Sharpness-aware Quantization for Deep Neural Networks

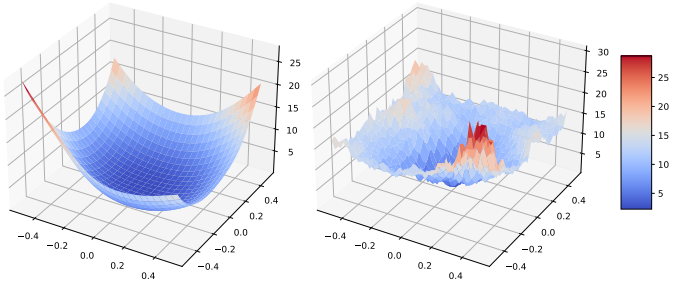
Jing Liu Jianfei Cai Bohan Zhuang†
Data Science & AI, Monash University, Australia

Abstract

Network quantization is an effective compression method to reduce the model size and computational cost. Despite the high compression ratio, training a low-precision model is difficult due to the discrete and non-differentiable nature of quantization, resulting in considerable performance degradation. Recently, Sharpness-Aware Minimization (SAM) is proposed to improve the generalization performance of the models by simultaneously minimizing the loss value and the loss curvature. In this paper, we devise a Sharpness-Aware Quantization (SAQ) method to train quantized models, leading to better generalization performance. Moreover, since each layer contributes differently to the loss value and the loss sharpness of a network, we further devise an effective method that learns a configuration generator to automatically determine the bitwidth configurations of each layer, encouraging lower bits for flat regions and vice versa for sharp landscapes, while simultaneously promoting the flatness of minima to enable more aggressive quantization. Extensive experiments on CIFAR-100 and ImageNet show the superior performance of the proposed methods. For example, our quantized ResNet-18 with 55.1× Bit-Operation (BOP) reduction even outperforms the full-precision one by 0.7% in terms of the Top-1 accuracy. Code is available at https://github.com/zhuang-group/SAQ.

1. Introduction

With powerful high performance computing and large amounts of labeled data, convolutional neural networks (CNNs) have dramatically improved the accuracy of many visual tasks, such as image classification [30] and dense prediction [43], to the level of being ready for real-world applications. Despite the great breakthroughs that deep learning has achieved, the considerable computational overhead and model size greatly hampers the development and deployment of deep learning techniques at scale, especially on resource-limited devices. To obtain lightweight compact models, many network quantization methods [23, 62] have been proposed to tackle the efficiency bottlenecks.



(a) Full-precision ResNet-20 (b) 2-bit ResNet-20

Figure 1. The loss landscapes of the full-precision and 2-bit ResNet-20 on CIFAR-100. We plot the loss landscapes using the visualization methods in [32].

Despite the high compression ratio, training a low-precision model is very challenging due to the discrete and non-differentiable nature of network quantization. Compared with the full-precision ones, the low-precision models represent weights, activations, and even gradients with only a small set of values, which limits the representation power of the quantized models and result in a sharp loss landscape [37]. As illustrated in Figure 1, the converged quantized model reveals much sharper local minima than the full-precision counterpart. In this case, a small change in weights coming from the quantization noise or gradient update may cause severe loss fluctuations in sharp regions, making more gradients unreliable during optimization. As a consequence, it reflects the degraded generalization capability for the low-precision models in light of the higher probability to get trapped in poor local minima, leading to a significant performance drop.

There have been some studies showing that the flatness of minima of the loss function found by stochastic gradient-based methods results in good generalization [14, 21, 24, 27]. Recently, Sharpness-aware Minimization (SAM) [16] is proposed to simultaneously minimize loss value and loss sharpness. To reduce the difficulty of training and improve the generalization ability of the low-precision models, it is straightforward to apply SAM for training.

†Corresponding author. Email: bohan.zhuang@monash.edu

However, the introduced small perturbations in SAM might be diminished by the discretization process in network quantization. As a result, SAM might be reduced to the trivial loss minimization in this case.

To tackle the above challenges, we devise a novel training method, called Sharpness-Aware Quantization (SAQ), to improve the generalization performance of the low-precision models. To this end, we propose to introduce small perturbations on the quantized weights instead of the full-precision ones. Specifically, our proposed SAQ first finds a perturbation on the quantized weights that maximizes the change of loss and then updates the full-precision weights to minimize the perturbed loss. In this way, we can simultaneously minimize the loss value and smooth the loss landscapes of the quantized models, which greatly improves the generalization performance of the quantized models.

Note that different layers contribute differently to the loss value and the loss sharpness of a network. According to the Minimum Description Length (MDL) theory [22,47], fewer bits are required to sufficiently model flat minima and vice versa for sharp minima. Motivated by this, we further propose Sharpness-Aware Mixed-precision Quantization (SAMQ) method to automatically search the bitwidth of each layer considering the loss value and the loss sharpness of the quantized models. To be specific, our proposed SAMQ learns a configuration generator using reinforcement learning, where the reward function is designed to take the loss value, the loss sharpness, and the computational constraint into consideration. By training the configuration generator in conjunction with network parameters, our proposed method encourages the resulting bitwidth configurations to lie in flatter loss landscapes and thus enables more aggressive quantization with low computational cost. Extensive experiments on CIFAR-100 and ImageNet show the promising performance of our proposed method.

Our main contributions are summarized as follows:

- We propose a Sharpness-Aware Quantization (SAQ) method to simultaneously minimize the loss value and the loss landscape of the quantized models. In this way, the generalization performance of the quantized models can be greatly improved.
- We propose Sharpness-Aware Mixed-precision Quantization (SAMQ) to learn a configuration generator to automatically determine the optimal bitwidth of each layer, which is learned in conjunction with other network parameters, encouraging flatter loss landscapes of the quantized model.
- We evaluate our proposed SAQ and SAMQ on CIFAR-100 and ImageNet. Extensive experiments on various architectures show that our method performs favorably against SOTA quantization methods. For example, our quantized ResNet-18 with 55.1× Bit-Operation (BOP) reduction even surpasses the full-precision one by 0.7% on the Top-1 accuracy.

2. Related work

Network quantization. Network quantization seeks to reduce the model size and computational cost by mapping weights, activations, and even gradients to low-precision ones. Existing quantization methods can be roughly divided into two categories according to the quantization bitwidth, namely, fixed-point quantization [4, 8, 15, 26, 60, 62, 63] and binary quantization [23, 34, 37, 39, 46, 64]. To improve the quantization performance, existing methods [8, 15, 26, 55, 60] explicitly parameterize the quantizer and train it jointly with network parameters. To reduce the optimization difficulty incurred by the non-differentiable discretization, extensive methods [10, 17, 56] have been proposed to approximate the gradients. Moreover, when it comes to binary quantization, some studies [2, 37, 38, 41] show that Adam [28] performs better than SGD [48] for optimization. Compared with these methods, our SAQ focuses on improving the generalization performance of the quantized models from a new perspective by minimizing both the loss value and the loss sharpness.

Considering that different layers contribute differently to the accuracy and computational cost of a network, several studies proposed mixed-precision quantization [5, 7, 11, 51–54, 57] that assigns different bitwidths according to the sensitivity of each layer. Compared with these methods, our SAMQ learns a configuration generator to determine the bitwidth of each layer considering the loss value and the loss sharpness of the quantized models. Compared with HAWQ [11, 57] that uses second-order Hessian information to measure the quantization sensitivity of each layer, our proposed method differs in two aspects. 1) Our method not only encourages lower bits for flat minima and vice versa for sharp regions during configuration search but also promotes the flatter minima to improve the generalization performance while HAWQ only supports the former one. 2) Our quantized model with lower BOPs outperforms those of HAWQ by a large margin on ImageNet (See Table 2).

