A Mixed-Precision Quantization Method without Accuracy Degradation Using Semilayers

Kengo Matsumoto MATSUMOTO.KENGO@CS28.CS.KOBE-U.AC.JP
Graduate School of Science, Technology and Innovation, Kobe University, 1-1 Rokkoudai, Nada-ku, Kobe, Hyogo 657-8501, Japan

Tomoya Matsuda MATSUDA.TOMOYA@CS28.CS.KOBE-U.AC.JP
Graduate School of Science, Technology and Innovation, Kobe University, 1-1 Rokkoudai, Nada-ku, Kobe, Hyogo 657-8501, Japan

Atsuki Inoue AINOUE@GODZILLA.KOBE-U.AC.JP
Graduate School of Science, Technology and Innovation, Kobe University, 1-1 Rokkoudai, Nada-ku, Kobe, Hyogo 657-8501, Japan

Hiroshi Kawaguchi KAWAPY@GODZILLA.KOBE-U.AC.JP
Graduate School of Science, Technology and Innovation, Kobe University, 1-1 Rokkoudai, Nada-ku, Kobe, Hyogo 657-8501, Japan

Yasufumi Sakai SAKAIYASUFUMI@FUJITSU.COM
Fujitsu Research, Fujitsu Limited, 4-1-1 Kamikodanaka, Nakahara-ku, Kawasaki, Kanagawa 211-8588, Japan

Editors: Berrin Yanıkoğlu and Wray Buntine

Abstract

Reducing the memory usage and computational complexity of high-performance deep neural networks while minimizing degradation of accuracy is a key issue in implementing these models on edge devices. To address this issue, partial quantization methods have been proposed to partially reduce the weight parameters of neural network models. However, the accuracy of existing methods degrades rapidly with increasing compression ratio. Although retraining can compensate for this issue to some extent, it is computationally very expensive. In this study, we propose a mixed-precision quantization algorithm without retraining or degradation in accuracy. In the proposed method, first, the difference between values after and before quantization losses of each channel in the layers of the pretrained model is calculated for all channels. Next, the layers are divided into two groups called semilayers according to whether the loss difference is positive or negative. The priorities for quantization in the semilayers are determined based on the Kulback-Leibler divergence derived from the probability distribution of the softmax output after and before quantization. The same process is repeated as a mixed-precision quantization while gradually decreasing the bitwidth, for example, with 8-, 6-, and 4-bit quantizations, and so forth. The results of an experimental evaluation show that the proposed method successfully compressed a ResNet-18 model by 81.44%, a ResNet-34 model by 84.25%, and a ResNet-50 model by 80.39% on image classification tasks using the ImageNet dataset, and a ResNet-18 model by 80.56% on image classification tasks using the CIFAR-10 dataset, with no degradation of the inference accuracy of the pretrained models.

Keywords: convolutional neural network; mixed-precision quantization; sensitivity analysis; without retraining; without accuracy degradation

1. Introduction

Edge AI describes AI systems embedded in devices operated by users, especially mobile devices. Recently, systems designed to make inferences with neural networks have implemented models on this edge device side to reduce communication with the cloud as much as possible. This approach has attracted attention owing to some notable advantages such as real-time performance and improved security. In general, larger neural network models are required to achieve sufficiently high inference accuracy (Bianco et al. (2018a); Kaplan et al. (2020a)). However, the computational resources of edge devices such as power consumption and memory usage are typically relatively limited. In general, neural network structures become more difficult to implement on simple edge devices. To overcome such problems, neural network models must be sufficiently compressed while minimizing the degradation of the accuracy of inferences. Compression methods such as quantization (Wu et al. (2016)) and pruning (Li et al. (2016)) have been proposed to reduce the size and computational cost of such models. In particular, quantization is an effective compression method because it can flexibly reduce the size of neural network models, as well as their memory usage and computational cost. However, there is a tradeoff between the compression ratio of the model and the accuracy of the compressed model. In other words, there is a limit to the extent that quantization can compress a network because the accuracy decreases as the compression ratio increases.

In previous studies on the quantization of neural networks (Jacob et al. (2018); Wu et al. (2020)), models were uniformly quantized with the same bitwidth. As this approach does not specifically consider important portions in the model, accuracy is easily degraded with quantization to a lower bitwidth. Partial quantization, in which a portion of the model is selectively quantized (e.g., a layer), can resolve the tradeoff between accuracy and compression ratio; sensitivity analysis (Wu et al. (2020)) has been proposed to solve problems with per-layer (Tsuji et al. (2021, 2022)) and per-channel (Choukroun et al. (2019); Lee et al. (2018); Okado et al. (2022)) quantization. In these works, the difference between the accuracy ($\Delta accuracy$) or the loss ($\Delta loss$) of the model after and before quantization was used to represent sensitivity. Furthermore, to resolve the tradeoff between accuracy (or loss) and compression ratio, retraining after quantization is used to improve the compression ratio while maintaining accuracy (Chen et al. (2021)). However, the computational cost involved in retraining neural networks is generally high, and many users may lack abundant computational resources. Therefore, quantization methods that do not require retraining are preferable. Given this background, several previous studies (Huang et al. (2021); Okado et al. (2022); Tsuji et al. (2021, 2022)) have proposed quantization methods without retraining. However, compressing learning models sufficiently while maintaining high inference accuracy remains challenging.

