PROFIT: A Novel Training Method for sub-4-bit MobileNet Models

Eunhyeok Park[1][0000−0002−7331−9819] and Sungjoo Yoo[2][0000−0002−5853−0675]

[1] canusglow@gmail.com and [2]sungjoo.yoo@gmail.com
[1] Inter-university Semiconductor Research Center (ISRC)
[2] Department of Computer Science and Engineering
[2] Neural Processing Research Center (NPRC)
[1,2]Seoul National University, Seoul, Korea

Abstract. 4-bit and lower precision mobile models are required due to the ever-increasing demand for better energy efficiency in mobile devices. In this work, we report that the activation instability induced by weight quantization (AIWQ) is the key obstacle to sub-4-bit quantization of mobile networks. To alleviate the AIWQ problem, we propose a novel training method called PROgressive-Freezing Iterative Training (PROFIT), which attempts to freeze layers whose weights are affected by the instability problem stronger than the other layers. We also propose a differentiable and unified quantization method (DuQ) and a negative padding idea to support asymmetric activation functions such as h-swish. We evaluate the proposed methods by quantizing MobileNet-v1, v2, and v3 on ImageNet and report that 4-bit quantization offers comparable (within 1.48 % top-1 accuracy) accuracy to full precision baseline. In the ablation study of the 3-bit quantization of MobileNet-v3, our proposed method outperforms the state-of-the-art method by a large margin, 12.86 % of top-1 accuracy. The quantized model and source code is available at https://github.com/EunhyeokPark/PROFIT.

Keywords: Mobile network, quantization, activation distribution, h-swish activation

1 Introduction

Neural networks are widely adopted in various embedded applications, e.g., smartphones, AR/VR devices, and drones. Such applications are characterized by stringent constraints in latency (for real-time constraints) and energy consumption (because of battery). Thus, it is imperative to optimize neural networks in terms of latency and energy consumption, while maintaining the quality of the neural networks, e.g., accuracy.

Quantization is one of the most effective optimization techniques. By reducing the bit-width of activations and weights, the performance and energy efficiency can be improved by executing more computations with the same amount of memory accesses and computation resources (e.g., the silicon chip area and

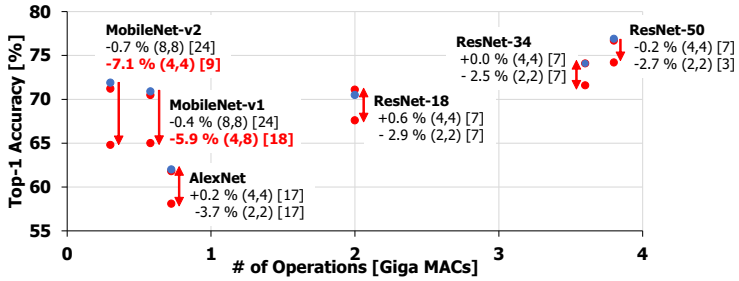


Fig. 1: Recent results of quantization studies. The blue dots represent full-precision accuracy and the red dots quantized accuracy. The tuple (a,w) represents the bit-width of activation and weight, respectively.

battery). The 8-bit computation is already popular [32, 33, 36, 38], and NVIDIA recently announced that tensor core supports 4-bit precision which gives more than 50 % of performance improvement on ResNet-50 [25]. We expect 4-bit and lower precision computation will become more and more important and make further contributions to the energy efficiency and real-time characteristics of future deep learning applications [26, 28, 30, 35, 36].

In order to support the up-coming hardware acceleration, there have been active studies on sub-4-bit quantization [3, 4, 15, 17, 21, 27, 39–42]. These studies show that deep networks, e.g., AlexNet or ResNet-18 for ImageNet classification [5], can be quantized into sub-4 bits with negligible accuracy loss, as shown in Figure 1. However, these out-dated networks are prohibitively expensive to use in mobile devices. In mobile devices, it is imperative to quantize the optimized networks, e.g., MobileNet-v2 [29] or MobileNet-v3 [13].

However, the previous quantization methods do not work well on the advanced optimized networks. These networks adopt novel structures like depth-wise separable convolution [14], inverted residual block with linear expansion layer [29], squeeze-excitation module, and h-swish activation function [13]. These structures have less redundancy and are vulnerable to quantization, and the h-swish activation function generates an asymmetric distribution. Previous quantization methods did not consider the optimizations, thus having a significant accuracy degradation in the sub-4-bit quantization of the advanced networks.

In this study, we propose two novel ideas that enable 4-bit quantization for the optimized networks. First, we report that the primary reason for the accuracy loss in sub-4-bit quantization is the activation instability induced by weight quantization (AIWQ). Weight quantization can skew the statistics of the output activation, i.e., the mean and variance, at every iteration during fine-tuning. This adversely affects the following layers and finally prevents the network from converging to a good optimal point in low-precision quantization. In order to address this problem, we propose a novel training method called **PRO**gressively-**F**reezing **I**terative **T**raining (PROFIT) that minimizes the effects of AIWQ by progressively freezing the weights sensitive to AIWQ during training.

Second, we identify the limitations of the state-of-the-art trainable methods of linear quantization in terms of asymmetric activation support, and we propose a novel quantization method called differentiable and unified quantization (DuQ) and negative padding. Many advanced networks begin to adopt the activation functions allowing a small amount of the negative value, e.g., hard swish (h-swish) of MobileNet-v3 and Gaussian error linear unit (GeLU) of BERT [6]. These activation functions increase accuracy with minimal overhead. However, they produce asymmetric output distributions. Existing quantization methods are only designed to support symmetric or non-negative distributions; therefore, they are unsuitable for the asymmetric distributions. The proposed DuQ method resolves the above problems without limiting the value range while minimizing both rounding and truncation errors in a differentiable manner. Furthermore, the novel negative padding idea contributes to accuracy improvement by appropriately utilizing the quantization levels under an asymmetric distribution.