Loss geometry and generalization. Hochreiter *et al.* [21] pioneered the proposition that flat local minima generalizes better in neural networks. Following that, several studies have been proposed to investigate the relation between the geometry of the loss landscape and generalization performance of the models [6, 14, 24, 27, 35, 42, 50]. Recently, sharpness-aware minimization (SAM) [16] seeks to find parameters that lie in a region with both low loss value and loss sharpness and shows promising performance across various architectures and benchmark datasets. Moreover, several methods have been proposed to improve the performance [31] or efficiency [13] of SAM. Specifically,

ASAM [31] introduces a concept of adaptive sharpness to mitigate the effect of parameter re-scaling while ESAM [13] reduces the computational overhead without performance drop. Compared with these existing methods, our proposed SAQ focuses on improving the generalization performance of the quantized models. We further use the loss value and the loss sharpness of the quantized models to guide the bitwidth configuration search of each layer.

3. Preliminary

3.1. Network quantization

For convenience, we revisit the uniform quantization function. For a CNN with L layers, let w^l and z^l be the weights and input activations of a convolutional layer l. For simplicity, we omit the layer index l. Before performing quantization, we first use normalization functions $T_w(\cdot)$ and $T_z(\cdot)$ to map w and z into the scale of $[0, 1]$, respectively:

$$\hat{w} = T_w(w) = \frac{1}{2}\left(\text{clip}\left(\frac{w}{\alpha_w}, -1, 1\right) + 1\right), \quad (1)$$

$$\hat{z} = T_z(z) = \text{clip}\left(\frac{x}{\alpha_z}, 0, 1\right), \quad (2)$$

where α_w and α_z are trainable clipping levels that limit the range of weights and activations. Here, we use the function $\text{clip}(v, v_{\min}, v_{\max})$ to clamp any value v into the range of $[v_{\min}, v_{\max}]$. We then use the following function to quantize the normalized value $\hat{v} \in \{\hat{w}, \hat{z}\}$ to the discrete one $\bar{v}$:

$$\bar{v} = D(\hat{v}, s) = \frac{\lfloor \hat{v} \cdot s \rceil}{s}, \quad (3)$$

where $\lfloor \cdot \rceil$ is a rounding operator that returns the nearest integer of a given value and $s = 2^b - 1$ is the number of quantization levels (except zero) for b-bit quantization. Last, we obtain the quantized w and z by

$$Q_w(w, b) = T_w^{-1}(D(\hat{w}, s)) = \alpha_w \cdot (2D(\hat{w}, s) - 1), \quad (4)$$

$$Q_z(z, b) = T_z^{-1}(D(\hat{z}, s)) = \alpha_z \cdot D(\hat{z}, s), \quad (5)$$

where $T_w^{-1}(\cdot)$ and $T_z^{-1}(\cdot)$ are the inverse functions of $T_w(\cdot)$ and $T_z(\cdot)$, respectively. During training, the rounding operation $\lfloor \cdot \rceil$ is non-differentiable. Following [23, 62], we apply the straight through estimation (STE) [1] to approximate the gradient of the rounding operator for backward propagation.

3.2. Sharpness-aware minimization

Without loss of generality, let $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ be the training data and $\mathbf{w}$ be the trainable parameters of a model. The goal of model training is to minimize the empirical risk $\mathcal{L}_S(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^n l(\mathbf{w}, \mathbf{x}_i, y_i)$, where $l(\mathbf{w}, \mathbf{x}_i, y_i)$ is a loss function for the sample $(\mathbf{x}_i, y_i)$. Instead of seeking a single minimum with low loss, sharpness-aware minimization [16] (SAM) seeks a region that has uniformly low training loss (both low loss and flat curvature). Specifically, the formulation of SAM is a min-max optimization problem which is defined as

$$\min_{\mathbf{w}} \max_{\|\boldsymbol{\epsilon}\|_2 \le \rho} \mathcal{L}_S(\mathbf{w} + \boldsymbol{\epsilon}) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2, \quad (6)$$

where λ is a coefficient of ℓ_2 regularization term; the perturbed loss $\mathcal{L}_S(\mathbf{w} + \boldsymbol{\epsilon})$ is the sum of empirical risk $\mathcal{L}_S(\mathbf{w})$ and loss sharpness $\mathcal{L}_S(\mathbf{w}+\boldsymbol{\epsilon}) - \mathcal{L}_S(\mathbf{w})$. In Eq. (6), the inner optimization problem attempts to find a weight perturbation $\boldsymbol{\epsilon}$ in an l^2 Euclidean ball with radius ρ that maximizes the perturbed loss $\mathcal{L}_S(\mathbf{w}+\boldsymbol{\epsilon})$. To solve the inner problem, SAM uses a one-step gradient ascent as

$$\hat{\boldsymbol{\epsilon}} = \arg\max_{\|\boldsymbol{\epsilon}\|_2 \le \rho} \mathcal{L}_S(\mathbf{w} + \boldsymbol{\epsilon}) \approx \rho \frac{\nabla \mathcal{L}_S(\mathbf{w})}{\|\nabla \mathcal{L}_S(\mathbf{w})\|_2}. \quad (7)$$

By substituting back into Eq. (6), we then have the following optimization problem:

$$\min_{\mathbf{w}} \mathcal{L}_S(\mathbf{w} + \hat{\boldsymbol{\epsilon}}) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2. \quad (8)$$

4. Proposed method

To improve the generalization performance, we propose a Sharpness-Aware Quantization (SAQ) method to simultaneously minimize the loss value and the loss sharpness of the quantized models (See Section 4.1). We further propose Sharpness-Aware Mixed-precision Quantization (SAMQ) to automatically find the optimal bitwidth configuration of each layer considering their different characteristics (See Section 4.2).

4.1. Sharpness-aware quantization

Let $\mathbf{b} = \{b^1, b^2, \ldots, b^L\} \in \mathcal{B}$ be a bitwidth configuration where b^l is the bitwidth for layer l and $\mathcal{B}$ is the bitwidth configuration space. As shown in Figure 1, the low-precision models show a much sharper loss landscape compared with the full-precision models. In this case, a small perturbation on the weights may cause severe loss fluctuations, making the gradients unstable during training. As a result, the quantized models are more likely to get trapped in poor local minima, resulting in performance degradation. To smooth the loss landscape and improve the generalization performance, one may apply SAM to train the quantized models and formulate the optimization problem as follows:

$$\min_{\mathbf{w}} \max_{\|\boldsymbol{\epsilon}\|_2 \le \rho} \mathcal{L}_S(Q_w(\mathbf{w} + \boldsymbol{\epsilon}, \mathbf{b})) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2, \quad (9)$$

where we apply the quantization function $Q_w(\mathbf{w} + \boldsymbol{\epsilon}, \mathbf{b})$ on the perturbed full-precision weights $\mathbf{w} + \boldsymbol{\epsilon}$ with the bitwidth

