



# Semilayer-Wise Partial Quantization Without Accuracy Degradation or Back Propagation

Matsuda, Tomoya  
Matsumoto, Kengo  
Inoue, Atsuki  
Kawaguchi, Hiroshi  
Sakai, Yasufumi

---

**(Citation)**

Artificial Neural Networks and Machine Learning - ICANN 2023:283-295

**(Issue Date)**

2023-09-23

**(Resource Type)**

conference paper

**(Version)**

Accepted Manuscript

**(Rights)**

© 2023 The Author(s), under exclusive license to Springer Nature Switzerland AG

**(URL)**

<https://hdl.handle.net/20.500.14094/0100491826>



# Semilayer-Wise Partial Quantization without Accuracy Degradation or Back Propagation

Tomoya Matsuda<sup>1</sup>, Kengo Matsumoto<sup>1</sup>, Atsuki Inoue<sup>1</sup>,  
Hiroshi Kawaguchi<sup>1</sup>, and Yasufumi Sakai<sup>2</sup>

<sup>1</sup> Graduate School of Science, Technology and Innovation, Kobe University,  
1-1 Rokkoudai, Nada, Kobe, Hyogo 657-8501, Japan

<sup>2</sup> Fujitsu Research, Fujitsu Limited,  
4-1-1 Kamikodanaka, Nakahara, Kawasaki, Kanagawa 211-8588, Japan  
matsuda.tomoya@cs28.cs.kobe-u.ac.jp,  
matsumoto.kengo@cs28.cs.kobe-u.ac.jp,  
ainoue@godzilla.kobe-u.ac.jp,  
kawapy@godzilla.kobe-u.ac.jp,  
sakaiyasufumi@fujitsu.com

**Abstract.** In edge AI technologies, reducing memory bandwidth and computational complexity without reducing inference accuracy is a key challenge. To address this difficulty, partial quantization has been proposed to reduce the number of bits in weight parameters of neural network models. However, existing techniques monotonically degrade accuracy with the compression ratio without retraining. In this paper, we propose an algorithm for semilayer-wise partial quantization without accuracy degradation or back-propagation retraining. Each layer is divided into two channel groups (semilayers): one being positive for loss degradation and the other negative. Each channel is classified as positive or negative in terms of cross-entropy loss and assigned to a semilayer accordingly. The evaluation was performed with validation data as input. Then, the quantization priority for every semilayer is determined based on the magnitude in the Kullback-Leibler divergence of the softmax output before and after quantization. We observed that ResNet models achieved no degradation in accuracy at certain parameter compression ratios (i.e., 79.43%, 78.01%, and 81.13% for ResNet-18, ResNet-34, and ResNet-50, respectively) in partial 6-bit quantization on classification tasks using the ImageNet dataset.

**Keywords:** Partial Quantization, Sensitivity Analysis, Image Classification

## 1 Introduction

Compression methods such as quantization [21], which reduces the number of bits, have been proposed to reduce the size and computational costs of neural network models. Quantization is an effective method of compressing learning models because it can reduce their size, memory requirements, and computational cost. Nonetheless, there is a tradeoff relation between the compression ratio and the accuracy of the

compressed model; that is, quantization degrades accuracy. In previous studies on full quantization of neural networks, models were uniformly quantized with the same bit width [8, 16]. However, differing distributions of weights in each layer have been shown to exhibit different impacts on accuracy in quantization. Therefore, partial quantization, in which quantization is selectively performed for some parts of a model (e.g., layers), can be used to make a tradeoff between accuracy and size. Sensitivity analysis methods [20] have been proposed to answer the question of which layers and channels should be quantized; either accuracy or loss are used as measures of sensitivity. Layer-wise [18, 19] and channel-wise quantization [2, 11, 15] methods have been used to prioritize network regions where quantization should be performed. In these methods, quantization is performed on a layer-by-layer or channel-by-channel basis. Layer-wise quantization is more compatible with edge AI and is easy to handle for hardware owing to its larger granularity. In some earlier studies [7, 15, 18, 19], learning models were quantized without retraining. Naturally, quantized models must be created in a practical computation time. This approach also prevents the possible degradation of generalization performance owing to retraining. However, models quantized using these existing methods cannot be compressed sufficiently while maintaining accuracy.