In this study, we propose a mixed-precision quantization method that combines multiple quantization bitwidths to achieve sufficient compression of pretrained neural networks without retraining or degradation of inference accuracy. The quantization target is limited to the weights of the convolutional layer of the neural network. Our proposed method first derives the $\Delta loss$ when the weights of a certain channel are quantized with multiple bitwidths. Then, each layer is divided into two groups, including a positive-value group and a negative-value group based on the $\Delta loss$ value. These groups are denoted as "semilayers" in this study. Because the $\Delta loss$ values differ for each quantization bitwidth, we create multiple semilayers corresponding to each quantization bitwidth. In the proposed method, quantization is performed starting from semilayers with large quantization bitwidth values. Then, the quantization bitwidths are reduced and the compression ratio is improved by the appropriate quantization bitwidth for each channel.

In the proposed approach, we adopt sensitivity analysis as a method of prioritizing quantization. We use the Kullback-Leibler (KL) divergence derived from the probability distribution of the softmax output after and before quantization. As the sensitivity of this process, we take the KL divergence normalized by the number of semilayer parameters. In fact, existing methods have used

$\Delta loss$ as a sensitivity parameter (Okado et al. (2022)). However, these techniques cannot account for changes of network outputs after quantization at an output layer. In contrast, the KL divergence can measure changes after and before quantization. To account for differences in the amount of information due to differences in the number of channels possessed by the each semilayer, the KL divergence is normalized by the number of parameters of the each semilayer. Hence, this sensitivity is effective in controlling large changes in the output of the quantization process.

To evaluate the proposed method, we investigated image classification problems using the ImageNet (Deng et al. (2009)) and CIFAR-10 datasets (Krizhevsky and Geoffrey (2009)) for various ResNet models (He et al. (2016)).

The contributions of this study can be summarized as follows:

- We propose a new quantization granularity referred to as semilayers for the compression of convolutional neural network (CNN) models. In a layer, the positive-value group has positive channels in $\Delta loss$ and the negative-value group has negative channels in $\Delta loss$. As a result, the negative-value groups can improve the accuracy of the model.

- In sensitivity analysis, we propose KL divergence normalized by the number of parameters in each semilayer as sensitivity. This approach can measure changes in network outputs after quantization, preventing large changes in output and thus avoiding degradation of accuracy due to the quantization process.

- We extend semilayer-based quantization to mixed-precision quantization with multiple quantization bitwidths. As a result, the optimal quantization bitwidth can be assigned to each channel.

- Our proposed method achieved a compression ratio of 81.44% for ResNet-18, 84.25% for ResNet-34, and 80.39% for ResNet-50 on image classification tasks using the ImageNet dataset, and 80.56% for ResNet-18 on the image classification tasks using the CIFAR-10 dataset; inference accuracy was not degraded without retraining.

2. Related Work

In recent years, CNN models have been successfully used in various computer applications such as image classification (Simonyan and Zisserman (2014)) and object detection (Girshick et al. (2014); Girshick (2015)). This is mainly due to their deep layered network architecture and the large amounts of training data that can be processed (Krizhevsky et al. (2017)). However, the computational complexity of CNN models increases exponentially for both training and inference for increasingly deep network structures with more layers (Bianco et al. (2018); Kaplan et al. (2020)), which can place a heavy burden on computational resources. Executing deep convolutional operations on edge AI devices can be almost impossible due to their limited computational power and memory capacity. Therefore, the development of methods to compress CNN models and reduce memory usage is important to support these new applications.

In a previous study, quantized CNN models were proposed as a framework to accelerate CNN models and reduce the storage space and memory usage required (Wu et al. (2016)); their results showed a speedup by a factor of 4-6 and compression by a factor of 15-20 compared to the conventional benchmark, with less than 1% degradation in image classification accuracy. This method also enables accurate image classification in less than one second, even on mobile devices. Many other compression methods utilizing model quantization have also been proposed. Along these lines a previous study introduced per-channel quantization instead of per-layer quantization, and stated that 8-bit quantization could be performed while minimizing the accuracy degradation due to quantization (Lee et al. (2018)). In another study (Choukroun et al. (2019)), per-channel quantization was shown to be effective for 4-bit quantization with low bitwidths by solving a