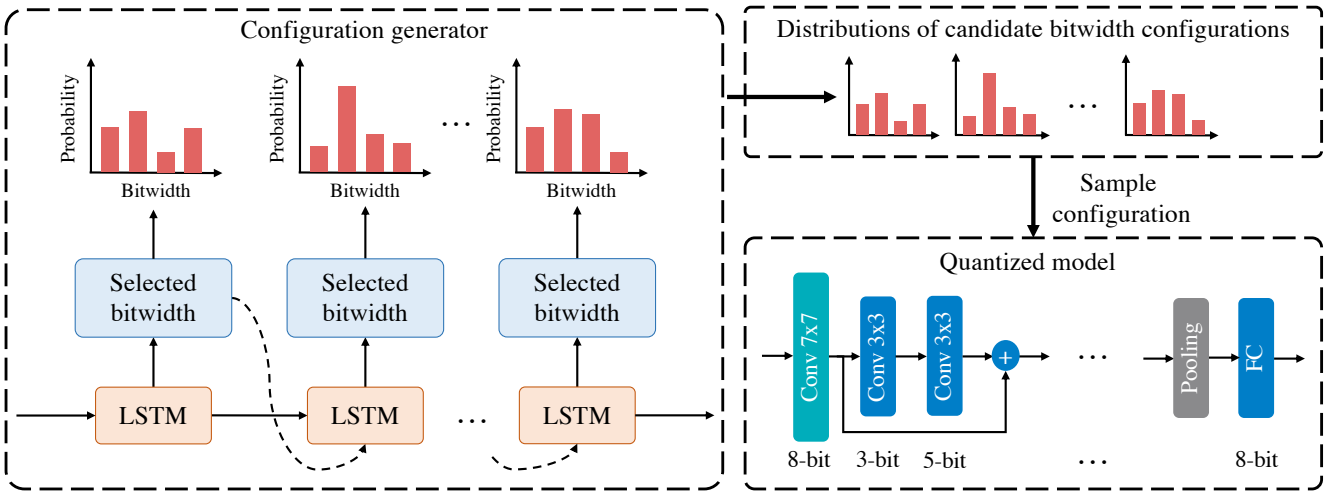


Figure 2. An overview of the configuration generator. We formulate the bitwidth search problem as a sequential prediction problem where each element of a sequence denotes the bitwidth configuration of the corresponding layer. The configuration generator is an LSTM network with a fully-connected layer that takes an empty embedding as input and outputs the bitwidth distributions of different layers. We then perform sampling from the learned distributions to obtain a bitwidth configuration that satisfies the computational constraint.

configuration $\mathbf{b}$. Nevertheless, due to the discretization process (*i.e.*, rounding operation) in Eq. (3), the small perturbation $\boldsymbol{\epsilon}$ introduced by SAM may not change the resulting discrete weights, *i.e.*, $Q_w(\mathbf{w}+\boldsymbol{\epsilon},\mathbf{b}) = Q_w(\mathbf{w},\mathbf{b})$. To address this issue, instead of adding perturbation to the full-precision weights in Eq. (9), we propose to introduce small perturbations on the quantized weights. Specifically, the objective function of Sharpness-Aware Quantization (SAQ) is defined as

$$\min_{\mathbf{w}} \max_{\|\boldsymbol{\epsilon}\|_2 \leq \rho} \mathcal{L}_S(Q_w(\mathbf{w},\mathbf{b})+\boldsymbol{\epsilon}) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2. \tag{10}$$

Given $\mathbf{w}$ and $\mathbf{b}$, the inner optimization problem attempts to find a perturbation $\boldsymbol{\epsilon}$ within a ball around the quantized weight $Q_w(\mathbf{w},\mathbf{b})$ with the radius ρ to maximize the perturbed loss. In order to solve the inner optimization problem, following SAM [16], we apply one-step gradient ascent as

$$\begin{aligned}\hat{\boldsymbol{\epsilon}} &= \arg\max_{\|\boldsymbol{\epsilon}\|_2 \leq \rho} \mathcal{L}_S(Q_w(\mathbf{w},\mathbf{b})+\boldsymbol{\epsilon}) \\ &\approx \rho \frac{\nabla \mathcal{L}_S\left(Q_w(\mathbf{w},\mathbf{b})\right)}{\|\nabla \mathcal{L}_S\left(Q_w(\mathbf{w},\mathbf{b})\right)\|_2}.\end{aligned} \tag{11}$$

After solving the inner optimization problem, SAQ updates the full-precision weights $\mathbf{w}$ based on the perturbed loss gradient $\nabla_{\mathbf{w}} \mathcal{L}_S(Q_w(\mathbf{w},\mathbf{b})+\hat{\boldsymbol{\epsilon}})$ using some optimizers. By minimizing the perturbed loss, SAQ enforces the quantized models to find minima with both low loss value and low loss curvature and thus yields better generalization performance.

4.2. Sharpness-aware mixed-precision quantization

To achieve a high compression ratio, one may set each layer with a uniform ultra-low bitwidth (*e.g.*, $b^1 = b^2 = \ldots = b^L = 3$). However, different layers of a network can have different sensitivity to network quantization. Therefore, setting all layers to a uniform ultra-low bitwidth may significantly degrade the performance of the quantized models. To reduce the performance drop, one may use mixed-precision quantization that assigns different bitwidths according to the quantization sensitivity of each layer [11, 57]. However, finding a suitable criterion to measure the sensitivity of each layer is not trivial.

Note that different layers contribute differently to the loss value and the loss sharpness of a quantized model. It has been shown that fewer bits are required to encode flat curvature and vice versa for sharp minima in the Minimum Description Length (MDL) theory [22, 47]. Based on this intuition, we assume and highlight that the perturbed loss is a good metric to measure the quantization sensitivity of different layers as it takes both the loss value and the loss sharpness into consideration. Motivated by this, we propose a Sharpness-Aware Mixed-precision Quantization (SAMQ) method to automatically determine the bitwidth configuration of each layer based on the perturbed loss.

Given a computational constraint C, we seek to learn a configuration generator using reinforcement learning to obtain a bitwidth configuration $\mathbf{b}$, and its computational cost satisfies the constraint $c(\mathbf{b}) \leq C$, as shown in Figure 2. Following [18, 44], we formulate the bitwidth search problem as a sequential prediction problem where each element of a sequence is the bitwidth configuration of the corresponding layer. Our configuration generator $G(\boldsymbol{\theta})$ is based on an LSTM network with a fully-connected layer to predict the bitwidth of each layer, where $\boldsymbol{\theta}$ is the parameters of $G(\cdot)$. At the first step, the configuration generator receives

an empty embedding as input. Then, the decision at the current step is fed as input embedding into the next step. Last, we can obtain a candidate bitwidth configuration by sampling $\mathbf{b} \sim \pi(\boldsymbol{\theta})$, where $\pi(\cdot)$ is a policy learned by the generator, *i.e.*, bitwidth distributions of all layers.