In this study, to achieve compression of neural models while suppressing accuracy degradation resulting from quantization, we propose a new layer-wise partial quantization method. The proposed method examines the  $\Delta loss$  of the entire model when quantizing only one channel for all convolutional layers of a pretrained neural network. The evaluation was performed with validation data as input. It then divides each convolutional layer into two channel groups according to the positive and negative values of  $\Delta loss$  for all channels. These are referred to as “semilayers”. The proposed method quantizes the model on a semilayer basis. We also incorporate Kullback-Leibler (KL) divergence in the sensitivity analysis. In fact,  $\Delta loss$  includes more information than accuracy. Moreover, it is effective in terms of sensitivity [15]. However, the change compared to the original model cannot be considered by  $\Delta loss$ , whereas the KL divergence measures changes between models and tends to exhibit a large value when the absolute value of  $\Delta loss$  is large. Thus, introducing KL divergence as a measure of sensitivity is effective in suppressing large model changes due to compression.

To evaluate the performance of the proposed method, we conducted experiments on standard image classification tasks using the ImageNet dataset [3] with various ResNet models [24] and the CIFAR-10 dataset [22] with a VGG-16-bn model [25]. The contributions of this study are summarized as follows:

- We propose a new quantization granularity for convolutional neural network (CNN) models, which considers “semilayers” as an alternative to layers and channels. A positive semilayer has positive channels in  $\Delta loss$ . A negative semilayer contains negative channels in  $\Delta loss$ . Consequently, the accuracy of this approach can be improved after priority quantization of the negative semilayers.
- For the sensitivity analysis, we consider KL divergence for each semilayer. Introducing KL divergence improves the tradeoff between accuracy and compression.

- We observed that the proposed semilayer-wise partial quantization exhibited maximum accuracy greater than that of baseline models, and that it caused no accuracy degradation at certain compression ratios in the image classification tasks of the ImageNet and CIFAR-10 datasets. For example, the proposed method maintained a 76.12% accuracy at a compression ratio of 81.13% for ResNet-50.

## 2 Related Work

Prior works have proposed a unified framework for CNNs, referred to as quantized CNNs (Q-CNNs) [21], which simultaneously accelerates and compresses CNN models with only slight performance degradation. The results indicate that Q-CNN models can perform especially fast computation in the testing phase and that they significantly reduce storage and memory requirements. Specifically, this approach can achieve 4–6× speedup and compress a model by a factor of 15–20 while decreasing classification accuracy by less than 1%. Furthermore, the Q-CNN models can be implemented on mobile devices and have been shown to classify images in less than one second. Quantization has also been widely adopted to assess compression methods. An earlier report [11] described that introducing channel-wise quantization instead of layer-wise quantization could reduce the degradation in accuracy after 8-bit quantization without finetuning. One study [2] indicated that a channel-wise quantization scheme that minimized the mean square error was effective for 4-bit quantization. In addition, a quantization step size designed to minimize cross-entropy loss has been used [14]. Another report [13] proposed the use of approximate loss functions to optimize rounding. As another approach, one study proposed performing partial quantization based on individually investigated layer sensitivities before employing quantization aware learning [20]. Reportedly,  $\Delta loss$  analysis (DLA) [15], in which quantization is selectively performed depending on the parts of the model (e.g., layers), is useful to make a tradeoff between accuracy and model size. Along these lines, a sensitivity search method has also been proposed based on the idea that accuracy can be improved using  $\Delta loss$ , especially at the channel level of each convolutional layer. In other studies [18, 19], a deterministic greedy search algorithm (GSA) inspired by submodular optimization was used to derive a practical solution to the bit assignment problem without retraining. Another approach proposed a mixed hardware-friendly quantization (MXQN) method [7] that applies fixed-point quantization and logarithmic quantization without finetuning deep CNNs. Constrained-optimization-based algorithm for mixed-precision quantization (CQ) exploited Hessians [1], but it required retraining. In the present work, our proposed method allows sufficient quantization of the trained model without accuracy degradation or back-propagation retraining.

## 3 Proposed Method

For simplicity, the proposed method is described in this section using the case of a ResNet-18 model for the ImageNet dataset. We consider quantization for weights only and not for activation.

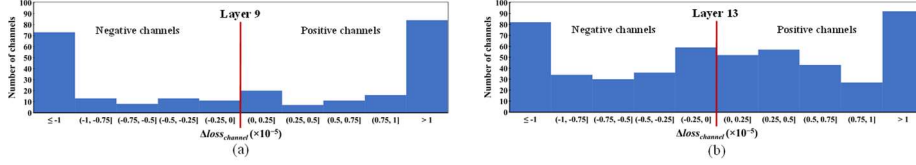


Fig. 1. Histograms of  $\Delta loss_{channel}$  for all channels in layers (a) 9 and (b) 13 at 6-bit quantization.