2 Related Work

The neural network architecture has been continuously improved while increasing accuracy at a lower computation cost. MobileNet-v1 [14] and -v2 [29] introduced a depth-wise separable convolution and an inverted residual structure respectably. MNasNet was designed based on AutoML, which automates the network architecture search, considering the computation cost [34]. MobileNet-v3 is the state-of-the-art network, which was designed from MNasNet by improving it with the h-swish activation function and squeeze-excitation modules [13]. Compared to the conventional deep networks like AlexNet [20], VGG [31], and ResNet [11], the recent networks have adopted optimized structures for better accuracy at a low computation cost. However, such optimized structures render quantization challenging, especially for 4-bit and lower precision quantization.

Several studies have been proposed for sub-4-bit quantization. [27] and [39] are the representative studies showing that neural networks can be quantized into sub-4-bits with marginal accuracy loss. [42] proposed progressive quantization, which reduces the precision in a progressive manner. [3, 4, 7, 9, 17] and [23] show that networks for large-scale datasets, e.g., ResNet [11] for ImageNet [5], can be quantized into sub-4-bits without accuracy loss. Recently, [8] and [24] are focused on post-training 8-bit quantization, and showed that quantization introduces a bias shift, which acts as the main cause of accuracy degradation. The previous works show the potential of aggressive quantization in terms of sub-4-bits (with fine-tuning) for AlexNet and ResNet, or 8-bit quantization (without fine-tuning) for MobileNet-v2. In this study, we focus on the 4-bit and lower precision quantization (with fine-tuning) for recently optimized networks such as MobileNet-v2 and -v3.

Many of the previous quantization methods utilize hand-crafted loss, e.g., L2 distance between full-precision and quantized data [3]. For a lower precision, it is desirable to learn the quantization interval of the loss of the target task, e.g., cross-entropy loss for classification. In [3, 4, 17] and [7], the quantization

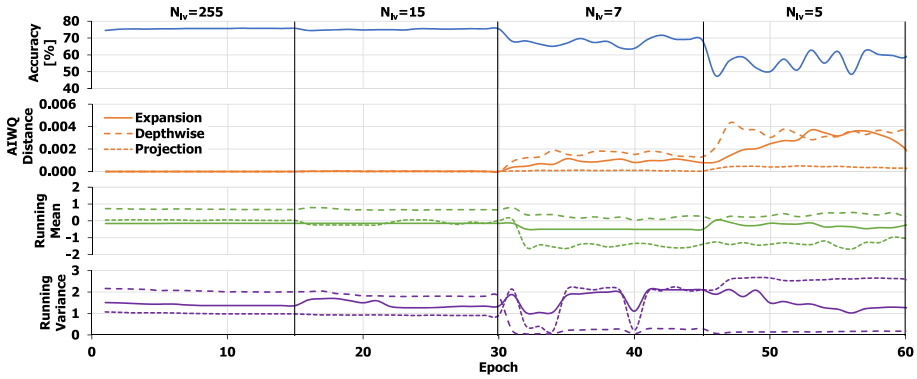


Fig. 2: Top-1 accuracy [%], AIWQ distance, and the running mean, and the running variance during fine-tuning for quantization where N_{lv} represents the number of available quantization levels. Running mean and variance are extracted from the arbitrary channels of batch normalization layer after the depth-wise convolution in the 2nd inverted residual module of MobileNet-v3.

interval is learned via backpropagation. In this study, we also propose DuQ with a negative padding idea that learns the quantization interval by utilizing the gradients and asymmetric activations better than the previous methods.

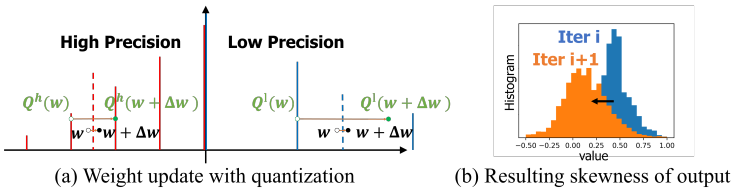
3 AIWQ and PROFIT

In this section, we first demonstrate that the AIWQ problem is strongly correlated with the accuracy degradation at low precision. Then, we present a metric to measure the activation instability and propose a training method called PROFIT that controls the training of each layer to minimize the effect of AIWQ based on the presented metric.

3.1 Observation

Figure 2 (Accuracy) shows the accuracy of MobileNet-v3 for the Cifar-100 dataset [19]. The accuracy is measured during fine-tuning in progressive weight quantization [42], where the number of quantization levels of the weights N_{lv} are gradually reduced from 255 to 5 while using full-precision activation. The accuracy curves in Figure 2 show that, in each precision case, e.g., $N_{lv} = 15$, the test accuracy is gradually recovered as the fine-tuning advances. However, when the number of available quantization levels reduces to 7 or lower, the accuracy significantly oscillates and fails to converge.

From our analysis, that will be given in the next subsection, this is mainly due to the activation instability during fine-tuning, as shown in Figure 3. The challenge is that the effects of the weight update, due to backpropagation, could be amplified by the quantization operation, i.e., rounding operation. Particularly,



(a) Weight update with quantization (b) Resulting skewness of output

Fig. 3: Activation instability induced by weight quantization.

as Figure 3 (a) shows, if a weight near the quantization threshold is updated to change its value crossing the threshold, then the quantization will result in different quantized weight values before and after the weight update. Thus, the results of the convolution operation will change due to the weight update and quantization. As Figure 3 (b) shows, this skews the statistics of the output activation, which affects the following layers, including the normalization layers; therefore, yielding inaccurate running mean and variance in these layers. The inaccurate running mean and variance, obtained during training, degrades the test accuracy as illustrated in Figure 2 because they are utilized in the normalization layers during the test but can't represent the actual statistics of the activation.