minimum mean square error problem. Furthermore, a method of quantization designed to minimize cross-entropy error has been proposed (Nahshan et al. (2021)). A method of quantization that approximates a loss function and optimizes a rounding problem has also been proposed (Nagel et al. (2020)). As another approach (Wu et al. (2020)), a partial quantization which investigates the sensitivity of individual layers was considered. One of the studies referring to this partial quantization is the greedy search algorithm (GSA) (Tsuji et al. (2021, 2022)), which formulated the partial quantization of neural networks as a simple combinatorial problem. This algorithm was inspired by the submodular function and achieves both compression and speedup in quantization. Other works have considered an analysis of the $\Delta loss$ function (DLA) (Okado et al. (2022)). In this study, we use a sensitivity analysis method based on the idea that accuracy can be improved by using the per-channel $\Delta loss$ in the convolutional layer, and aim to improve the tradeoff between accuracy and compression ratio. These studies have mainly focused on quantization at a single bitwidth. Furthermore, several studies have found that assigning different bitwidths to layers or channels of quantization can contribute to the compression ratio. One example is the constraint optimization algorithm (COA) (Chen et al. (2021)) for mixed-precision quantization that utilizes the Hessian and achieves a high compression ratio yet uses retraining. In addition, hardware-friendly mixed quantization (MXQN) (Huang et al. (2021)) has also been applied to deep convolutional neural networks without retraining or finetuning. However, this technique does degrade the inference accuracy of the pretrained model.

In contrast, our proposed method uses multiple bitwidths of quantization to provide sufficient compression of the pretrained model without retraining or degradation of inference accuracy.

3. Semilayer-Wise Mixed-Precision Quantization Algorithm

In this section, we describe the proposed mixed-precision quantization method, which combines multiple quantization bitwidths. We also illustrate the application of our approach with a case study utilizing a ResNet-18 (He et al. (2016)) model and the ImageNet dataset (Deng et al. (2009)).

3.1 $\Delta loss$ per Channel

The ResNet-18 model used in this study includes 16 convolutional layers in the hidden layer. The convolutional layers are designated as layers 1, 2, ..., 16, starting with the convolutional layer closest to the input layer. A pre-trained model (PytorchDevTeam (2015)) was prepared on the ImageNet dataset for quantization. First, the first channel of layer 1 was quantized and tested using validation data. The value $\Delta loss$ is defined as the difference of the cross-entropy loss after quantization $L(\boldsymbol{w} + \Delta \boldsymbol{w})$ and the cross-entropy loss before quantization $L(\boldsymbol{w})$.

$$\Delta loss = L(\boldsymbol{w} + \Delta \mathbf{w}) - L(\boldsymbol{w}), \tag{1}$$

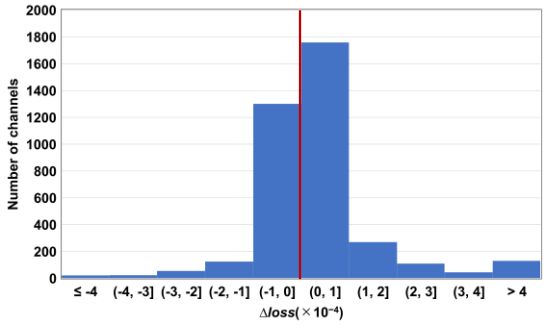

Figure 1: $\Delta \boldsymbol{loss}$ distribution for all channels in the convolution layer of a ResNet-18 model. The red line represents a $\Delta \boldsymbol{loss}$ value of zero. The left side of the red line represents channels with negative $\Delta \boldsymbol{loss}$ values, which may improve accuracy in the model.

where $\Delta \boldsymbol{w}$ is a quantization error, which is the difference in the weights after and before quantization. The quantization method is the same as the "qint" cast in PyTorch (PytorchDevTeam (2022)).

We evaluate $\Delta loss$ for all channels in the model. Figure 1 shows the $\Delta loss$ distribution of ImageNet image classification when 4-bit quantization was applied to all channels on a ResNet-18 model. The horizontal axis represents the $\Delta loss$ and the vertical axis represents the number of channels. Figure 1 shows that about half of the channels have negative $\Delta loss$ values. This indicates that quantization may be expected to improve the inference accuracy for about half of the channels in the pretrained model.

Figure 2 shows the $\Delta loss$ distribution of the 512 channels of the 16th convolutional layer of ResNet-18 as an example. Experiments were conducted using 8-bit, 6-bit, and 4-bit quantizations. The vertical axis represents the $\Delta loss$ quantized by a single channel and the horizontal axis represents the L^1 norm of the quantization error $\Delta \boldsymbol{w}$ at that time. In Figure 2, when the same channel was quantized with different quantization bitwidths, we experimentally confirmed that the $\Delta loss$ values with different quantization bitwidths had different values of $\Delta loss$ even in the same channel.