### 3.1 Semilayer-Wise Quantization Using $\Delta loss$ per Channel

The ResNet-18 network has 16 convolutional layers. Here, we refer to these as layers 1, 2, ..., 16, starting from the layer closest to the input. First, a pretrained model to be quantized<sup>1</sup> is prepared. Next, we perform a validation test on the trained model to check its baseline cross-entropy  $loss_{before}$  as the classification error. The evaluation using ImageNet classification was performed with 50-k validation data as input. Then, only one of the first channels of the first layer is quantized, and a validation test is performed to obtain its cross-entropy loss  $loss_{after-1,1}$ . Similarly, after  $loss_{after-i,j}$  is calculated for the  $i$ -th layer and the  $j$ -th channel, the difference between the loss functions  $\Delta loss_{i,j}$  is calculated for every layer and every channel as

$$\Delta loss_{i,j} = loss_{after-i,j} - loss_{before} . \quad (1)$$

Some examples of the channel-wise distribution of  $\Delta loss_{channel}$  for ImageNet classification when 6-bit quantization is applied in ResNet-18 are shown in Fig. 1. The horizontal axis represents the  $\Delta loss_{channel}$  bins. Almost half of the channels have negative values in  $\Delta loss_{channel}$ . This may indicate that quantization of half of a pretrained model could improve its accuracy.

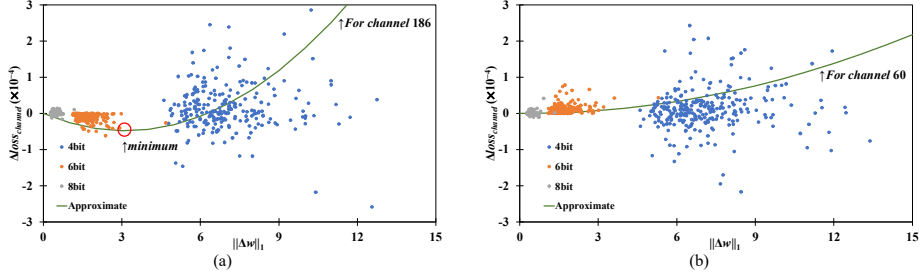
In quantization, a set of weights  $\mathbf{w}$  are shifted by a quantization error  $\Delta \mathbf{w}$ . In channel-wise quantization, the weights and quantization errors respectively have vectors with channel sizes  $\mathbf{w}$  and  $\Delta \mathbf{w}$ . In this case,  $\Delta loss$  is approximated as a quadratic function [13]

$$\Delta loss \approx \Delta \mathbf{w}^T \cdot \mathbf{g} + 1/2 \Delta \mathbf{w}^T \cdot \mathbf{H} \cdot \Delta \mathbf{w} + \dots , \quad (2)$$

where  $\mathbf{g}$  and  $\mathbf{H}$  denote the gradients and Hessians, respectively. Fig. 2 shows the relationships between  $\|\Delta \mathbf{w}\|_1$  (an L1 norm for weights) and  $\Delta loss$ . Some channels apparently behave as quadratic functions; this phenomenon has been analyzed in [13]. In particular, some channels in the negative semilayer seem to have a minimum point in  $\Delta loss$ . That is, the 6-bit quantization is superior to others in this case. Therefore, we mainly chose 6-bit quantization.

We propose the division of each layer into two channel groups (semilayers) according to the positive and negative values of channel-wise  $\Delta loss$ . For example, if  $\Delta loss$  of the first channel of layer 1 is negative, then it is assigned to the semilayer

<sup>1</sup> Note that the quantization method is the same approach as the ‘‘qint’’ cast in PyTorch [6].



**Fig. 2.** Scattering plots of  $\|\Delta w\|_1$  and  $\Delta loss_{channel}$  for channels in layer 13 of a ResNet-18 model. (a) only channels with negative  $\Delta loss_{channel}$  for 6-bit quantization and corresponding points for 4-bit and 8-bit quantizations are shown. (b) only channels with positive  $\Delta loss_{channel}$  for 6-bit quantization and corresponding points for 4-bit and 8-bit quantizations are shown.

1-negative. The positive channel of layer 1 is assigned to the semilayer 1-positive. This process is performed for all channels in all layers. The purpose of this operation is to divide each layer into channels for which quantization is expected to improve accuracy and channels for which there is some possibility of decreasing accuracy.

