirement of data without retraining poses our method very near to zero shot post training quantization methods. For our analysis we have considered benchmark dataset, ImageNet for classification task. At first, we do convergence analysis of our proposed algorithm (Algorithm 1) to compute expected average gradient norm and then show the state-of-the-art quantization results achieved by our method.

Convergence Analysis: We perform convergence analysis of the Algorithm 1 and show that the expected average gradient norm vary very little as we increase the number of iterations over 50 as shown in Figure 3 *(Left)*. We further study the impact of number of images used to compute the expected average gradient norm. We can clearly observe in Figure 3 *(Right)* that expected average gradient norm converges sharply for the images over 1024 which are used to compute the gradient of the loss function in the ϵ-*neighbourhood* of original weights. Based on convergence analysis we have computed the expected average gradient norm shown in Figure 1 *(Left)* for all the blocks of ResNet50 on ImageNet. In addition to the precise bit precision selection using ILPP problem formulation, the Algorithm 1 makes our method much faster. The total time to finish



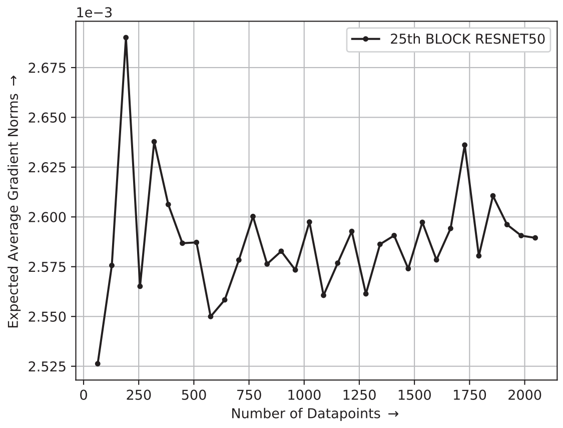


Figure 3: *(Left)* Relationship between the number of iterations used for estimation and expected average gradient norm on block 25 in ResNet50. *(Right)* Relationship between the number of datapoints used and the convergence of expected average gradient norm on block 25 in ResNet50.

the whole bit-width assignment procedure of all blocks of ResNet50 using single CPU take less that 10 seconds to finish.

4.1. ImageNet for classification task

We demonstrate mixed precision post training quantization results on ResNet50, MobileNet-V2 and Inception-V3 for classification and summarize the results in Table 1. Additional results for quantizing ResNet50 on ImageNet is provided in Appendix.

For ResNet50, we compare our results with notable state-of-the-art methods which do not require retraining such as ZEROQ [2] and achieve almost the same accuracy drop (-1.72% vs. -1.64%) with larger compression ratio ($12\times$ vs. $8\times$). At the same time our method perform better than OMSE [6] and ZEROQ [2] with similar compression ratio(Weight CR:$8\times$; Activation CR: $4\times$). In comparison to OCS [25] and DPFMNQ [4], we achieve comparable accuracy with larger compression for both weight and activation. We also compare our result with the methods which require retraining after quantization and achieve almost same accuracy drop for all models except MPQNNCO [3]. This shows that our method can be adopted as an alternative to the methods which require retraining with same performance but with very limited access of the training data. Please refer Table 1 for detailed information.

For Mobilenet-V2, we compare our method with much more efficient and lightweight architecture. For the model which require retraining after quantization, we compare with DC [10] and achieve better performance (-0.49% vs. -0.63%). We also compare with HAQ [8] and achieve similar accuracy drop (-0.49% vs. -0.40%) with the similar compression ratio. MPQNNCO [3] performs better than our method which may be due to very extensive efforts on finetuning as methods of finetuning are not mentioned in the paper [3]. For the model which require retraining after quantization, we compare with DFQ [20], ZEROQ [2] and DPFMNQ [4]. we achieve the significant performance improvement with a similar weight compression ratio.

At last, Inception-V3 is considered for further evaluation. We achieve significant performance improvement when compared with RVQuant [21] which require retraining after quantization. We also achieve comparable performance with other methods like HAWQ [8], IntOnly [13] and HAWQ-V2 [7] in the same category (requires retraining). When compare to the notable methods which do not require retraining like OCS [25], we achieve much smaller accuracy drop(-1.30% vs. -4.60%) which larger weight compression ratio ($8\times$ vs. $5.33\times$). We also achieve almost the same accuracy drop (-1.30% vs. -1.31%) when compared to ZEROQ [2] which shows the applicability of our method in general.