In quantization, the weights $\boldsymbol{w}$ are shifted by a quantization error $\Delta \boldsymbol{w}$. In this case, $\Delta loss$ is approximated as a quadratic function (Nagel et al. (2020)).

$$\Delta loss \approx \Delta \boldsymbol{w}^{\mathrm{T}} \cdot \boldsymbol{g} + \frac{1}{2} \Delta \boldsymbol{w}^{\mathrm{T}} \cdot \boldsymbol{H} \cdot \Delta \boldsymbol{w}, \tag{2}$$

where $\boldsymbol{g}$ is the gradient and $\boldsymbol{H}$ is the Hessian. From Figure 2, it may be observed that for a single channel, there are different values of $\Delta loss$ obtained by quantization. The results also confirmed that there exists an excellent quantization bitwidth for which the value takes the minimum, i.e., a given model is considered to have a high affinity for a certain quantization bitwidth. In the proposed method, quantization is performed sequentially starting from the value with the largest quantization (8-bit quantization in this paper) bitwidth. Then, gradually decreasing the quantization bitwidth exploits a smaller $\Delta loss$ value with high quantization affinity for each channel.

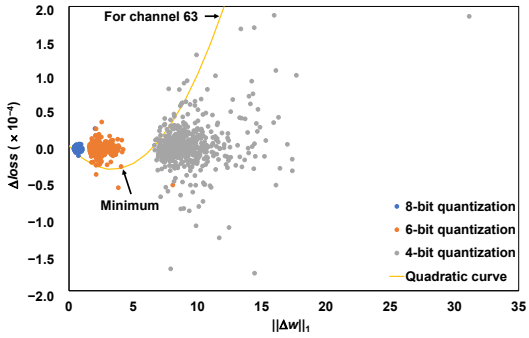

Figure 2: Scattering plots of $\|\Delta \boldsymbol{w}\|_1$ and $\Delta \boldsymbol{loss}$ for all channels in layer 16 of a ResNet-18 model. Blue dots represent 8-bit quantization, orange dots represent 6-bit quantization, and gray dots represent 4-bit quantization. The yellow quadratic curve shows the $\Delta \boldsymbol{loss}$ values for the 63-rd channel as an example.

3.2 Semilayer-Wise Quantization

Each layer is divided into two semilayers based on the positive and negative values of $\Delta loss$. A schematic diagram of a single layer divided into semilayers is shown in Figure 3. This process is applied to all hidden convolutional layers. For example, ResNet-18 has 16 hidden convolutional layers, for a total of 32 semilayers. The purpose of this operation is to divide each layer into channels for which quantization may be expected to improve accuracy and channels for which there is some possibility of decreasing accuracy. Table 1 summarizes the $\Delta loss$ value at 6-bit quantization of an entire semilayer for layer 1, layer 5, layer 9, and layer 13 in ResNet-18. We observed that all negative semilayers exhibited negative loss values, and vice versa.

The semilayer is recombined with each quantization bitwidth and quantized semilayer because different quantization bitwidths have different values of $\Delta loss$. For example, after quantizing an 8-bit semilayer to 8 bits, the semilayer was recombined to a 6-bit semilayer and quantized to 6 bits. This process may result in channels that were not quantized with 8-bit quantization being quantized with 6-bit quantization. This is intended to further increase the compression ratio. In our method, 8-, 6-, and 4-bit quantization bitwidths are selected.

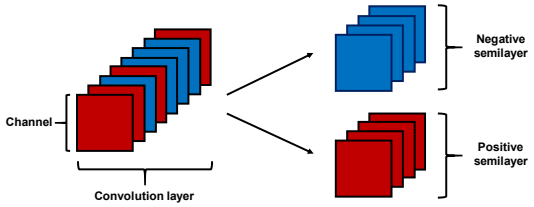

Figure 3: Schematic diagram of one layer divided into semilayers. Red channels represent positive $\Delta \boldsymbol{loss}$ and blue channels represent negative $\Delta \boldsymbol{loss}$.

Table 1: $\Delta \boldsymbol{loss}$ analysis of semilayer corresponding to the group with positive and negative $\Delta \boldsymbol{loss}$ and negative $\Delta \boldsymbol{loss}$ in convolution layer 1, layer 5, layer 9, and layer 13.