### 3.2 KL Divergence in Sensitivity Analysis

We calculated the KL divergence for all semilayers. The KL divergence  $D_{KL}$  for a semilayer is calculated as follows.

$$D_{KL}(P||Q) = \sum_{x \in \mathcal{X}} P(x) \log \frac{P(x)}{Q(x)}, \quad (3)$$

where  $x$  is one of output,  $\mathcal{X}$  is number of classifications (1000 for ImageNet classifications),  $P(\cdot)$  denotes the softmax output of the pretrained model, and  $Q(\cdot)$  denotes the softmax output of the model after quantization of all channels in a semilayer. The evaluation using ImageNet classification was performed with 50-k validation data as input. As described herein, KL divergence for each semilayer was calculated for all 50-k inputs individually. The average value of the 50-k data was taken as the KL divergence after quantization for that semilayer. This KL divergence was calculated for each of the 32 semilayers in ResNet-18 with 16 convolutional layers.

In the sensitivity analysis, KL divergence obtained using the operation above is normalized by the number of parameters in the semi-layer. In the proposed approach, KL divergence normalization is calculated as  $KL_{parameterized}$

$$KL_{parameterized} = \frac{D_{KL}(P||Q)}{param(X)}, \quad (4)$$

where  $param(\cdot)$  represents the number of parameters in the semilayer. In the case of a process that quantizes multiple parameters together, such as layer-wise quantization, a greater number of parameters quantized is associated with a greater effect on the output of the neural network. Therefore, to consider the number of parameters in a semilayer, we normalize KL divergence in Equation (3) by the number of parameters in a given semilayer.

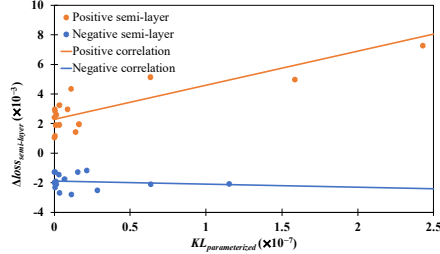


Fig. 3. Correlation between  $KL_{parameterized}$  and  $\Delta loss_{semilayer}$  with semilayers quantized.

Actually, in sensitivity analysis, it might be seen that the  $\Delta loss$  is more informative and effective for partial quantization [15]. However, the degree of change from the original model is not considered in  $\Delta loss$  when it is used as a sensitivity. That is, a semi-layer with larger absolute values of the  $\Delta loss$  ( $= \Delta loss_{semilayer}$ ) might incur greater model change. The large changes in weight prevent continuous compression without accuracy degradation. Thus, we adopt the KL divergence to limit changes in the model or its weights to moderate levels. KL divergence tends to be smaller when the absolute values of the  $\Delta loss_{semilayer}$  are smaller. Fig. 3 shows the correlation between  $KL_{parameterized}$  and  $\Delta loss_{semilayer}$  for 6-bit quantization in ResNet-18 when each semilayer was quantized separately. The correlation coefficient of the negative semi-layer was  $-0.129$ . That of the positive semi-layer was  $0.834$ . The figure shows that the absolute values of the  $\Delta loss_{semilayer}$  with larger  $KL_{parameterized}$  were larger. In fact, a more negative  $\Delta loss_{semilayer}$  indicates good performance in terms of loss, but it is not effective as a measure of sensitivity owing to the large changes in the model. To suppress this shortcoming, the KL divergence is prioritized for the evaluation index.

### 3.3 Postponing Strategy after Sensitivity Analysis

For the  $KL_{parameterized}$  obtained in the preceding subsection, all semilayers are quantized in order of decreasing value according to the sensitivity analysis findings. As an exception, we introduce “postponing” only once for each semilayer. This process ensures that the accuracy is improved and contributes to the efficiency of parameter compression. To perform this process within a practical computational time, we use semilayers, which is not a channel-wise approach, but rather is closer to layer-wise. In all, 32 semilayer-wise quantizations are performed on ResNet-18 models in order of decreasing  $KL_{parameterized}$ . In the postponing strategy, however, if the accuracy is lower than the state before semilayer-wise quantization, then the semilayer is not quantized at this time step, and is instead postponed. The next semilayer is examined and inference performed similarly, starting in the original sorted order. When all sorted semilayers have been examined, postponing the first trial is completed. An improvement in accuracy is expected in the first trial.

Next, the remaining semilayers that have not yet been quantized are quantized and inferred again as the second trial, according to the sensitivity analysis. The postponing strategy is not adopted in the second trial, where accuracy degradation is expected.

Based on this tendency, by improving the accuracy in the first trial and slightly decreasing the accuracy in the second trial, accuracy and the parameter compression ratio are maximally improved without any reduction in accuracy, as shown by the results of the experiments described below. The pseudocode of the algorithm for the proposed semilayer-wise quantization is shown in Algorithm 1.