To learn the configuration generator, following NAS [65] and ENAS [44], we formulate the bitwidth configuration search as a bi-level optimization problem as

$$\begin{aligned} \min_{\boldsymbol{\theta}} \ & \mathbb{E}_{\mathbf{b}\sim\pi(\boldsymbol{\theta})}\left[R(\mathbf{w}^*(\mathbf{b}),\mathbf{b})\right] - \alpha H(\pi(\boldsymbol{\theta})) \\ & \text{s.t. } \mathbf{w}^*(\mathbf{b}) = \arg\min_{\mathbf{w}} \mathcal{L}_S(Q_w(\mathbf{w},\mathbf{b}) + \hat{\boldsymbol{\epsilon}}), \end{aligned} \tag{12}$$

where $R(\mathbf{w}^*(\mathbf{b}),\mathbf{b})$ is a reward function, $H(\cdot)$ is an entropy term that encourages the bitwidth diversity and α is a hyper-parameter that controls the strength of the entropy term. To guide the training of the configuration generator, we hope the quantized model with the obtained bitwidth configuration not only satisfies the computational constraint but also has a flat landscape with both low loss value and loss sharpness. By considering the loss value, the loss sharpness and the computational constraint, we have the reward as

$$R(\mathbf{w}^*(\mathbf{b}),\mathbf{b}) = \mathcal{L}_V(Q_w(\mathbf{w}^*(\mathbf{b}),\mathbf{b}) + \tilde{\boldsymbol{\epsilon}}) + \beta\left[c(\mathbf{b}) - C\right]^2, \tag{13}$$

where β is a hyper-parameter that makes a balance between the perturbed loss and the computational cost. Here, $\tilde{\boldsymbol{\epsilon}}$ is quantized weight perturbation computed by

$$\begin{aligned} \tilde{\boldsymbol{\epsilon}} &= \arg\max_{\|\boldsymbol{\epsilon}\|_2 \le \rho} \mathcal{L}_V(Q_w(\mathbf{w}^*(\mathbf{b}),\mathbf{b}) + \boldsymbol{\epsilon}) \\ &\approx \rho \frac{\nabla \mathcal{L}_V\left(Q_w(\mathbf{w}^*(\mathbf{b}),\mathbf{b})\right)}{\|\nabla \mathcal{L}_V\left(Q_w(\mathbf{w}^*(\mathbf{b}),\mathbf{b})\right)\|_2}, \end{aligned} \tag{14}$$

where $\mathcal{L}_V(\cdot)$ is an empirical loss function on validation set V. Note that solving Problem (12) directly is computationally expensive since we need to train each sampled bitwidth configuration until convergence. Following [44, 65], we update $\boldsymbol{\theta}$ and $\mathbf{w}$ in an alternating manner to solve the problem. We summarize the training method for SAMQ in Algorithm 1. By training the configuration generator in conjunction with the network parameters, our proposed method encourages the searched configurations to have smooth loss landscapes with low loss value, thus promoting more aggressive quantization.

To accelerate the training process, we take advantage of the parameter sharing [44] technique. Specifically, we obtain the quantized models of different bitwidths by performing quantization on the shared full-precision weights $\mathbf{w}$ following AdaBits [25]. The parameter sharing enables the candidate bitwidth configurations to learn jointly rather than separately, which greatly reduces the computational overhead and optimization difficulty. We also apply Switchable Clipping Level [25] and Switchable Batch Norm [59] that employs independent clipping levels and batch normalization for different bitwidths in each layer.

Algorithm 1 Training method for SAMQ.

Require: The pre-trained full-precision model parameters $\mathbf{w}^f$, low-precision model parameters $\mathbf{w}$, controller parameters $\boldsymbol{\theta}$, configuration's policy $\pi(\cdot)$, the number of training epochs T, and learning rate η.

Initialize $\mathbf{w} = \mathbf{w}^f$.
for $t = 1$ to T **do**
  // Train the configuration generator.
  for each iteration on validation data **do**
    Sample $\mathbf{b} \sim \pi(\boldsymbol{\theta})$.
    Compute $\tilde{\boldsymbol{\epsilon}}$ using Eq. (14).
    Compute the reward function using Eq. (13).
    Update $\boldsymbol{\theta}$ by descending its gradient:
      $R(\mathbf{w}^*(\mathbf{b}),\mathbf{b})\nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta}) - \alpha \nabla_{\boldsymbol{\theta}} H(\pi(\boldsymbol{\theta}))$.
  end for
  // Train the low-precision model.
  for each iteration on training data **do**
    Sample $\mathbf{b} \sim \pi(\boldsymbol{\theta})$.
    Compute $\hat{\boldsymbol{\epsilon}}$ using Eq. (11).
    Update the model parameters by:
      $\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla_{\mathbf{w}} \mathcal{L}_S(Q_w(\mathbf{w},\mathbf{b}) + \hat{\boldsymbol{\epsilon}})$.
  end for
end for

Inferring bitwidth configurations. Once we have trained the configuration generator, we can use it to obtain bitwidth configurations that satisfy the computational constraint. Specifically, following [44, 65], we first sample multiple bitwidth configurations from the learned policy $\pi(\boldsymbol{\theta})$. We will repeat the sampling process if the sampled bitwidth configuration does not satisfy the constraint. Then, we select the bitwidth configuration with the highest validation accuracy. Finally, we fine-tune the quantized model with the selected bitwidth configuration.

5. Experiments

Datasets. We evaluate our method on CIFAR-100 [29] and ImageNet [9]. CIFAR-100 contains 50k training samples and 10k testing images with 100 classes. ImageNet consists of 1.28 million training images and 50k testing samples with 1k classes.

Evaluation metrics. We measure the performance of different methods using the Top-1 and Top-5 accuracy. For the results on CIFAR-100, we repeat the experiments 5 times and report the mean and standard deviation of the Top-1 and Top-5 accuracy. Following [57, 58], we measure the computational cost of the quantized models by Bit-Operation (BOP) count for all the compared methods. We also define the BOP compression ratio as the ratio between the total BOPs of the uncompressed and compressed model.

Implementation details. To investigate the effectiveness of the proposed method, we apply SAQ and SAMQ to different architectures, such as ResNet [20] and Mo-

Table 1. Performance comparisons of different methods on CIFAR-100. We use Bit-Operation (BOP) count to measure the computational cost. "BOP comp. ratio" is the BOP compression ratio. "MP" indicates mixed-precision quantization.

Network	Method	Bitwidth	BOPs (M)	BOP comp. ratio	Top-1 Acc. (%)	Top-5 Acc. (%)
ResNet-20	Full-precision	32	41798.6	1.0	67.5	90.8
	PACT [8]	4	674.6	62.0	66.7±0.2	90.3±0.1
	SAQ (Ours)	4	674.6	62.0	**68.7±0.1**	**91.2±0.0**
	DQ [51]	MP	1180.0	35.4	67.7±0.6	90.4±0.5
	HAQ [52]	MP	673.2	62.1	66.0±0.2	89.9±0.1
	SAMQ (Ours)	MP	**659.3**	**63.4**	**68.7±0.1**	**91.2±0.1**
	PACT [8]	3	392.1	106.6	66.7±0.2	90.3±0.0
	SAQ (Ours)	3	392.1	106.6	**67.7±0.2**	**90.8±0.1**
	DQ [51]	MP	636.2	65.7	64.9±0.2	88.9±0.3
	HAQ [52]	MP	391.5	106.8	65.1±0.4	89.4±0.3
	SAMQ (Ours)	MP	**374.4**	**111.6**	**68.6±0.1**	**91.2±0.0**
MobileNetV2	Full-precision	32	93460.6	1.0	75.1	94.0
	PACT [8]	4	1508.9	61.9	74.6±0.1	93.4±0.2
	SAQ (Ours)	4	1508.9	61.9	**75.6±0.2**	**93.7±0.1**
	DQ [51]	MP	1460.0	64.0	72.1±0.6	92.0±0.3
	HAQ [52]	MP	1513.2	61.8	74.5±0.1	93.1±0.1
	SAMQ (Ours)	MP	**1482.1**	**63.1**	**75.5±0.3**	**93.6±0.2**
	PACT [8]	3	877.1	106.6	74.2±0.1	**93.2±0.1**
	SAQ (Ours)	3	877.1	106.6	**74.4±0.2**	**93.2±0.1**
	DQ [51]	MP	1001.2	93.3	71.6±0.6	92.0±0.2
	HAQ [52]	MP	880.1	106.2	74.0±0.1	93.1±0.1
	SAMQ (Ours)	MP	**869.5**	**107.5**	**75.5±0.1**	**93.7±0.1**

bileNetV2 [49]. We first train the full-precision models and use them to initialize the low-precision ones. Following HAQ [52] and LSQ [15], we quantize both weights and activations for all layers where the first and last layers are quantized to 8-bit. Relying on SGD with the momentum term of 0.9, we apply SAQ for optimization.