5. Conclusion

In this work, we present a novel mixed-precision post training quantization method which requires very limited access of training data with no retraining required after quantization. To measure the sensitivity of the layers against perturbation we have used only first-order information. As per our knowledge, this is the first attempt to measure the sensitivity of the layers using first-order information only. We have modelled the relative change in loss which is proportional to the product of L_2 norm of the quantization perturbation and the expected average gradient norm. Furthermore, we formulate the mixed-precision quantization problem as Integer linear programming prob-

Table 1: Comparison with state-of-the-art methods on ImageNet. 'RT' refers whether retraining of network is required or not. 'W Bits' and 'A Bits' stands for quantization bits used for weights and activations, respectively. The 'W CR' and 'A CR' stands for weight and activation compression ratio, respectively. The 'MP' refers to mixed-precision quantization, where we report the lowest bits used for weights and activations. We use '-' where data is not provided in the manuscript.

Network	Method	RT	Top-1 Full	W Bit	A Bit	W CR	A CR	Top-1 Quant	Top-1 Drop
ResNet-50	PACT* [5]	✓	76.90	2	4	7.24 ×	7.99 ×	74.50	-2.40
	AutoQ [17]	✓	74.80	MP	MP	10.26 ×	7.96 ×	72.51	-2.29
	HAWQ [8]	✓	77.39	2_{MP}	4_{MP}	12.28 ×	-	75.48	-1.91
	HAWQ-V2 [7]	✓	77.39	2_{MP}	4_{MP}	12.24 ×	-	75.76	-1.63
	HAQ [23]	✓	76.15	MP	32	10.57 ×	1.00 ×	75.30	-0.85
	MPQNNCO [3]	✓	76.13	2_{MP}	4_{MP}	12.24 ×	8.00 ×	75.28	-0.85
	OMSE [6]	✗	77.72	4	32	8 ×	1 ×	70.06	-7.66
	ZEROQ [2]	✗	77.72	MP	8	8 ×	4 ×	76.08	-1.64
	DFPNMQ [4]	✗	76.13	MP	32	6.43 ×	1 ×	75.32	-0.81
	OCS [25]	✗	76.10	8	8	4 ×	4 ×	75.7	-0.40
	Ours	✗	**76.13**	2_{MP}	**8**	**8** ×	**4** ×	**75.44**	**-0.69**
	Ours	✗	**76.13**	2_{MP}	**8**	**12** ×	**4** ×	**74.41**	**-1.72**
MobileNet-V2	DC [10]	✓	71.87	MP	32	7.47 ×	1.00 ×	71.24	-0.63
	HAQ [23]	✓	71.87	MP	32	7.47 ×	1.00 ×	71.47	-0.40
	MPQNNCO [3]	✓	71.88	3_{MP}	8	7.49 ×	4.00 ×	71.83	-0.05
	DFQ [20]	✗	73.03	8	8	4 ×	4 ×	71.20	-1.83
	ZEROQ [2]	✗	73.03	MP	8	8 ×	4 ×	69.44	-3.59
	DFPNMQ [4]	✗	71.88	MP	32	6.32 ×	1 ×	70.35	-1.53
	ZEROQ [2]	✗	73.03	8	8	4 ×	4 ×	72.91	-0.12
	Ours	✗	**71.87**	2_{MP}	**16**	**4** ×	**2** ×	**71.88**	**+0.01**
	Ours	✗	**71.87**	2_{MP}	**16**	**8** ×	**2** ×	**71.38**	**-0.49**
Inception-V3	RVQuant [21]	✓	78.88	8	8	4 ×	4 ×	74.22	-4.66
	HAWQ [8]	✓	77.45	2_{MP}	4_{MP}	12.04 ×	-	75.52	-1.93
	IntOnly [13]	✓	77.45	8	8	4.00 ×	4.00 ×	75.40	-2.05
	HAWQ-V2 [7]	✓	77.45	2_{MP}	4_{MP}	12.04 ×	-	75.68	-1.77
	OCS [25]	✗	75.90	6	6	5.33 ×	5.33 ×	71.30	-4.60
	ZEROQ [2]	✗	78.88	MP	8	8 ×	4 ×	77.57	-1.31
	Ours	✗	**76.10**	2_{MP}	**8**	**8** ×	**4** ×	**74.75**	**-1.30**
	Ours	✗	**76.10**	8_{MP}	**8**	**4** ×	**4** ×	**75.97**	**-0.13**

* do not quantize the first and last layer

lem to decide the precise bit settings of each layer in the network. We presented state-of-the-art results on image classification for ResNet18, ResNet50, and Inception-V3. We demonstrate that using our proposed method for post training mixed-precision quantization, we achieve better results than several state-of-the-art methods in both categories which require retraining of the model with full training dataset and which do not require retraining model.