---

**Algorithm 1:** Proposed method
 

---

**Input:** Pre-trained FP32 model.

```

1: for  $i = 1, 2, \dots, I, j = 1, 2, \dots, J$  do
2:    $\Delta loss_{i,j} = loss_{after-i,j} - loss_{before}$  (1)
3: end for
4: for  $i = 1, 2, \dots, I$  do
5:   semilayer_i-negative  $\leftarrow$  Index of channel with negative  $\Delta loss_{i,j}$ 
6:   semilayer_i-positive  $\leftarrow$  Index of channel with positive  $\Delta loss_{i,j}$ 
7:   semilayers  $\leftarrow$  semilayer_i-negative, semilayer_i-positive
8: end for
9: for semilayers do
10:   $D_{KL}(P||Q) = \sum_{x \in \mathcal{X}} P(x) \log\left(\frac{P(x)}{Q(x)}\right)$  (3)
11:   $KL_{parameterized} = \frac{D_{KL}(P||Q)}{param(\mathbf{X})}$  (4)
12: end for
13: sorted_semilayers  $\leftarrow$  Index of  $KL_{parameterized}$  sorted in decreasing order
14: for  $s \in$  sorted_semilayers do
15:   Quantize  $s$  and Inference model.
16:   if the accuracy deteriorates when  $s$  is quantized
17:     Postpone quantization of  $s$  that was quantized.
18:     skipped_semilayers  $\leftarrow$   $s$ 
19:   end if
20: end for
21: for  $s \in$  skipped_semilayers do
22:   Quantize  $s$  and Inference model.
23: end for
Output: Compressed model.

```

---

## 4 Experiments

### 4.1 ResNets for ImageNet

We used ResNet-18, ResNet-34, and ResNet-50 [24] in the experimental evaluation of the proposed approach. Fig. 4 shows the results obtained from 6-bit and 4-bit quantizations using a pretrained ResNet-18 model as the ImageNet dataset. Our method compressed the number of parameters of the model by 79.43% in 6-bit quantization and by 33.82% for 4-bit quantization without degradation of accuracy.

Fig. 5 shows other results for the 6-bit and 4-bit quantizations on ResNet-34. Our method compressed the number of parameters by 78.01% for 6-bit quantization and by 38.05% compression for 4-bit quantization without degradation of accuracy. On ResNet-50, 6-bit quantization compressed the number of parameters by 81.13%, as shown in Fig. 6.

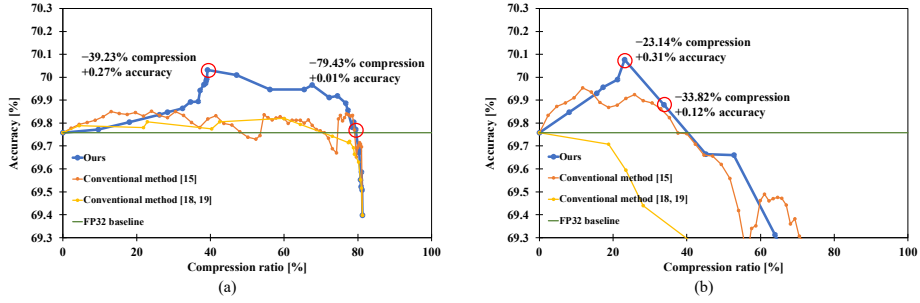


Fig. 4. Accuracies in (a) 6-bit quantization and (b) 4-bit quantization of a ResNet-18 model.

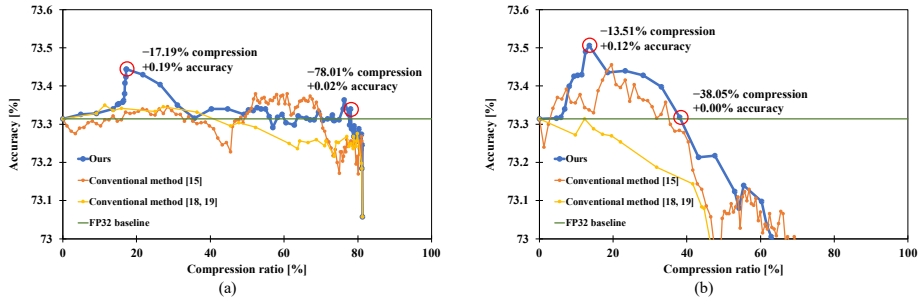


Fig. 5. Accuracies in (a) 6-bit quantization and (b) 4-bit quantization of a ResNet-34 model.