For the fixed-precision quantization, we train all the models for 200 epochs with a mini-batch size of 128 on CIFAR-100. The learning rate starts from 0.01 and is divided by 10 at epochs 80 and 120. We set the weight decay term λ to 1×10^{-4}. On ImageNet, we fine-tune 90 epochs for ResNet-18 and 150 epochs for MobileNetV2. The mini-batch size is set to 512. The learning rate is initialized to 0.02 and decreased to 0 following the cosine annealing [40]. We set λ to 1×10^{-4} and 4×10^{-5} for ResNet-18 and MobileNetV2, respectively. For the hyper-parameter ρ, we conduct grid search over $\{0.02, 0.05, 0.1, 0.15, 0.2, \ldots, 0.9, 1.0\}$ to find appropriate values of ρ.

For the mixed-precision quantization, we first apply SAMQ to train the configuration generator. We select the bitwidth of each layer from $\{2, 3, 4, 5\}$. To reduce the search space, the weights and activations of a layer share the same bitwidth. We train the configuration generator and the quantized model for 100 epochs. For the training of the configuration generator, we use Adam [28] with a weight decay term λ of 5×10^{-5}. The learning rate and α are set to 5×10^{-4} and 5×10^{-3}, respectively. For the training of the low-precision models, we train the quantized models with a mini-batch size of 128 on CIFAR-100 and 512 on ImageNet. The learning rate is initialized to 0.01 and divided by 10 at epoch 50. Once we obtain a bitwidth configuration, we fine-tune the resulting quantized model using the same settings as the fixed-precision quantization. We put more implementation details in the supplementary material.

5.1. Main results

We apply SAQ and SAMQ to quantize ResNet-20 and MobileNetV2 and evaluate the performance on CIFAR-100 in Table 1. We further apply our proposed methods to quantize ResNet-18 and MobileNetV2 on ImageNet and report the results in Table 2. We also illustrate the detailed bitwidth configurations of different quantized models in the supplementary material. For the fixed-precision quantization, our SAQ outperforms all the compared methods. For example, on CIFAR-100, our quantized 4-bit ResNet-20 surpasses that of PACT by 2% in terms of the Top-1 accuracy. On ImageNet, 4-bit ResNet-18 obtained by SAQ outperforms LSQ [15] by 0.2% on the Top-1 accuracy. Remarkably, on CIFAR-100, SAQ quantized 4-bit ResNet-20 even outperforms the full-precision one by 1.2% on the Top-1 accuracy. On ImageNet, our quantized mixed-precision ResNet-18 even achieves 0.7% performance improvement over the

Table 2. Performance comparisons of different methods on ImageNet. We use Bit-Operation (BOP) count to measure the computational cost. "BOP comp. ratio" is the BOP compression ratio. "MP" indicates mixed-precision quantization. "-" denotes that the results are not reported.

Network	Method	Bitwidth	BOPs (G)	BOP comp. ratio	Top-1 Acc. (%)	Top-5 Acc. (%)
ResNet-18	Full-precision	32	1857.6	1.0	70.7	89.8
	PACT [8]	4	34.7	53.5	69.2	89.0
	LQ-Net* [60]	4	34.7	53.5	69.3	88.8
	QIL* [26]	4	34.7	53.5	70.1	-
	LSQ** [15]	4	34.7	53.5	71.1	90.0
	SAQ (Ours)	4	34.7	53.5	**71.3**	**90.0**
	ALQ [45]	MP	58.5	31.8	67.7	-
	DJPQ [58]	MP	35.0	53.1	69.3	-
	DQ [51]	MP	40.7	45.6	68.9	88.6
	HAQ [52]	MP	34.4	54.0	69.2	89.0
	HAWQ [12]	MP	34.0	54.6	68.5	-
	SAMQ (Ours)	MP	**33.7**	**55.1**	**71.4**	**89.9**
	PACT [8]	2	14.4	129.0	64.4	85.6
	LQ-Net* [60]	2	14.4	129.0	64.9	85.9
	QIL* [26]	2	14.4	129.0	65.7	-
	LSQ** [15]	2	14.4	129.0	66.9	-
	SAQ (Ours)	2	14.4	129.0	**67.1**	**87.3**
MobileNetV2	Full-precision	32	306.8	1.0	71.9	90.3
	PACT [8]	4	5.3	57.9	61.4	83.7
	DSQ* [17]	4	15.8	19.4	64.8	-
	LLSQ* [61]	4	15.8	19.4	67.4	88.0
	SAQ (Ours)	4	5.3	57.9	**70.2**	**89.4**
	HAQ [52]	MP	8.3	37.0	69.5	88.8
	DJPQ [58]	MP	7.9	38.8	69.3	-
	SAMQ (Ours)	MP	**5.3**	**57.9**	**70.3**	**89.4**

* denotes that the first and last layers are not quantized.
** represents that models are trained with the weight decay term of 1×10^{-4} for fair comparisons.

full-precision one on the Top-1 accuracy. One possible reason is that performing quantization with SAQ helps to regularize the networks and thus improve the performance of the quantized models. Moreover, compared with the fixed-precision method, mixed-precision quantization further reduces the BOPs while still achieving comparable or even better performance. For instance, on CIFAR-100, SAMQ quantized mixed-precision ResNet-20 with a BOP compression ratio of 111.6 outperforms the 3-bit one by 0.9% in terms of the Top-1 accuracy. On ImageNet, SAMQ quantized ResNet-18 with more BOPs reduction obtains 0.1% improvement over the fixed-precision one on the Top-1 accuracy. These results show the promising performance of our proposed SAQ and SAMQ.

5.2. Further studies

Effectiveness of the sharpness-aware quantization. To investigate the effectiveness of SAQ, we apply different methods to train the fixed-precision models on CIFAR-100, including 1) **SGD**: using the vanilla SGD to train the quantized models; 2) **SAM**: using vanilla sharpness-aware minimization (SAM) [16] that introduces perturbation on the full-precision weights to learn the quantized models; 3) **SAQ**: our proposed method that adds perturbation on the quantized weights to learn the quantized models. To evaluate the loss landscape curvature, we report the largest eigenvalue λ_{max} of the Hessian of different quantized models. The results are shown in Table 3. We also visualize the loss landscapes in the supplementary material. From the results, the quantized models trained by SAM outperform those of SGD by a large margin. For example, SAM quantized 4-bit ResNet-20 surpasses the SGD counterpart by 1.7% on the Top-1 accuracy. We also observe that the λ_{max} values of the quantized ResNet-20 obtained by SAM are much lower than those of SGD, which shows that SAM finds much flatter minima. Critically, our proposed SAQ further improves the performance of the quantized models under all bitwidths and the performance improvement is more obvious when the bitwidth is 4. For example, SAQ quantized 4-bit ResNet-20 achieves 0.4% improvement over SAM on the Top-1 accuracy. Moreover, the λ_{max} values of the quantized ResNet-20 obtained by SAQ are lower than those of

Table 3. Effectiveness of the proposed sharpness-aware quantization. We report the results of different training methods on CIFAR-100. λ_{max} denotes the largest eigenvalue of the Hessian of the quantized model.