At lower precision, the activation instability induced by weight quantization (AIWQ) becomes more significant because the space between two adjacent quantization levels ($\sim 1/2^{\textbf{bit-width}}$) becomes large. Thus, the lower the precision gets, the more the activation instability can be incurred.[3] Besides, please note that the AIWQ problem is also found in conventional neural networks. However, because these networks use full convolution as their building block having more than hundreds of accumulation per output, the quantization error is likely to be amortized based on the law of large numbers. This makes the networks robust to quantize, but they also suffer from instability when the precision lower.

3.2 Activation Instability Metric

We present a metric to quantify the per-layer activation instability and use it to (1) prove that AIWQ is correlated with the test accuracy of the low-precision model (in Figure 2) and (2) utilize the per-layer sensitivity when determining the order of freezing the weights during training (to be explained in Section 3.3).

In order to measure per-layer activation instability, a desirable solution would be to calculate the KL divergence between two distributions of the outputs before (p_o^t) and after (q_o^t) a training iteration t. We first calculate the per-output channel KL divergence between p_o^t and q_o^t. Then, as shown below, we compute the layer-wise AIWQ metric D^l by averaging the per-output channel KL divergence across

[3] Note that, in higher (lower) precision, there will be more (less) occurrences of smaller (larger) amounts of activation instability. Our study empirically shows that a few occurrences of large activation instability at low precision tend to have a higher impact on the test accuracy than many occurrences of small activation instabilities at high precision.

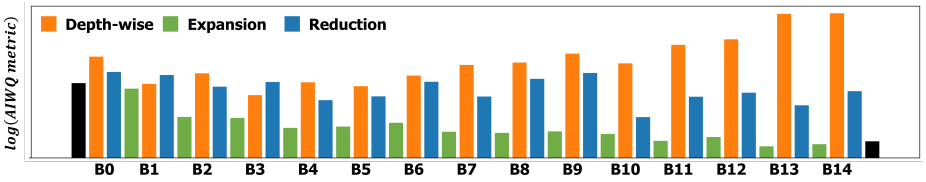


Fig. 4: Layer-wise AIWQ metric measured at MobileNet-V3 on Imagenet. Bx refers to x-th building block of MobileNet-v3. Please note that y-axis is in log-scale thus the layer-wise sensitivity is highly vary depending on layer index.

all the output channels of the current layer in the current training batch.

$$D^l = E_o^t\big[D_{KL}(p_o^t \;||\; q_o^t)\big]. \tag{1}$$

We simplify the computation of the KL divergence by adopting a second-order model which considers the mean and variance of per-channel output distribution because the computation of the KL divergence is expensive, and the second-order model proves sufficient for our goal of evaluating the relative order between layers.[4] Figure 4 shows the AIWQ metric of MobileNet-v3 on ImageNet. As the figure shows, depth-wise convolution layers tend to exhibit a large AIWQ while some reduction (point-wise 1x1 convolution) layers also give a larger AIWQ than the depth-wise layers in the early layers.

Recall Figure 2, which illustrates how the AIWQ metric varies during fine-tuning. The AIWQ increases in the 7- and 5-level quantization, which empirically proves that weight quantization at low precision can incur significant perturbation of the convolution output, i.e., makes the output activation unstable. As shown in the figure, such an instability causes the running mean and variance[5] to fluctuate, which prevents us from obtaining good running mean and variance during training. When comparing the accuracy, the AIWQ metric, and the mean and variance in Figure 2, they are closely correlated at low precision, i.e., $N_{lv} = 5$. In the following subsection, we use the AIWQ metric to schedule which layers to freeze first during the fine-tuning, which contributes to reducing the AIWQ, hence improving the test accuracy.

3.3 PROFIT

We propose a novel training method which aims at minimizing the AIWQ effect to improve the accuracy of low precision networks. Our basic idea is progressively

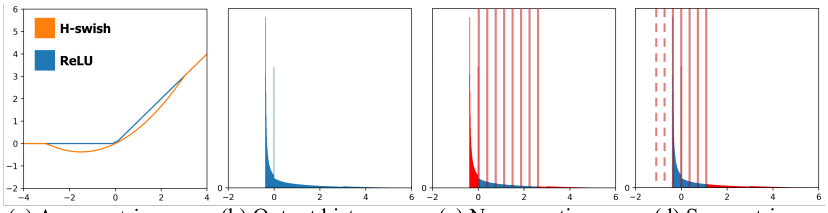
[4] Note that in [24, 8] the first-order momentum, i.e., the channel-wise mean is utilized to evaluate the difference in the output distributions. From our observation, the channel-wise variance is also significantly skewed by the AIWQ, and significantly affects the accuracy via normalization. AIWQ metric is designed to consider the two important momenta, the mean and variance, concurrently with an affordable cost.

[5] We show the mean and variance on three sampled output channels in Figure 2.