Negative semilayer	1-negative	5-negative	9-negative	13-negative
$\Delta loss$	-20.8×10^{-4}	-25.2×10^{-4}	-26.9×10^{-4}	-23.1×10^{-4}
Positive semilayer	1-positive	5-positive	9-positive	13-positive
$\Delta loss$	72.7×10^{-4}	51.4×10^{-4}	32.4×10^{-4}	28.3×10^{-4}

3.3 KL Divergence in Sensitivity Analysis

For all semilayers described in subsection 3.2, we compute a KL divergence between the probability distribution obtained from the softmax output of the original model and that of the model after quantizing all channels belonging to a semilayer. The input data used for the ImageNet classification is 50-k validation data. This indicator is computed for all semilayers. When quantizing multiple parameters collectively, the effect on the output of the neural network increases with the number of parameters involved. Therefore, in the sensitivity analysis, we normalize the KL divergence by dividing the number of parameters in the entire semilayer. Resnet-18 models include 64 channels in convolution layers 1 to 4, 128 channels in layers 5 to 8, 256 channels in layers 9 to 12, and 512 channels in layers 13 to 16. Because the amount of information per-channel differs depending on the number of channels, the difference in the amount of information is considered by normalizing the number of parameters held by the semilayer. The normalized sensitivity is derived over multiple bitwidths such as 8-bit, 6-bit, 4-bit, and 2-bit quantizations.

KL divergence use justification

$\Delta loss$ is used as the sensitivity in sensitivity analysis, which is considered informative and effective in partial quantization (Okado et al. (2022)). However, when $\Delta loss$ is used as the sensitivity, the degree of change from the original model is not considered. Alternatively, semilayers with larger absolute values of $\Delta loss$ may have more substantial changes in the model. If the model has a substantial change in weight, it becomes difficult to continue compressing the model without degrading accuracy. Therefore, this study uses KL divergence normalized by dividing the number of parameters in the entire semilayer as the sensitivity to limit the change in model weights to an appropriate level. Figure 4 shows the correlation between KL divergence normalized by dividing the number of parameters in the entire semilayer and $\Delta loss$ when each semilayer is quantized independently in ResNet-18 6-bit quantization. The correlation coefficient for the negative semilayer was -0.129 and for the positive semilayer was 0.834. Therefore, the larger the value of

the horizontal axis, the larger the absolute value of $\Delta loss$. Indeed, a more negative $\Delta loss$ following semilayer quantization improves the overall model loss performance. However, owing to significant model changes, this is not an effective measure of sensitivity. To mitigate this issue, we prioritized KL divergence as the preferred sensitivity measure.

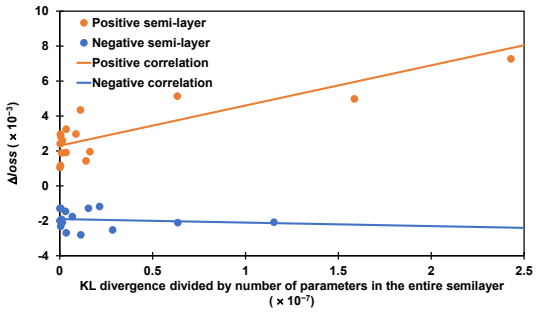

Figure 4: Correlation between KL divergence normalized by dividing the number of parameters in the entire semilayer and $\Delta loss$ with semilayers quantized.

3.4 Semilayer-Wise Mixed-Precision Quantization Analysis

This section summarizes subsections 3.1 through 3.3 and describes the full algorithm.

Accuracy Increasing Phase.

First, 8-bit quantization is applied to all channels to derive $\Delta loss$. Next, all semilayers are created. For these semilayers, all channels belonging to the semilayer are quantized in order of decreasing sensitivity. In the algorithm, postponing is introduced only once. This process ensures improved inference accuracy and is conducted to increase the compression ratio. If the inference accuracy after quantization is lower than the inference accuracy before quantization, this semilayer is not quantized, and is instead postponed. This is repeated for all semilayers. To achieve realistic computation time, the decision is made on a per-semilayer basis rather than on a per-channel basis. As a result of this process, only semilayers the accuracy of which increases after and before quantization are quantized to 8 bits. The other semilayers remain at 32 bits. The next quantization bitwidth is reduced to 6 bits. After that, semilayers are created based on the $\Delta loss$ of all channels derived in advance. All semilayers are quantized with postponing process. In the postponing process, the original bitwidth is restored, i.e., either 32 or 8 bits. As a result, the model is a mixture of 32-bit, 8-bit, and 6-bit channels. This process is performed while varying the bitwidth of quantization, such as 4-bit quantization, 2-bit quantization, and so forth. However, the goal of this phase is to increase inference accuracy. Therefore, if no increase in accuracy is observed, the method stops varying the bitwidth and moves on to the next phase.

Postponing Phase.

Next, quantization is performed on the 32-bit channels that have not yet been quantized. In this case, for the sake of realistic computation time and algorithmic simplicity, we do not perform an additional postponing process. Again, 8-bit quantization is chosen for the first quantization bitwidth. First, a semilayer is created for the remaining channels. Next, all semilayer quantizations are performed in order of decreasing sensitivity for each semilayer. Experiments are conducted independently by overwriting the quantization bitwidths. As a result, we confirm that the compression ratio increased for 6-bit width compared to 8 bits, and that 4 bits did not provide the same compression ratio as 6. Therefore, in this phase, a bitwidth of 6 was selected and the same processing was performed as in the stand-alone experiment.