Network	Method	Bitwidth	λ_{max}	Top-1 Acc. (%)	Top-5 Acc. (%)
ResNet-20	SGD	2	152.0	63.9±0.1	89.0±0.1
	SAM	2	89.1	64.2±0.1	**89.2±0.1**
	SAQ	2	**86.4**	**64.4±0.2**	89.1±0.1
	SGD	4	108.9	66.6±0.2	90.4±0.1
	SAM	4	57.9	68.3±0.1	**91.5±0.1**
	SAQ	4	**54.5**	**68.7±0.1**	91.2±0.0

Table 4. Effectiveness of the loss sharpness in the reward function. We report the results of different methods on CIFAR-100.

Network	Search Method	Fine-tuning Method	BOPs (M)	BOP comp. ratio	Top-1 Acc. (%)	Top-5 Acc. (%)
ResNet-20	w/o loss sharpness	SGD	380.3	109.9	66.4±0.1	90.1±0.2
	w/ loss sharpness	SGD	**374.4**	**111.6**	**66.5±0.1**	**90.1±0.1**
	w/o loss sharpness	SAQ	380.3	109.9	68.5±0.1	91.1±0.1
	w/ loss sharpness	SAQ	**374.4**	**111.6**	**68.6±0.1**	**91.2±0.1**

SAM, which shows that our SAQ is able to find flatter and smoother minima over SAM. These results demonstrate that flattening the loss landscape improves the generalization performance of the quantized models.

Effectiveness of the loss sharpness in reward function. We further conduct experiments to investigate the effectiveness of the loss sharpness in measuring the quantization sensitivity. Specifically, we apply SAMQ with and without the loss sharpness to obtain the quantized models on CIFAR-100. Here, SAMQ without the loss sharpness denotes that we replace the perturbed loss terms $\mathcal{L}_V(Q_w(\mathbf{w}^*(\mathbf{b}), \mathbf{b}) + \tilde{\boldsymbol{\epsilon}})$ in Eq. (13) with the empirical loss of the unperturbed quantized weight $\mathcal{L}_V(Q_w(\mathbf{w}^*(\mathbf{b}), \mathbf{b}))$. We then fine-tune the resulting quantized models using SGD and our proposed SAQ. The results are shown in Table 4. From the results, for the bitwidth configurations searched with and without the loss sharpness, the models fine-tuned by our SAQ achieve much better performance than those of SGD. For example, for the bitwidth configuration obtained with the loss sharpness, SAQ fine-tuned ResNet-20 achieves 2.1% performance improvement over the SGD counterpart in terms of the Top-1 accuracy. These results further demonstrate that our proposed SAQ is able to improve the generalization ability of the quantized models. Moreover, the bitwidth configuration searched with loss sharpness has better generalization performance. For example, using SGD to fine-tune the resulting models, the quantized model searched with the loss sharpness surpasses the one not equipped with the loss sharpness by 0.1% in terms of the Top-1 accuracy.

6. Conclusion and future work

In this paper, we have devised a new training method, called Sharpness-Aware Quantization (SAQ) to improve the generalization capability of the quantized models. To this end, we have proposed to find a perturbation on the quantized weights instead of the full-precision ones that maximizes the perturbed loss. By minimizing the perturbed loss, our proposed SAQ enforces the quantized models to have low loss value and loss sharpness, delivering improved generalization performance. We have further proposed SAMQ to automatically determine the bitwidth of each layer by encouraging lower bits for flat minima and vice versa for sharp regions. To achieve this, we have proposed to learn a configuration generator with reinforcement learning and have designed a reward function by simultaneously considering the loss value, the loss sharpness, and the computational constraint. By training the generator and the network parameters in an alternating manner, our proposed SAMQ promotes the flatness of the loss landscape, thus permitting more aggressive quantization. Experiments on CIFAR-100 and ImageNet have demonstrated that our proposed methods consistently improves the performance of the quantized models.

Our proposed method pushes the boundaries for developing efficient deep learning applications at scale, especially on embedded devices with limited computation resources. However, as shown in Eq. (10), our proposed method has to solve the min-max optimization problem. Therefore, the computational cost for training is roughly doubled compared with those of conventional optimizers, such as SGD, which leads to more energy consumption and carbon emission during training.

In the future, we may extend our methods in several aspects. First, we may consider low-precision training to improve the training efficiency of our proposed method. Second, we may extend our method to perform pruning and quantization simultaneously and obtain more compact models with better performance.

References

[1] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. 3

[2] Joseph Bethge, Christian Bartz, Haojin Yang, Ying Chen, and Christoph Meinel. Meliusnet: Can binary neural networks achieve mobilenet-level accuracy? *arXiv preprint arXiv:2001.05936*, 2020. 2

[3] Han Cai, Ligeng Zhu, and Song Han. ProxylessNAS: Direct neural architecture search on target task and hardware. In *ICLR*, 2019. 11

[4] Zhaowei Cai, Xiaodong He, Jian Sun, and Nuno Vasconcelos. Deep learning with low precision by half-wave gaussian quantization. In *CVPR*, pages 5918–5926, 2017. 2

[5] Zhaowei Cai and Nuno Vasconcelos. Rethinking differentiable search for mixed-precision neural networks. In *CVPR*, pages 2349–2358, 2020. 2

[6] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer T. Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. In *ICLR*, 2017. 2

[7] Weihan Chen, Peisong Wang, and Jian Cheng. Towards mixed-precision quantization of neural networks via constrained optimization. In *ICCV*, pages 5350–5359, 2021. 2

[8] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018. 2, 6, 7, 11

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009. 5

[10] Ruizhou Ding, Ting-Wu Chin, Zeye Liu, and Diana Marculescu. Regularizing activation distribution for training binarized deep networks. In *CVPR*, pages 11408–11417, 2019. 2

[11] Zhen Dong, Zhewei Yao, Daiyaan Arfeen, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. HAWQ-V2: hessian aware trace-weighted quantization of neural networks. In *NeurIPS*, 2020. 2, 4

[12] Zhen Dong, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Hawq: Hessian aware quantization of neural networks with mixed-precision. In *CVPR*, pages 293–302, 2019. 7, 11

[13] Jiawei Du, Hanshu Yan, Jiashi Feng, Joey Tianyi Zhou, Liangli Zhen, Rick Siow Mong Goh, and Vincent YF Tan. Efficient sharpness-aware minimization for improved training of neural networks. *arXiv preprint arXiv:2110.03141*, 2021. 2, 3, 11

[14] Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. In Gal Elidan, Kristian Kersting, and Alexander T. Ihler, editors, *UAI*, 2017. 1, 2

[15] Steven K. Esser, Jeffrey L. McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S. Modha. Learned step size quantization. In *ICLR*, 2020. 2, 6, 7

[16] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *ICLR*, 2021. 1, 2, 3, 4, 7

[17] Ruihao Gong, Xianglong Liu, Shenghu Jiang, Tianxiang Li, Peng Hu, Jiazhen Lin, Fengwei Yu, and Junjie Yan. Differentiable soft quantization: Bridging full-precision and low-bit neural networks. In *ICCV*, pages 4852–4861, 2019. 2, 7

[18] Yong Guo, Yaofo Chen, Yin Zheng, Qi Chen, Peilin Zhao, Jian Chen, Junzhou Huang, and Mingkui Tan. Pareto-frontier-aware neural architecture generation for diverse budgets. *arXiv preprint arXiv:2103.00219*, 2021. 4