Algorithm 1 Pseudo code of PROFIT algorithm

```
Initialize network and full-precision training (+ progressive quantization)
Set quantization parameters according to the target bit-width.
procedure AIWQ SAMPLING
    for layer ∈ network.layers do
        if layer is quantized convolution then
            metric_map[layer] = 0
    for i ∈ sampling iterations do
        for layer ∈ network.layers do
            layer.forward()
            if layer is quantized convolution then
                metric_map[layer] += layer.AIWQ_metric
        network.update()
AIWQ_list ← sort_by_value(metric_map, order=descending)
N_layers ← len(AIWQ_list)
procedure PROFIT
    for n ∈ N_PROFIT do
        for e ∈ Profit_Epoch do
            network.training_epoch()
        begin ← n * N_layers/N_PROFIT, end ← (n + 1) * N_layers/N_PROFIT
        freeze_target_layers ← AIWQ_list[begin:end]
        for layer ∈ freeze_target_layers do
            layer.learning_rate ← 0
    for e ∈ BN_Epoch do
        network.training_epoch()
```

freezing (the weights of) the most sensitive layer to AIWQ to remove the fluctuation source, thus allowing the rest of the layers to converge to a more optimal point. We determine the layer-wise order of the weight freezing considering the per-layer AIWQ metric in Eqn. 1. Algorithm 1 shows how our method, called **PRO**gressively-**F**reezing **I**terative **T**raining (PROFIT), works. When PROFIT is triggered, we start a sampling stage where we evaluate the per-layer AIWQ metric for each layer. After the sampling stage, we perform fine-tuning in an initial stage without freezing weights. Subsequently, after sorting all the weight layers in terms of the per-layer metric, we perform weight freezing stages by selecting the most sensitive layers (the ones having the largest AIWQ metric values) and freezing their weights. As shown in the algorithm, we iteratively perform N_{PROFIT} freezing stages. Thus, in each stage, N_{layers}/N_{PROFIT} (N_{layers} is the total number of quantized layers) are selected from the sorted layer list and their weights are frozen. Then, we perform training for all un-frozen layers. After finishing the additional training stage, we select the next set of un-frozen sensitive layers (another N_{layers}/N_{PROFIT} layers) and repeat the same procedure until there is no more un-frozen layer left. Finally, we perform an additional training stage (typically, 3-5 epochs) for the normalization layers while freezing all the other layers. This further stabilizes the statics of the normalization layers.



(a) Asymmetric range of h-swish function (b) Output histogram of h-swish function (c) Non-negative quantization (d) Symmetric quantization

Fig. 5: Characteristics of h-swish function and corresponding non-negative and symmetric quantization.

As will be shown in our experimental results, PROFIT improves the accuracy of low precision networks by significantly reducing the effect of AIWQ.

4 Quantization for Asymmetric Distributions

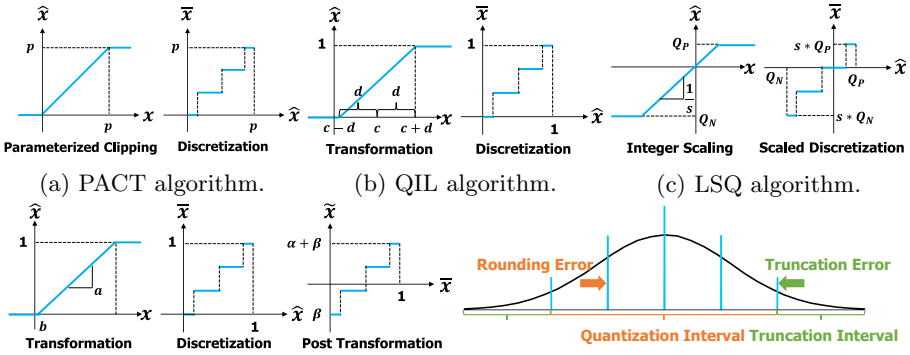
In many advanced networks, the activation functions allow a small number of negative values, e.g., h-swish of MobileNet-v3 and GeLU of BERT, are becoming more popular. These functions increase the accuracy with minimal computation overhead, thus gradually expanding their scope of use. However, they have a critical limitation in terms of quantization. Because of the negative range, the output has an asymmetric distribution. Even though the negative range is small, many values are concentrated in that area, as shown in Figure 5. These negative values should be carefully considered to maintain accuracy at low precision.

However, existing quantization methods are only designed for symmetric or non-negative output. In such a case, when we apply quantization to only the non-negative output of the h-swish function, many negative values are ignored (Figure 5 (c)). On the contrary, when we apply symmetric quantization, some of the quantization levels, allocated for large negative values, are wasted, and a significant truncation error is incurred due to the narrower value range for positive values (Figure 5 (d)). In either case, there is a significant loss in accuracy.

In order to quantize the asymmetric distribution with minimal accuracy loss, we propose two ideas: DuQ and negative padding. First, we propose a quantization algorithm called Differentiable and Unified Quantization (DuQ) that resolves the above problems without limiting the value range while minimizing the rounding and truncation errors in a differentiable manner. Second, we propose negative padding that allows us to avoid wasting quantization levels, hence improving accuracy at low precision.

4.1 Limitations of State-of-the-Art Methods

Our goal is to realize differentiable quantization, which minimizes the task loss of the asymmetric distribution of activation. There are three representative meth-



(a) PACT algorithm. (b) QIL algorithm. (c) LSQ algorithm.

(d) Proposed DuQ algorithm. (e) Two error sources of quantization.

Fig. 6: Quantization algorithm details.

ods, parameterized clipping activation function (PACT) [3], quantization interval learning (QIL) [17], and learned step size quantization (LSQ) [7]. In all the methods, the differentiable parameters and quantization intervals are updated through backpropagation to minimize the task loss.