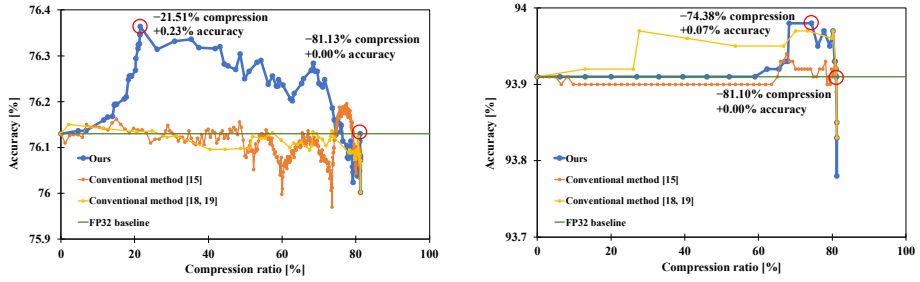


Fig. 6. Accuracies in 6-bit quantization of a ResNet-50 model.

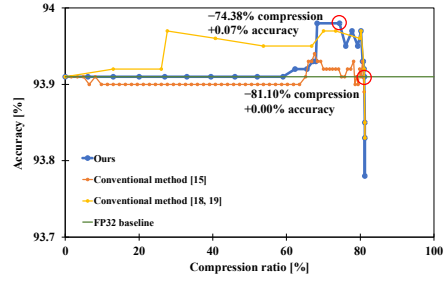


Fig. 7. Accuracies in 6-bit quantization on a VGG-16-bn model for CIFAR-10 dataset.

## 4.2 VGG-16-bn for CIFAR-10

Fig. 7 presents the results obtained for 6-bit quantization using VGG-16-bn as the trained model and a classification task on the CIFAR-10 evaluation dataset. Our method achieved 81.10% parameter compression with 6-bit quantization without accuracy degradation.

**Table 1.** Results for ImageNet dataset. The “*b*MP” refers to mixed-precision quantization, where *b* is the lowest bits used for weights.

Network	Method	Baseline	Quantization bit	Compression ratio	Top-1 / quant	Top-1 / drop	Retraining
ResNet-18	GSA [18,19]	69.75%	4	±00.00%	69.75%	±0.00%	No
	GSA [18,19]	69.75%	6	-64.34%	69.79%	+0.04%	No
	DLA [15]	69.75%	4	-35.23%	69.82%	+0.07%	No
	DLA [15]	69.75%	6	-79.06%	69.77%	+0.02%	No
	MXQN [7]	69.75%	8	-75.00%	67.61%	-2.14%	No
	CQ [1]	69.75%	3MP	-87.98%	69.66%	-0.09%	Yes
	CQ [1]	69.75%	2MP	-90.61%	69.39%	-0.36%	Yes
	Ours	69.75%	4	-33.82%	69.88%	+0.13%	No
<b>Ours</b>	<b>69.75%</b>	<b>6</b>	<b>-79.43%</b>	<b>69.77%</b>	<b>+0.02%</b>	<b>No</b>	
ResNet-34	GSA [18,19]	73.31%	4	-12.23%	73.31%	±0.00%	No
	GSA [18,19]	73.31%	6	-36.36%	73.33%	+0.02%	No
	DLA [15]	73.31%	4	-34.26%	73.35%	+0.04%	No
	DLA [15]	73.31%	6	-69.88%	73.32%	+0.01%	No
	MXQN [7]	73.31%	8	-75.00%	71.43%	-1.88%	No
	Ours	73.31%	4	-38.05%	73.31%	±0.00%	No
	<b>Ours</b>	<b>73.31%</b>	<b>6</b>	<b>-78.01%</b>	<b>73.34%</b>	<b>+0.03%</b>	<b>No</b>
	ResNet-50	GSA [18,19]	76.12%	6	-73.45%	76.13%	+0.01%
DLA [15]		76.12%	6	-79.36%	76.13%	+0.01%	No
MXQN [7]		76.12%	8	-75.00%	74.06%	-2.06%	No
CQ [1]		76.12%	2MP	-91.83%	75.28%	+0.16%	Yes
<b>Ours</b>		<b>76.12%</b>	<b>6</b>	<b>-81.13%</b>	<b>76.12%</b>	<b>±0.00%</b>	<b>No</b>
VGG-16-bn	GSA [18,19]	93.90%	6	-81.03%	93.91%	+0.01%	No
	DLA [15]	93.90%	6	-80.83%	93.90%	±0.00%	No
	<b>Ours</b>	<b>93.90%</b>	<b>6</b>	<b>-81.10%</b>	<b>93.90%</b>	<b>±0.00%</b>	<b>No</b>