[19] Yong Guo, Yaofo Chen, Yin Zheng, Peilin Zhao, Jian Chen, Junzhou Huang, and Mingkui Tan. Breaking the curse of space explosion: Towards efficient nas with curriculum search. In *ICML*, pages 3822–3831, 2020. 11

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 5

[21] Sepp Hochreiter and Jürgen Schmidhuber. Simplifying neural nets by discovering flat minima. In *NeurIPS*, pages 529–536, 1995. 1, 2

[22] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural computation*, 9(1):1–42, 1997. 2, 4

[23] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. *NeurIPS*, 29, 2016. 1, 2, 3

[24] Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *ICLR*, 2020. 1, 2

[25] Qing Jin, Linjie Yang, and Zhenyu Liao. Adabits: Neural network quantization with adaptive bit-widths. In *CVPR*, June 2020. 5

[26] Sangil Jung, Changyong Son, Seohyung Lee, Jinwoo Son, Jae-Joon Han, Youngjun Kwak, Sung Ju Hwang, and Changkyu Choi. Learning to quantize deep networks by optimizing quantization intervals with task loss. In *CVPR*, pages 4350–4359, 2019. 2, 7

[27] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *ICLR*, 2017. 1, 2

[28] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 2, 6

[29] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Tech Report*, 2009. 5

[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *NeurIPS*, 25:1097–1105, 2012. 1

[31] Jungmin Kwon, Jeongseop Kim, Hyun-Seok Park, and In Kwon Choi. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. In *ICML*, 2021. 2, 3, 11, 13

[32] Hao Li, Zheng Xu, Gavin Taylor, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *NeurIPS*, 2018. 1, 12

[33] Yuhang Li, Xin Dong, and Wei Wang. Additive powers-of-two quantization: An efficient non-uniform discretization for neural networks. In *ICLR*, 2020. 11

[34] Xiaofan Lin, Cong Zhao, and Wei Pan. Towards accurate binary convolutional neural network. In *NeurIPS*, pages 345–353, 2017. 2

[35] Chen Liu, Mathieu Salzmann, Tao Lin, Ryota Tomioka, and Sabine Süsstrunk. On the loss landscape of adversarial training: Identifying challenges and how to overcome them. In *NeurIPS*, 2020. 2

[36] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: differentiable architecture search. In *ICLR*, 2019. 11

[37] Zechun Liu, Zhiqiang Shen, Shichao Li, Koen Helwegen, Dong Huang, and Kwang-Ting Cheng. How do adam and training strategies help bnns optimization. In *ICML*, volume 139, pages 6936–6946, 2021. 1, 2

[38] Zechun Liu, Zhiqiang Shen, Marios Savvides, and Kwang-Ting Cheng. Reactnet: Towards precise binary neural network with generalized activation functions. In *ECCV*, pages 143–159, 2020. 2

[39] Zechun Liu, Baoyuan Wu, Wenhan Luo, Xin Yang, Wei Liu, and Kwang-Ting Cheng. Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In *ECCV*, pages 722–737, 2018. 2

[40] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *ICLR*, 2017. 6

[41] Brais Martínez, Jing Yang, Adrian Bulat, and Georgios Tzimiropoulos. Training binary neural networks with real-to-binary convolutions. In *ICLR*, 2020. 2

[42] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Jonathan Uesato, and Pascal Frossard. Robustness via curvature regularization, and vice versa. In *CVPR*, pages 9078–9086, 2019. 2

[43] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *ICCV*, pages 1520–1528, 2015. 1

[44] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In *ICML*, pages 4095–4104, 2018. 4, 5

[45] Zhongnan Qu, Zimu Zhou, Yun Cheng, and Lothar Thiele. Adaptive loss-aware quantization for multi-bit networks. In *CVPR*, pages 7988–7997, 2020. 7

[46] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *ECCV*, pages 525–542, 2016. 2

[47] Jorma Rissanen. Modeling by shortest data description. *Autom.*, 14:465–471, 1978. 2, 4

[48] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951. 2

[49] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, pages 4510–4520, 2018. 6

[50] Samuel L. Smith and Quoc V. Le. A bayesian perspective on generalization and stochastic gradient descent. In *ICLR*, 2018. 2

[51] Stefan Uhlich, Lukas Mauch, Fabien Cardinaux, Kazuki Yoshiyama, Javier Alonso Garcia, Stephen Tiedemann, Thomas Kemp, and Akira Nakamura. Mixed precision dnns: All you need is a good parametrization. In *ICLR*, 2020. 2, 6, 7, 11

[52] Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. Haq: Hardware-aware automated quantization with mixed precision. In *CVPR*, pages 8612–8620, 2019. 2, 6, 7, 11

[53] Tianzhe Wang, Kuan Wang, Han Cai, Ji Lin, Zhijian Liu, Hanrui Wang, Yujun Lin, and Song Han. Apq: Joint search for network architecture, pruning and quantization policy. In *CVPR*, pages 2078–2087, 2020. 2

[54] Bichen Wu, Yanghan Wang, Peizhao Zhang, Yuandong Tian, Peter Vajda, and Kurt Keutzer. Mixed precision quantization of convnets via differentiable neural architecture search. *arXiv preprint arXiv:1812.00090*, 2018. 2

[55] Kohei Yamamoto. Learnable companding quantization for accurate low-bit neural networks. In *CVPR*, pages 5029–5038, 2021. 2

[56] Jiwei Yang, Xu Shen, Jun Xing, Xinmei Tian, Houqiang Li, Bing Deng, Jianqiang Huang, and Xian-sheng Hua. Quantization networks. In *CVPR*, 2019. 2

[57] Zhewei Yao, Zhen Dong, Zhangcheng Zheng, Amir Gholami, Jiali Yu, Eric Tan, Leyuan Wang, Qijing Huang, Yida Wang, Michael Mahoney, et al. Hawq-v3: Dyadic neural network quantization. In *ICML*, pages 11875–11886, 2021. 2, 4, 5

[58] Wang Ying, Lu Yadong, and Blankevoort Tijmen. Differentiable joint pruning and quantization for hardware efficiency. In *ECCV*, pages 259–277, 2020. 5, 7

[59] Jiahui Yu, Linjie Yang, Ning Xu, Jianchao Yang, and Thomas Huang. Slimmable neural networks. In *ICLR*, 2019. 5

[60] Dongqing Zhang, Jiaolong Yang, Dongqiangzi Ye, and Gang Hua. Lq-nets: Learned quantization for highly accurate and compact deep neural networks. In *ECCV*, pages 365–382, 2018. 2, 7

[61] Xiandong Zhao, Ying Wang, Xuyi Cai, Cheng Liu, and Lei Zhang. Linear symmetric quantization of neural networks for low-precision integer hardware. In *ICLR*, 2020. 7

[62] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016. 1, 2, 3

[63] Bohan Zhuang, Chunhua Shen, Mingkui Tan, Lingqiao Liu, and Ian Reid. Towards effective low-bitwidth convolutional neural networks. In *CVPR*, pages 7920–7928, 2018. 2

[64] Bohan Zhuang, Chunhua Shen, Mingkui Tan, Lingqiao Liu, and Ian Reid. Structured binary neural networks for accurate image classification and semantic segmentation. In *CVPR*, pages 413–422, 2019. 2

[65] Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. In *ICLR*, 2017. 5

Appendix

We organize our appendix as follows.