Both PACT and QIL have a critical limitation in supporting new activation functions because the transformation stage or parameterized clipping stage forces the activation data to be mapped to [0, 1] in [17] or [0, p] in [3], as shown in Figure 6 (a) and (b). Thus, it is not applicable to activation functions with asymmetric distributions. In the case of LSQ, as shown in Figure 6 (c), a trainable scale parameter s is adopted, and the hand-crafted parameters utilize the number of negative quantization levels Q_N and that of positive ones Q_P. LSQ also handles either symmetric ($Q_N = Q_P = 2^{bit-1} - 1$) or non-negative ($Q_N = 0, Q_P = 2^{bit} - 1$) distributions. Additionally, because the user predetermines the number of quantization levels for negative and positive ranges, it has a limitation in handling various distributions across the layers.

4.2 Proposed Method: DuQ

Our proposed method, called DuQ, learns the quantization and truncation intervals through back-propagation.This is an extension of QIL with scale (α) and shift(β) parameters that remove the limitation of QIL, i.e., the limited output range of [0, 1].

The two stages of transformation (Eqn. 2) and discretization (Eqn. 3) are identical to those of QIL except that the slope a and offset b are used in the transformation stage instead of the center c and width d in QIL. As Eqns. 2 and 4 show, we use the softplus function for a and α to make them positive values for improving the stability of the transformation stage. Eqn. 3 represents the discretization stage, where N_{lv} is the number of quantization levels.

$$\hat{x} = clip\big((x - b)/a', 0, 1\big),\ a' = softplus(a), \tag{2}$$

$$\bar{x} = round\big((N_{lv} - 1) \cdot \hat{x}\big)/(N_{lv} - 1), \tag{3}$$

$$\tilde{x} = \alpha' \cdot \bar{x} + \beta,\ \ \alpha' = softplus(\alpha). \tag{4}$$

Our proposed DuQ method allows us to utilize the full value range of activation including the negative ones, and to achieve that, the discretized data can be mapped to an arbitrary range through a scale α and offset β, in the post-transformation stage, as shown in Eqn. 4 as shown in Figure 6 (d). One additional advantage is DuQ utilizes all the gradients across the entire activation data. PACT only utilizes the gradients from the truncation interval (the value range larger than the truncation threshold), while QIL only utilizes the gradients from the quantization interval (between the minimum and maximum quantization levels). Both utilize only a portion of the backpropagated error. However, DuQ utilizes all the gradients to learn a good quantization interval considering the trade-off between rounding and truncation errors, as Figure 6 (e) shows.

4.3 Negative Padding

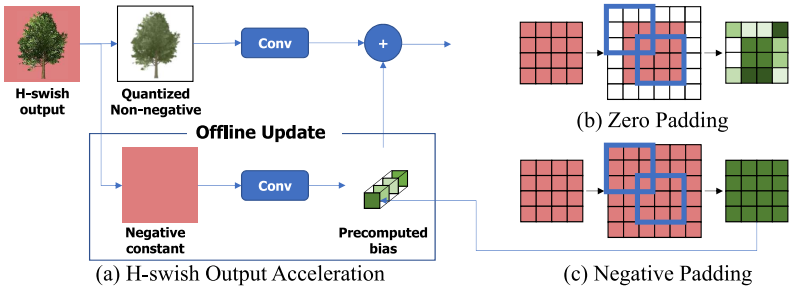


Fig. 7: Negative padding for h-swish function.

DuQ is flexible enough to support asymmetric distributions. However, many of existing hardware accelerators support only symmetric and non-negative integer types. We propose a idea called negative padding to accelerate the quantized network having asymmetric distributions on such hardware accelerators.

Even though the output of the h-swish function has an asymmetric distribution, it has a constant minimal value, -0.375. As shown in Figure 7 (a), by shifting the activations by the minimal value, the activations can be broken down into two components, constant negative ones, and shifted non-negative ones. The output corresponding to the constant negative feature can be calculated offline, and the output corresponding to the input-dependent non-negative activations can be efficiently calculated by the hardware accelerator. By combining those two outputs, we can obtain the output. Note that, because of the linearity of the convolution and matrix multiplication, the output is correct.

However, because of conventional zero padding, the output corresponding to constant negative input activations has inconsistent values on the edges, as

shown in Figure 7 (b). We need to store the spatial position-dependent values, which can increase the cost of memory access and computation. This problem can be solved by adopting negative padding instead of zero padding (Figure 7 (c)). When we pad the edge of the activation with the constant minimal value of the h-swish function, i.e., -0.375, the output has an identical value for all the features in the same spatial dimension. This constant output enables the pre-computed result to be mapped by the efficient channel-wise bias-add operator, thereby minimizing computational/storage costs.

The proposed negative padding makes the minimum quantization level of input activation zero. Thus, the proposed method can also be beneficial to zero skipping solutions like zero activation-skipping hardware accelerators [1, 37] to improve the inference speed. Please note that the negative padding is applicable to any non-linear activation function, which has asymmetric distributions with a constant minimal value.

5 Experiments

We implemented the proposed methods in PyTorch 1.4 and demonstrated their effectiveness by measuring the accuracy of the quantized networks. We applied quantization to well-known optimized CNNs, MobileNet-v1 to v3 and MNasNet-A1. The networks were trained on 4-GPU with a 256 batch, SGD with momentum, and cosine learning rate decay with warmup [10, 22]. In order to improve the accuracy, we adopted the progressive quantization method [42] that gradually decreases the bit-width to 8, 5, and 4 bits during fine-tuning, and use knowledge distillation [12] using the ResNet-101 as teacher. We used an exponential moving average of parameters with a momentum of 0.9997 [16], and all networks were trained using PROFIT and DuQ (with negative padding if applicable). We trained the model for 15 epochs at every progressive quantization and PROFIT fine-tuning step. The entire fine-tuning for the 4-bit network took 140 epochs for weight update. Please note that both weights and activations on all the layers of the networks were quantized, including the first and last layers. We did not apply quantization only for the input image of the first convolution layer and the activation of the squeeze-excitation module.