## 4.3 Comparison with Other Methods

Table 1 presents our experimental results. Compared with conventional methods, the proposed method provided sufficient compression without back-propagation retraining or accuracy degradation. The GSA and The DLA were ineffective because they achieved lower parameter compression values than the proposed method did. Mixed hardware-friendly quantization (MXQN) [7] and constrained optimization-based algorithm for mixed-precision quantization (CQ) [1] achieved a parameter compression ratio of 75.0% and 91.83%, respectively, in ResNet-50; especially, CQ was superior in terms of compression ratio. However, CQ was impractical because it required retraining and degraded the accuracy compared to the baseline.

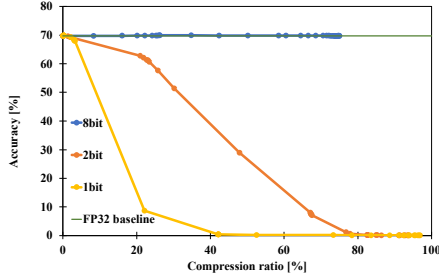


Fig. 8. Accuracies in 8-bit, 2-bit, and 1-bit quantizations.

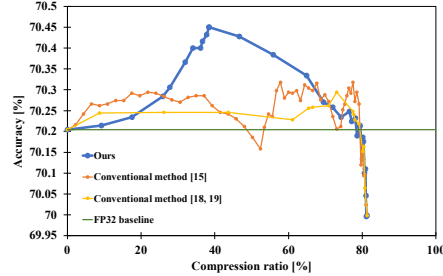


Fig. 9. Accuracies in 6-bit quantizations on 10 epochs trained ResNet-18.

#### 4.4 Quantization with More and Fewer Bits

Here, we also provide experimental results obtained from 8-bit, 2-bit, and 1-bit quantizations of a ResNet-18 model. Fig. 8 shows the results for 8-bit, 2-bit, and 1-bit quantization for the pretrained ResNet-18 model performing a classification task on the ImageNet evaluation dataset. The proposed method is not suitable for 8-bit quantization because the maximum compression ratio by bit reduction is lower than that of 6-bit quantization that can compress parameters efficiently. The 8-bit quantization is near the original pretrained model and far from the minimum point in  $\Delta loss$  (see Fig. 2(a)). In the 2-bit and 1-bit quantizations,  $\Delta loss$  seems to have reached a massive value, which cannot maintain accuracy.

#### 4.5 Experiment with Another Trained Model

We ran the same experiment on another trained model under varying conditions to demonstrate the reproducibility of the proposed method. We experimented with a publicly available ResNet-18 model [24] that was retrained for 10 epochs by 1.2-M training data. Fig. 9 shows the results obtained from 6-bit quantization using the retrained ResNet-18 model; we observed that the baseline was improved by 0.45% (compare to Fig. 4(a)). Even in such case, our method achieved 79.40% parameter compression for 6-bit quantization without accuracy degradation. These results show that the proposed method can provide efficient compression for models with different levels of training.

## 5 Conclusions

In this paper, we proposed a new method for the quantization of trained neural networks. This method enables more efficient compression without reduction of accuracy. The proposed method is a semilayer-wise quantization in which each layer is classified into two channel groups according to  $\Delta loss$ , in contrast to layer-wise quantization. The results of the experimental evaluation of the proposed method on classification tasks with the ImageNet dataset using various trained neural networks have

shown that it can improve the tradeoff between reducing the size of a model and degrading its accuracy. Specifically, using a trained network as a starting point, the proposed method successfully compressed the number of parameters of a ResNet-18 model through 6-bit quantization by 79.43% without accuracy degradation, and by 78.01% and 81.13% for a ResNet-34 model with 6-bit quantization and a ResNet-50 model with 6-bit quantization, respectively.