The algorithm is intended to improve the compression ratio by increasing the inference accuracy of the model in the accuracy-increasing phase and gradually degrading the inference accuracy of the model in the postponing phase. Algorithm 1 shows pseudocode that summarizes this subsection.

Algorithm 1 Semilayer-Wise Mixed-Precision Quantization

Input: pretrained FP32 model, $quantizationBit \in \{8, 6, 4, 2\}$
Output: compressed model

for all channels in convolution layers **do**
    calculate $\Delta loss$ for all elements of the $quantizationBit$ (1)
end for
for $quantizedBit$ **do**
    for All convolution layers **do**
        if $\Delta loss > 0$ **then**
            $positiveList \leftarrow channel$
        else[$\Delta loss <= 0$]
            $negativeList \leftarrow channel$
        end if
    end for
    $semilayerList \leftarrow positiveList, negativeList$
    for $c \in semilayerList$ **do**
        calculate KL divergence
        calculate $Sensitivity$ ▷ KL divergence divided by the number of semilayer parameters
    end for
    $SemilayerList$ sorted by $Sensitivity$ in descending order
    for $c \in semilayerList$ **do** ▷ Accuracy-Increasing Phase
        quantize c and infer the model
        if accuracy after quantized < accuracy before quantized **then**
            restore c before quantization
        end if
    end for
end for
$postponedSemilayerList \leftarrow elements$ never quantized in $semilayerList$ ▷ Postponing Phase
for $quantizationBit$ **do**
    perform single-bit quantization for $postponedSemilayerList$
end for
select the best single-bit quantization

4. Experiments

We evaluated experiments using ResNet-18, ResNet-34, and ResNet-50 (He et al. (2016)) for the proposed method. The evaluation method was based on inference results from image classification; the ImageNet and CIFAR-10 datasets (Krizhevsky and Geoffrey (2009)) were used with a system

with Nvidia Quadro RTX 8000 GPU (Nvidia (2023)). The source code of the algorithm and the weights of the model are available at the URL given below[1].

4.1 ResNet-18

Figure 5 shows the results obtained by quantization with a single bitwidth and the proposed algorithm for the ResNet-18 model using the ImageNet and CIFAR-10 datasets. The results are obtained by moving from the accuracy-increasing phase to the postponing phase without changing the bitwidth. The horizontal axis is the compression ratio, and the vertical axis is the inference accuracy. Our proposed method achieved a compression ratio of 81.44% for the ImageNet dataset and 80.56% for the CIFAR-10 dataset to the extent that accuracy degradation is not less than the accuracy of the original model, exceeding the result of single bit quantization.

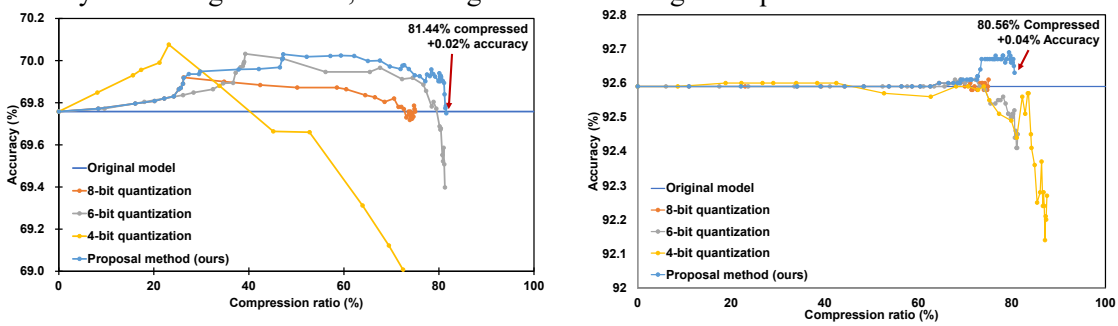

Figure 5: Comparison of accuracy degradation in ResNet-18. The left figure shows the results when using the ImageNet dataset, and the right figure shows the results when using the CIFAR-10 dataset.

4.2 ResNet-34

The left figure in Figure 6 shows the results obtained by quantization with a single bitwidth and our method when a ResNet-34 model and the ImageNet dataset were used. The proposed method achieved a compression ratio of 84.25% to the extent that accuracy degradation was not less than the accuracy of the original model, exceeding the result of the single-bitwidth quantization.

4.3 ResNet-50

The right figure in Figure 6 shows the results obtained by quantization with a single bitwidth and our method when a ResNet-50 model and the ImageNet dataset were used. The proposed method achieved a compression ratio of 80.39% to the extent that accuracy degradation was not less than the accuracy of the original model. However, it did not exceed the compression ratio of the 6-bit quantization. In the ResNet-50 model, many 8-bit quantization channels remain in the accuracy-increasing phase. The 8-bit channels would be better to be selected as 6-bit quantization in the postponing phase in terms of compression ratio. However, all channels in the model were quantized at a compression ratio of 80.39%, and the inference accuracy achieved 0.136% over the original model baseline. This exhibits excellent performance of the proposed method in the high compression ratio and the high inference accuracy.