5.1 Accuracy on ImageNet Dataset

In Table 1, the accuracy of quantized networks under our proposed methods are shown. In the table, 'Full' represents our reproduction of the original training procedure. 'Full+' adopts teacher-student and weight averaging algorithms on top of 'Full'. Compared to the full precision accuracy ('Full+' in the table), our 4-bit models give comparable accuracy at a top-1 accuracy loss of 1.48%. To the best of the authors' knowledge, this is the SOTA result on the 4-bit quantization of MobileNet-v3 including the first and last layers. From the table, it is also evident that compared with full-precision ('Full+'), our method loses only less than 0.5% of top-1 accuracy on 4-bit MobileNet-v1 and v2.

Table 1: Top-1 / Top-5 accuracy [%] of the quantized networks on ImageNet.

	MobileNet-v1	**MobileNet-v2**	**MobileNet-v3**	**MNasNet-A1**
Full	68.848 / 88.740	71.328 / 90.016	74.728 / 92.136	73.130 / 91.276
Full+	69.552 / 89.138	71.944 / 90.470	75.296 / 92.446	73.396 / 91.464
8-bit	70.164 / 89.370	72.352 / 90.636	75.166 / 92.498	73.742 / 91.756
5-bit	69.866 / 89.058	72.192 / 90.498	74.690 / 92.092	73.378 / 91.244
4-bit	69.056 / 88.412	71.564 / 90.398	73.812 / 91.588	72.244 / 90.584

5.2 Comparison with the Existing Works

We compare the accuracy of MobileNet-v1 and v2 with the existing works in Table 2 where (a,w) represent a-bit activation and w-bit weight quantization and c and l channel-wise and layer-wise quantization, respectively. * represents the post-training quantization. Compared to [18], our 4-bit layer-wise quantization gives comparable accuracy to the 8-bit models of previous work, and better accuracy than the channel-wise 8/4-bit models. Furthermore, our 4-bit MobileNet-v2 model outperforms the previous best 4-bit model [9] by 6.76 %, and it outperforms even the 8-bit models of existing works [8, 24].

5.3 Ablation Study

In the ablation study, we compared the existing methods and our proposed method on the 4-bit quantization of MobileNet-v3 on ImageNet. Moreover, to decompose the effect of each of our methods, we evaluated the effect on the existing method and our own method. Note that we use DuQ with negative padding by default except DuQ, Sym (zero-padding with symmetric quantization) and DuQ, Non (zero-padding with non-negative quantization). We used two existing methods, QIL [17] and PACT with SAWB [3]. When we applied the QIL algorithm to 4-bit MobileNet-v3, the fine-tuning failed to converge. It is because QIL has a critical limitation that its output range is bounded from 0 to 1, which is detrimental to the squeeze-excitation layer and h-swish activation function. As shown in Table 3, PACT gives 70.16 % of top-1 accuracy with a 5.14 % accuracy drop from full-precision accuracy ('Full+' in Table 1). Our proposed methods are beneficial to the existing method. By applying PROFIT to PACT, we can obtain 2.78 % accuracy improvement, as shown in the table. DuQ outperforms PACT by 1.53 % (=69.504 % - 67.978 %). Under teacher-student and progressive quantization, our solution (DuQ + PROFIT) gives 3.65 % (=73.812 % - 70.160 %) accuracy gain over PACT.

In order to evaluate the effect of incremental weight freezing, we also applied only the last stage of PROFIT that stabilizes the normalization layers by fine-tuning after freezing all the convolution layers. This option ($\triangle$ in the table) gives approximately half the gain of PROFIT, which confirms that PROFIT is essential in minimizing the effect of AIWQ.

Table 2: Top-1 accuracy [%] comparison of existing works on MobileNet-v1 (MV1) and MobileNet-v2 (MV2).

	MV1	**MV2**
(8,8), c [18]	70.7	71.1
(8,8), l [18]	70.0	70.9
(8,4), c [18]	64.0	58.0
(4,8), c [18]	65.0	62.0
(4,4), l [9]	-	64.80
(8,8), l [8]*	70.10	70.60
(8,8), l [24]*	70.5	71.2
(5,5), l, Ours	69.866	72.192
(4,4), l, Ours	69.056	71.564

Table 3: Top-1 / Top-5 accuracy [%] of 4-bit MobileNet-v3 on ImageNet. TS represents teacher-student, PG progressive quantization and PF PROFIT.

Algorithm	**TS**	**PG**	**PF**	**Accuracy**
QIL				fail
PACT				67.978 / 88.204
PACT	✓	✓		70.160 / 89.576
PACT	✓	✓	✓	72.948 / 90.892
DuQ				69.504 / 88.946
DuQ	✓			71.006 / 90.018
DuQ	✓	✓		71.466 / 90.200
DuQ	✓	✓	✓	73.812 / 91.588
DuQ			✓	72.260 / 90.772
DuQ	✓	✓	△	72.826 / 91.068
DuQ, Sym				68.480 / 88.496
DuQ, Non				68.724 / 88.566
PACT3b	✓	✓		57.086 / 80.898
PACT3b	✓	✓	✓	66.458 / 87.360
DuQ3b	✓	✓		65.674 / 86.480
DuQ3b	✓	✓	✓	69.942 / 89.340

The table also shows the effect of negative padding. Comparing DuQ (with negative padding) with DuQ, Sym and DuQ, Non, negative padding gives 1.02 % and 0.78 % of accuracy improvement, respectably. In addition, with negative padding, 27.5 % of activations of h-swish output can be mapped to zero, thus the conventional zero-skipping accelerator can additionally improve performance and energy efficiency without the modification of the network. When we reduce the bit-width down to 3-bit (PACT3b and DuQ3b in Table 3), our proposed methods bring more accuracy improvement. Compared to the conventional PACT, our methods (DuQ+PROFIT) give 12.86 % (=69.942 %-57.086 %) improvement.