## References

1. Chen, W., Wang, P., Cheng, J.: Towards Mixed-Precision Quantization of Neural Networks via Constrained Optimization. In: IEEE/CVF International Conference on Computer Vision (ICCV), pp. 5350–5359 (2021).
2. Choukroun, Y., Kravchik, E., Yang, F., Kisilev, P.: Low-bit Quantization of Neural Networks for Efficient Inference. arXiv preprint arXiv:1902.06822v2 (2019).
3. Deng, J., Dong, W., Socher, R., Li, L., Li, K., Fei-Fei, L.: Imagenet: A Large-Scale Hierarchical Image Database. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 248-255 (2009).
4. Girshick, R.: Fast R-CNN. In: IEEE International Conference on Computer Vision (ICCV), pp. 1440-1448 (2015).
5. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 580-587 (2014).
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770-778 (2016).
7. Huang, C., Liu, P., Fang, L.: MXQN: Mixed Quantization for Reducing Bit-width of Weights and Activations in Deep Convolutional Neural Networks. In: Applied Intelligence, vol. 51, pp. 1–14 (2021).
8. Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A. G., Adam, H., Kalenichenko, D.: Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2704-2713 (2018).
9. Krizhevsky, A., Sutskever, I., Hinton, G. E.: Imagenet Classification with Deep Convolutional Neural Networks. In: Neural Information Processing Systems (NIPS), pp. 1097-1105 (2012).
10. LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., Jackel, L. D.: Backpropagation Applied to Handwritten Zip Code Recognition. In: Neural Computation, vol. 1, pp. 541-551, (1989).
11. Lee, J. H., Ha, S., Choi, S., Lee, W., Lee, S.: Quantization for Rapid Deployment of Deep Neural Networks. arXiv:1810.05488 (2018).
12. Markidis, S., Chien, S. W. D., Laure, E., Peng, I. B., Vetter, J. S.: Nvidia Tensor Core Programmability, Performance Precision. arXiv:1803.04014v1 (2018).
13. Nagel, M., Amjad, R. A., Baalen, M. V., Louizos, C., Blankevoort, T.: Up or Down? Adaptive Rounding for Post-Training Quantization. In: International Conference on Machine Learning, PMLR 119:7197-7206 (2020).
14. Nahshan, Y., Chmiel, B., Baskin, C., Zheltonozhskii, E., Banner, R., Bronstein, A. M., Mendelson, A.: Loss Aware Post-Training Quantization. In: Machine Learning vol. 110, pp. 3245–3262 (2021).

15. Ohkado, K., Matsumoto, K., Inoue, A., Kawaguchi, H., Sakai, Y.: Channel-Wise Quantization Without Accuracy Degradation Using  $\Delta/loss$  Analysis. In: International Conference on Machine Learning Technologies (ICMLT), pp.56-61 (2022).
16. Shuchang, Z., Yuxin, W., Zekun, N., Xinyu, Z., He, W., Yuheng, Z.: DoReFa-Net: Training Low Bitwidth Convolutional Neural Networks with Low Bitwidth Gradients. arXiv:1606.06160 (2016).
17. Simonyan, K., Zisserman, A.: Very Deep Convolutional Networks for Large-scale Image Recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), arXiv:1409.1556 (2015).
18. Tsuji, S., Yamada, F., Kawaguchi, H., Inoue, A., Sakai, Y.: Greedy search algorithm for partial quantization of convolutional neural networks inspired by submodular optimization. In: 2020 India Intl. Congress on Computational Intelligence (2020).
19. Tsuji, S., Kawaguchi, H., Inoue, A., Sakai, Y., Yamada, F.: Greedy Search Algorithm for Mixed Precision in Post-Training Quantization of Convolutional Neural Network Inspired by Submodular Optimization. In: Asian Conference on Machine Learning, PMLR 157:886-901 (2021).
20. Wu, H., Judd, P., Zhang, X., Isaev, M., Micikevicius, P.: Integer Quantization for Deep Learning Inference: Principles and Empirical Evaluation. arXiv:2004.09602 (2020).
21. Wu, J., Leng, C., Wang, Y., Hu, Q., Cheng, J.: Quantized Convolutional Neural Networks for Mobile Devices. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4820-4828 (2016).
22. CIFAR-10, <https://www.cs.toronto.edu/~kriz/cifar.html>, last accessed 2023/4/19.
23. PyTorch, <https://pytorch.org>, last accessed 2023/4/19.
24. ResNet-18, <https://download.pytorch.org/models/resnet18-5c106cde.pth>, last accessed 2023/4/19.  
ResNet-34, <https://download.pytorch.org/models/resnet34-333f7ec4.pth>, last accessed 2023/4/19.  
ResNet-50, <https://download.pytorch.org/models/resnet50-19c8e357.pth>. last accessed 2023/4/19.
25. VGG-16-bn, [https://pytorch.org/vision/main/models/generated/torchvision.models.vgg16\\_bn.html](https://pytorch.org/vision/main/models/generated/torchvision.models.vgg16_bn.html), last accessed 2023/4/19.