References

[1] Ron Banner, Yury Nahshan, and Daniel Soudry. Post training 4-bit quantization of convolutional networks for rapid-deployment. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7948–7956, 2019.

[2] Yaohui Cai, Zhewei Yao, Zhen Dong, Amir Gholami,

Michael W. Mahoney, and Kurt Keutzer. Zeroq: A novel zero shot quantization framework. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 13166–13175. Computer Vision Foundation / IEEE, 2020.

[3] Weihan Chen, Peisong Wang, and Jian Cheng. Towards mixed-precision quantization of neural networks via constrained optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5350–5359, October 2021.

[4] Vladimir Chikin and Mikhail Antiukh. Data-free network compression via parametric non-uniform mixed precision quantization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 450–459. IEEE, 2022.

[5] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. PACT: parameterized clipping activation for quantized neural networks. *CoRR*, abs/1805.06085, 2018.

[6] Yoni Choukroun, Eli Kravchik, Fan Yang, and Pavel Kisilev. Low-bit quantization of neural networks for efficient inference. In *2019 IEEE/CVF International Conference on Computer Vision Workshops, ICCV Workshops 2019, Seoul, Korea (South), October 27-28, 2019*, pages 3009–3018. IEEE, 2019.

[7] Zhen Dong, Zhewei Yao, Daiyaan Arfeen, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Hawq-v2: Hessian aware trace-weighted quantization of neural networks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18518–18529. Curran Associates, Inc., 2020.

[8] Z. Dong, Z. Yao, A. Gholami, M. Mahoney, and K. Keutzer. Hawq: Hessian aware quantization of neural networks with mixed-precision. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 293–302, Los Alamitos, CA, USA, nov 2019. IEEE Computer Society.

[9] Amir Gholami, Sehoon Kim, Zhen Dong, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. A survey of quantization methods for efficient neural network inference. *CoRR*, abs/2103.13630, 2021.

[10] Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[11] Matan Haroush, Itay Hubara, Elad Hoffer, and Daniel Soudry. The knowledge within: Methods for data-free model compression. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 8491–8499. Computer Vision Foundation / IEEE, 2020.

[12] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015.

[13] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew G. Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 2704–2713. Computer Vision Foundation / IEEE Computer Society, 2018.

[14] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[15] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *ICCV*, pages 2755–2763. IEEE Computer Society, 2017.

[16] Zechun Liu, Haoyuan Mu, Xiangyu Zhang, Zichao Guo, Xin Yang, Kwang-Ting Cheng, and Jian Sun. Metapruning: Meta learning for automatic neural network channel pruning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[17] Qian Lou, Feng Guo, Minje Kim, Lantao Liu, and Lei Jiang. Autoq: Automated kernel-wise neural network quantization. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[18] Asit K. Mishra and Debbie Marr. Apprentice: Using knowledge distillation techniques to improve low-precision network accuracy. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[19] Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? Adaptive rounding for post-training quantization. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 7197–7206. PMLR, 13–18 Jul 2020.

[20] Markus Nagel, Mart van Baalen, Tijmen Blankevoort, and Max Welling. Data-free quantization through weight equalization and bias correction. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 1325–1334. IEEE, 2019.

[21] Eunhyeok Park, Sungjoo Yoo, and Peter Vajda. Value-aware quantization for training and inference of neural networks. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part IV*, volume 11208 of *Lecture Notes in Computer Science*, pages 608–624. Springer, 2018.

[22] Mariam Rakka, Mohammed E. Fouda, Pramod Khargonekar, and Fadi Kurdahi. Mixed-precision neural networks: A survey, 2022.

[23] Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. HAQ: hardware-aware automated quantization with mixed precision. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 8612–8620. Computer Vision Foundation / IEEE, 2019.

[24] Dongqing Zhang, Jiaolong Yang, Dongqiangzi Ye, and Gang Hua. Lq-nets: Learned quantization for highly accurate and compact deep neural networks. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VIII*, volume 11212 of *Lecture Notes in Computer Science*, pages 373–390. Springer, 2018.

[25] Ritchie Zhao, Yuwei Hu, Jordan Dotzel, Christopher De Sa, and Zhiru Zhang. Improving neural network quantization without retraining using outlier channel splitting. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 7543–7552. PMLR, 2019.

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