The conventional models also suffer from the AIWQ problem in more aggressive quantization, and PROFIT is helpful to improve accuracy. With PROFIT, ResNet-18 can be quantized into 3-bit without accuracy loss and 2-bit with 2.31 % accuracy loss compared to the baseline model accuracy, which is the state-of-the-art result as far as we know.

5.4 Computation Cost and Model Size Analysis

We compared the computation overhead and memory efficiency over the accuracy of the quantized network (Figure 8). We used the bit-operations (BOPS) metric [2] that estimates the computation cost based on the required silicon area of the hardware accelerator for the quantized network. In terms of computation,

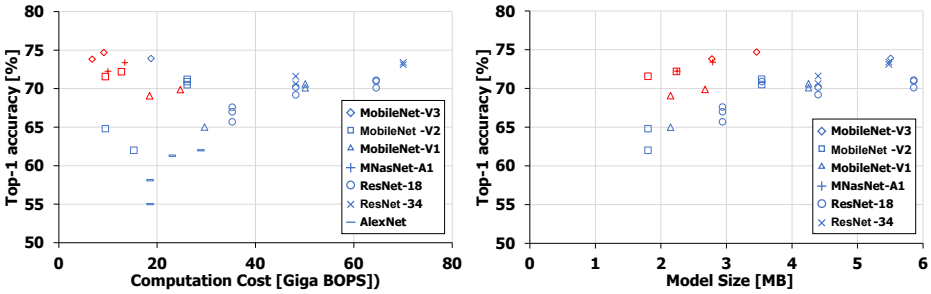


Fig. 8: Comparison of accuracy and estimated computation cost based on the HW accelerator model (the bit-operations (BOPS), [2]), and comparison of accuracy and model size of the quantized network. The tuple (a,w) represents the bit-width of activation and weight, respectively. The red markers represent our results, and the blue markers the results of state-of-the-art methods [3, 7–9, 13, 17, 18, 24].

our quantized model offers much higher efficiency with the same accuracy. For instance, at accuracy higher than 73 %, our 4-bit MobileNet-v3 model takes 2.77× less computation cost than the previous best 8-bit MobileNet-v3 model [13]. Compared to 3-bit ResNet-34 [7], we reduce the cost by up to 10.3×. In terms of the model size, our model gives higher accuracy within the same memory constraint. For instance, the 4-bit MobileNet-v2 model shows 6.76 % higher accuracy than the previous best model (4-bit MobileNet-v2 [9]) within the model size constraint of 2 MB, and the 4-bit MobileNet-v3 model gives 6.21 % higher accuracy (versus 2-bit ResNet-18 [7]) at 3 MB. These benefits come from our proposed method and the efficiency of the advanced network structure.

6 Conclusion

We proposed a novel training method called PROFIT, a quantization method called DuQ, and negative padding. PROFIT aims at minimizing the effect of activation instability induced by weight quantization, and DuQ and negative padding enable the quantization of asymmetric distribution in optimized networks. Based on the proposed methods, we can quantize the optimized networks into 4 bits with less than 1.5 % (MobileNet-v3) and 0.5 % (MobileNet-v1/2) of top-1 accuracy loss. We anticipate that our proposed methods can contribute to advancing towards sub-4-bit precision computation on mobile and edge devices.

Acknowledgement

This work was supported by Samsung Electronics, the National Research Foundation of Korea grants, NRF-2016M3A7B4909604 and NRF-2016M3C4A7952587 funded by the Ministry of Science, ICT & Future Planning (PE Class Heterogeneous High Performance Computer Development). We appreciate valuable comments from Dr. Andrew G. Howard and Dr. Jaeyoun Kim at Google.

References

1. Albericio, J., Judd, P., Hetherington, T.H., Aamodt, T.M., Jerger, N.D.E., Moshovos, A.: Cnvlutin: Ineffectual-neuron-free deep neural network computing. International Symposium on Computer Architecture (ISCA) (2016)
2. Baskin, C., Schwartz, E., Zheltonozhskii, E., Liss, N., Giryes, R., Bronstein, A.M., Mendelson, A.: Uniq: Uniform noise injection for non-uniform quantization of neural networks. arXiv:1804.10969 (2018)
3. Choi, J., Chuang, P.I., Wang, Z., Venkataramani, S., Srinivasan, V., Gopalakrishnan, K.: Bridging the accuracy gap for 2-bit quantized neural networks. arXiv:1807.06964 (2018)
4. Choi, J., Wang, Z., Venkataramani, S., Chuang, P.I., Srinivasan, V., Gopalakrishnan, K.: Pact: Parameterized clipping activation for quantized neural networks. arXiv:1805.06085 (2018)
5. Deng, J., Dong, W., Socher, R., Li, L., Li, K., Li, F.: Imagenet: A large-scale hierarchical image database. Computer Vision and Pattern Recognition (CVPR) (2009)
6. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT) (2019)
7. Esser, S.K., McKinstry, J.L., Bablani, D., Appuswamy, R., Modha, D.S.: Learned step size quantization. arXiv:1902.08153) (2019)
8. Finkelstein, A., Almog, U., Grobman, M.: Fighting quantization bias with bias. arXiv:1906.03193 (2019)
9. Gong, R., Liu, X., Jiang, S., Li, T., Hu, P., Lin, J., Yu, F., Yan, J.: Differentiable soft quantization: Bridging full-precision and low-bit neural networks. arXiv:1908.05033 (2019)
10. Goyal, P., Dollár, P., Girshick, R.B., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., He, K.: Accurate, large minibatch SGD: training imagenet in 1 hour. arXiv:1706.02677 (2017)
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. Computer Vision and Pattern Recognition (CVPR) (2016)
12. Hinton, G.E., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv:1503.02531 (2015)
13. Howard, A., Sandler, M., Chu, G., Chen, L., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Q.V., Adam, H.: Searching for mobilenetv3. International Conference on Computer Vision (ICCV) (2019)
14. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv:1704.04861 (2017)
15. Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., Bengio, Y.: Quantized neural networks: Training neural networks with low precision weights and activations. Journal of Machine Learning Research (JMLR) (2017)
16. Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D.P., Wilson, A.G.: Averaging weights leads to wider optima and better generalization. Uncertainty in Artificial Intelligence (UAI) (2018)
17. Jung, S., Son, C., Lee, S., Son, J., Han, J., Kwak, Y., Hwang, S.J., Choi, C.: Learning to quantize deep networks by optimizing quantization intervals with task loss. Computer Vision and Pattern Recognition (CVPR) (2019)