[1] https://github.com/kenm-28/Semilayer-Wise-Mixed-Precision-Quantization/tree/main

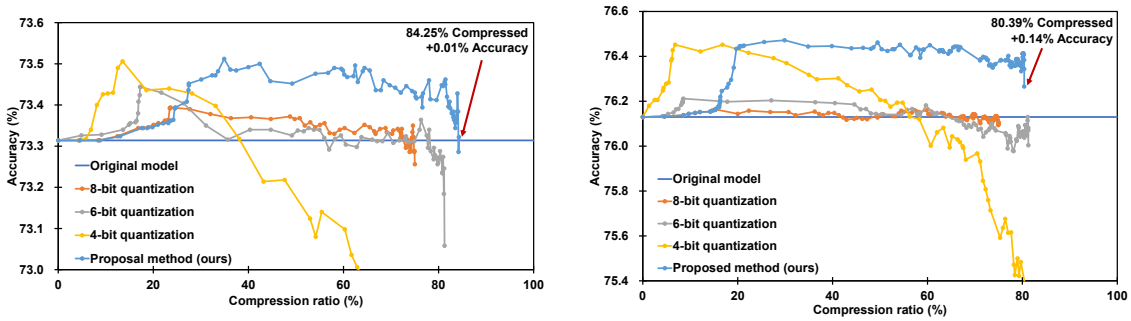

Figure 6: Comparison of accuracy degradation. The left figure shows the results when using the ResNet-34 model and the ImageNet dataset, and the right figure shows the results when using the ResNet-50 model and the ImageNet dataset.

4.4 Comparison with Other Methods

Table 2 shows the results of the proposed method compared to the conventional methods. The proposed method achieved a sufficient compression ratio compared to the conventional method within the range of accuracy degradation without retraining. The ResNet-18 and the ResNet-50 models did not outperform the COA (Chen et al. (2021)) compression ratio, but the COA is computationally expensive because of the retraining process, and the accuracy was below that of the baseline. Our method was always superior to the baseline in terms of accuracy.

Table 2: Results of comparison with other methods for the ImageNet dataset. "P" and "MP" respectively refer to partial quantization and mixed-precision quantization. The best results are **bolded**, and the underlines mark the second-best results.

Network	Dataset	Method	Baseline (%)	bitwidth	Comp. ratio (%)	Top-1 acc. (%)	Top-1 increase (%)	Re-train
ResNet-18	ImageNet	FP32	69.75	–	–	–	–	No
		GSA	69.75	6P	64.34	**69.79**	**+0.04**	No
		DLA	69.75	6P	79.06	69.77	+0.02	No
		MQXN	69.75	8P	75.00	67.61	−2.14	No
		COA	69.75	3MP	**87.98**	69.66	−0.09	Yes
		Ours	69.75	3MP	81.44	69.77	+0.02	No
	CIFAR-10	FP32	92.59	–	–	–	–	No
		GSA	92.59	6P	74.16	92.60	+0.01	No
		DLA	92.59	6P	79.07	92.59	0.00	No
		Ours	92.59	4MP	**80.56**	**92.63**	**+0.04**	No
ResNet-34	ImageNet	FP32	73.31	–	–	–	–	No
		GSA	73.31	6P	36.36	**73.33**	**+0.02**	No
		DLA	73.31	6P	69.88	73.32	+0.01	No
		MQXN	73.31	8P	75.00	71.43	−1.88	No
		Ours	73.31	3MP	**84.25**	73.32	+0.01	No
ResNet-50	ImageNet	FP32	76.12	–	–	–	–	No
		GSA	76.12	6P	73.45	76.13	+0.01	No
		DLA	76.12	6P	79.36	76.13	+0.01	No
		MQXN	76.12	8P	75.00	74.06	−2.06	No
		COA	76.12	2MP	**91.83**	75.28	−0.84	Yes
		Ours	76.12	3MP	80.39	**76.26**	**+0.14**	No

4.5 Ablation Study

We conduct an ablation study to assess the effective of the semilayers in our proposed method over layer-base quantization. Both experimental evaluations incorporate the postponing process and are compared in terms of the compression ratio to the extent that accuracy degradation was not less than the accuracy of the original model. The results of the ablation study are shown in Table 3. The compression ratios for the proposed semilayer-base method all exceed those for the layer-base one. It was confirmed that the introduction of semilayers can effectively compress models.

Table3: Ablation study of the proposal method.