18. Krishnamoorthi, R.: Quantizing deep convolutional networks for efficient inference: A whitepaper. arXiv:1806.08342 (2018)
19. Krizhevsky, A., Nair, V., Hinton, G.: The cifar-10 dataset. https://www.cs.toronto.edu/~kriz/cifar.html (2014), accessed: 2020-03-03
20. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Neural Information Processing Systems (NeurIPS) (2012)
21. Liu, Z., Wu, B., Luo, W., Yang, X., Liu, W., Cheng, K.: Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. European Conference on Computer Vision (ECCV) (2018)
22. Loshchilov, I., Hutter, F.: SGDR: stochastic gradient descent with warm restarts. International Conference on Learning Representations (ICLR) (2017)
23. Mishra, A.K., Nurvitadhi, E., Cook, J.J., Marr, D.: WRPN: wide reduced-precision networks. International Conference on Learning Representations (ICLR) (2018)
24. Nagel, M., van Baalen, M., Blankevoort, T., Welling, M.: Data-free quantization through weight equalization and bias correction. arXiv:1906.04721 (2019)
25. Int4 precision for ai inference. https://devblogs.nvidia.com/int4-for-ai-inference/ (2019), accessed: 2020-03-03
26. Park, E., Kim, D., Yoo, S.: Energy-efficient neural network accelerator based on outlier-aware low-precision computation. International Symposium on Computer Architecture (ISCA) (2018)
27. Rastegari, M., Ordonez, V., Redmon, J., Farhadi, A.: Xnor-net: Imagenet classification using binary convolutional neural networks. European Conference on Computer Vision (ECCV) (2016)
28. Samsung low-power npu solution for ai deep learning. https://news.samsung.com/global/samsung-electronics-introduces-a-high-speed-low-power-npu-solution-for-ai-deep-learning (2019), accessed: 2020-03-03
29. Sandler, M., Howard, A.G., Zhu, M., Zhmoginov, A., Chen, L.: Mobilenetv2: Inverted residuals and linear bottlenecks. Computer Vision and Pattern Recognition (CVPR) (2018)
30. Sharma, H., Park, J., Suda, N., Lai, L., Chau, B., Kim, J.K., Chandra, V., Esmaeilzadeh, H.: Bit fusion: Bit-level dynamically composable architecture for accelerating deep neural networks. International Symposium on Computer Architecture (ISCA) (2018)
31. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. International Conference on Learning Representations (ICLR) (2015)
32. Snapdragon neural processing engine sdk. https://developer.qualcomm.com/docs/snpe/index.html (2017), accessed: 2020-03-03
33. Song, J., Cho, Y., Park, J.S., Jang, J.W., Lee, S., Song, J.H., Lee, J.G., Kang, I.: 7.1 an 11.5 tops/w 1024-mac butterfly structure dual-core sparsity-aware neural processing unit in 8nm flagship mobile soc. International Solid-State Circuits Conference (ISSCC) (2019)
34. Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., Le, Q.V.: Mnasnet: Platform-aware neural architecture search for mobile. Computer Vision and Pattern Recognition (CVPR) (2019)
35. Tulloch, A., Jia, Y.: High performance ultra-low-precision convolutions on mobile devices. arXiv:1712.02427 (2017)
36. Tulloch, A., Jia, Y.: Quantization and training of neural networks for efficient integer-arithmetic-only inference. Conference on Computer Vision and Pattern Recognition (CVPR) (2018)

37. Wang, T., Xiong, J., Xu, X., Shi, Y.: SCNN: A general distribution based statistical convolutional neural network with application to video object detection. Association for the Advancement of Artificial Intelligence (AAAI) (2019)
38. Wu, H.: NVIDIA Low Precision Inference on GPU. GPU Technology Conference (2019)
39. Zhou, S., Ni, Z., Zhou, X., Wen, H., Wu, Y., Zou, Y.: Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. arXiv:1606.06160 (2016)
40. Zhou, S., Wang, Y., Wen, H., He, Q., Zou, Y.: Balanced quantization: An effective and efficient approach to quantized neural networks. Journal of Computer Science and Technology (2017)
41. Zhu, C., Han, S., Mao, H., Dally, W.J.: Trained ternary quantization. INternational Conference on Learning Representations (ICLR) (2017)
42. Zhuang, B., Shen, C., Tan, M., Liu, L., Reid, I.D.: Towards effective low-bitwidth convolutional neural networks. Computer Vision and Pattern Recognition (CVPR) (2018)