Network	Dataset	Layer base	Semilayer base
		Comp. ratio (%)	
ResNet-18	ImageNet	69.71	**81.44**
ResNet-18	CIFAR-10	80.36	**80.56**
ResNet-34	ImageNet	78.42	**84.25**
ResNet-50	ImageNet	78.75	**80.39**

5. Conclusion

In this paper, we have proposed a new method of compressing neural networks by quantization. The proposed method achieved a higher compression ratio with less degradation of accuracy. The proposed method is not a single-bitwidth quantization, but rather a combined quantization of multiple bitwidths. Evaluation of the proposed method on a neural network-based image classification task on the ImageNet dataset showed that it improved the tradeoff between accuracy and compression ratio. In terms of specific results, image classification tasks using the ImageNet dataset, a ResNet-18 model was compressed with a ratio of 81.44%, a ResNet-34 was compressed with a ratio of 84.25%, and a ResNet-50 was compressed with a ratio of 80.39% by mixed-precision quantization. In image classification tasks using the CIFAR-10 dataset, a ResNet-18 was compressed with a ratio of 80.56% by the mixed-precision quantization.

References

Simone Bianco, Remi Cadene, Luigi Celona, and Paolo Napoletano. Benchmark Analysis of Representative Deep Neural Network Architectures. *IEEE access*, 6, pages 64270-64277. 2018.

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*, pages 5350-5359, 2021.

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 Workshop (ICCVW)*, pages 3009-3018. IEEE, 2019.

Jia Deng, Wei Dong, Richard Socher, Li.-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In 2009 *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. IEEE, 2009.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580-587, 2014.

Ross Girshick. Fast R-CNN. In *Proceedings of the IEEE international conference on computer vision*, pages 1440-1448, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770-778, 2016.

Chenglong Huang, Puguang Liu, and Liang Fang. MXQN:Mixed quantization for reducing bit-width of weights and activations in deep convolutional neural networks. *Applied Intelligence*, 51: pages 4561–4574, 2021.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2704-2713, 2018.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling Laws for Neural Language Models. *arXiv preprint arXiv:*2001.08361, 2020.

Alex Krizhevsky, and Hinton Geoffrey. Learning Multiple Layers of Features from Tiny Images, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6), pages 84–90, 2017.

Jun Haeng Lee, Sangwon Ha, Saerom Choi, Won-Jo Lee. and Seungwon Lee. Quantization for Rapid Deployment of Deep Neural Networks. *arXiv preprint arXiv:*1810.05488, 2018.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning Filters for Efficient ConvNets. *arXiv preprint arXiv:*1608.08710, 2016.

Markus Nagel, Rana Ali Amjad, Mart van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or Down? Adaptive Rounding for Post-Training Quantization. In *International Conference on Machine Learning*, PMLR, pages 7197-7206, 2020.

Yury Nahshan, Brian Chmiel, Chaim Baskin, Evgenii Zheltonozhskii, Ron Banner, Alex M. Bronstein, and Avi Mendelson. Loss aware post-training quantization. *Machine Learning*, 110(11–12), pages 3245–3262, 2021.

Nvidia. NVIDIA Quadro is now NVIDIA RTX, last accessed 2023/6/22. URL https://www.nvidia.com/en-us/design-visualization/quadro/

Kazuki Okado, Kengo Matsumoto, Atsuki Inoue, Hiroshi Kawaguchi, and Yasufumi Sakai. Channel-wise quantization without accuracy degradation using Δloss analysis. In 2022 7*th International Conference on Machine Learning Technologies* (*ICMLT*) pages 56-61, 2022.

PytorchDevTeam. Pytorch_vision_resnet master document, 2015. URL https://pytorch.org/hub/pytorch_vision_resnet/

PytorchDevTeam. Pratical quantization in pytorch master document, 2022. URL https://pytorch.org/blog/quantization-in-practice/

Karen Simonyan, and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv preprint arXiv:*1409.1556, 2014.

Satoki Tsuji, Hiroshi Kawaguchi, Atsuki Inoue, Yasufumi Sakai, and Fuyuka Yamada. 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, pages 886-901, 2021.

Satoki Tsuji, Fuyuka Yamada, Hiroshi Kawaguchi, Atsuki Inoue, and Yasufumi Sakai. Greedy search algorithm for partial quantization of convolutional neural networks inspired by submodular optimization. *Neural Computing and Applications,* pages 1-11, 2022.

Hao Wu, Patrick Judd, Xiaojie Zhang, Mikahail Isaev, and Paulius Micikevicius. Integer Quantization for Deep Learning Inference: Principles and Empirical Evaluation. *arXiv preprint arXiv:*2004.09602, 2020.

Jiaxiang Wu, Cong Leng, Yuhang Wang, Qinghao Hu, and Jian Cheng. Quantized Convolutional Neural Networks for Mobile Devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4820-4828, 2016.