- In Section A, we provide more implementation details about our SAQ and SAMQ.
- In Section B, we describe the details about the quantization configurations of different methods.
- In Section C, we include more results of SAQ on CIFAR-100 and ImageNet.
- In Section D, we show the searched bitwidth configurations of the quantized models on ImageNet.
- In Section E, we visualize the loss landscapes of different quantized models on CIFAR-100.
- In Section F, we extend our proposed SAQ to Adaptive Sharpness-Aware Quantization (ASAQ) by introducing adaptive sharpness [31] and show the results on CIFAR-100.

A. More implementation details

In this section, we provide more implementation details of our SAQ and SAMQ.

For the fixed-precision training, we apply m-sharpness strategy with $m = 128$ following ESAM [13] and ASAM [31]. Following [33], we introduce weight normalization during training. The clipping levels for weights and activations are initialized to 1. For the hyper-parameter ρ, we conduct grid search over $\{0.02, 0.05, 0.1, 0.15, 0.2, \ldots, 0.9, 1.0\}$ to find appropriate values of ρ. The detailed settings of ρ on CIFAR-100 and ImageNet are put in Tables A and B, respectively. To compute the largest eigenvalue λ_{max} of the Hessian of different quantized models on CIFAR-100, we use power iteration algorithm following [12]. To reduce the computational cost, we randomly sample 500 training images to compute λ_{max}.

For the mixed-precision quantization, we first train a configuration generator to obtain bitwidth configurations that satisfy the computational constraint. The configuration generator is an LSTM network with a fully-connected layer to predict the bitwidth of each layer. The dimension of the hidden state is set to 64. Following DARTS [36], we use half of the CIFAR-100 training data as the validation set. To accelerate the bitwidth search on ImageNet, we randomly sample 100 classes of samples for training. Following ProxylessNAS [3], we randomly sample 50k training images as the validation set. We set β to 1×10^{-4} for the experiments on CIFAR-100 and 5×10^{-3} for those on ImageNet. For the training of the low-precision models in SAMQ, we apply a warmup strategy following [19]. Specifically, we fix the configuration generator and only train the quantized models for 10 epochs to provide better initialization.

Table A. Hyper-parameter ρ for different quantized models on CIFAR-100.

Network	ResNet-20			MobileNetV2		
Bitwidth	2	3	4	2	3	4
ρ	0.4	0.7	0.9	0.5	0.7	0.9

Table B. Hyper-parameter ρ for different quantized models on ImageNet.

Network	ResNet-18			MobileNetV2
Bitwidth	2	3	4	4
ρ	0.15	0.15	0.3	0.3

B. More details about quantization configurations

In this section, we provide more details about the quantization configurations of different methods mentioned in Table 1. Specifically, for PACT [8], we initialize the clipping level of activations to 1. For DQ [51], we use the parameterization case U3 with $\boldsymbol{\theta} = [d, q_{\max}]$. We initialize the low-precision weights using the full-precision model. We set the initial step size to $d = 2^{\lfloor \log_2(\max(|\mathbf{W}|)/(2^{b-1}-1)) \rfloor}$ for weights and 2^{-3} for activations. The other quantization parameters are set such that the initial bitwidth is 4-bit. For HAQ [52], we first clamp the weights and activations into the range of $[-\alpha_w, \alpha_w]$ and $[0, \alpha_z]$, respectively. We then perform linear quantization on weights and activations. To choose appropriate values of α_w and α_z, we minimize the KL-divergence between the original weights $\mathbf{w}$ and the quantized weights $Q(\mathbf{w})$.

C. More results on CIFAR-100 and ImageNet

In this section, we provide more results of our proposed SAQ on CIFAR-100 and ImageNet in Tables C and D, respectively. From the results, the quantized models obtained by our SAQ consistently outperform the SGD counterparts at all

Table C. Performance comparisons of different methods on CIFAR-100.

Network	Method	Bitwidth	Top-1 Acc. (%)	Top-5 Acc. (%)
ResNet-20	Full-precision	32	67.5	90.8
	SGD	2	63.9±0.1	89.0±0.1
	SAQ	2	**64.4±0.2**	**89.1±0.1**
	SGD	3	66.4±0.1	90.2±0.1
	SAQ	3	**67.7±0.2**	**90.8±0.1**
	SGD	4	66.6±0.2	90.4±0.1
	SAQ	4	**68.7±0.1**	**91.2±0.0**
MobileNetV2	Full-precision	32	75.1	94.0
	SGD	2	70.1±0.2	91.7±0.1
	SAQ	2	**70.6±0.1**	**92.1±0.1**
	SGD	3	73.6±0.1	92.9±0.1
	SAQ	3	**74.4±0.2**	**93.2±0.1**
	SGD	4	74.2±0.1	93.1±0.1
	SAQ	4	**75.6±0.2**	**93.7±0.1**

Table D. Performance comparisons of different methods on ImageNet.

Network	Method	Bitwidth	Top-1 Acc. (%)	Top-5 Acc. (%)
ResNet-18	Full-precision	32	70.7	89.8
	SGD	2	66.8	87.3
	SAQ	2	**67.1**	**67.3**
	SGD	3	70.1	89.2
	SAQ	3	**70.2**	**89.3**
	SGD	4	71.1	89.8
	SAQ	4	**71.3**	**90.0**

bitwidths. For example, for 3-bit ResNet-20 on CIFAR-100, our SAQ surpasses SGD by 1.3% on the Top-1 accuracy. For 2-bit ResNet-18 on ImageNet, our SAQ achieves 0.3% performance improvement over the SGD in terms of the Top-1 accuracy. Surprisingly, some low-precision models fine-tuned by our SAQ even outperform the full-precision one. To be specific, for 3-bit ResNet-20 on CIFAR-100, our SAQ achieves 0.2% improvement over the full-precision one on the Top-1 accuracy. These results show that our proposed SAQ is able to significantly improve the performance of the quantized models.

D. More details about the bitwidth configurations of the quantized models

In this section, we illustrate the detailed bitwidth configurations of each layer from different quantized models on ImageNet. We show the results in Figures A and B. For the first and last layers of all quantized models, we quantize both weights and activations to 8-bit. For ResNet-18, our SAMQ assigns more bitwidths to the shallow layers and fewer bitwidths to the deeper layers. For MobileNetV2, our SAMQ tends to allocate more bitwidths to the pointwise convolutional layers. Intuitively, one possible reason is that these layers contribute more to the discriminative power of the quantized models. Compressing these layers lead to severe information loss, resulting in a significant performance drop.

E. Visualization of the loss landscapes

In this section, we show the loss landscape of different quantized models on CIFAR-100 using the visualization method in [32]. We show the results in Figures C, D and E. The x- and y-axes of the figures represent two randomly sampled orthogonal directions. From the results, the loss landscapes of the quantized models become smoother and flatter with the increase of bitwidth, which shows that training low-precision models is more difficult than training high-precision ones. Moreover, the loss landscapes of the quantized models obtained by our SAQ are less chaotic and show larger contour interval compared with the SGD counterpart, indicating that SAQ is able to find flatter and smoother minima over SGD.

Table E. Performance comparisons between ASAQ and SAQ on CIFAR-100.

Network	Method	Bitwidth	Top-1 Acc. (%)	Top-5 Acc. (%)
ResNet-20	Full-precision	32	67.5	90.8
	SAQ	2	**64.4±0.2**	89.1±0.1
	ASAQ	2	**64.4±0.1**	**89.2±0.1**
	SAQ	3	67.7±0.2	90.8±0.1
	ASAQ	3	**67.9±0.1**	**91.0±0.1**
	SAQ	4	68.7±0.1	91.2±0.0
	ASAQ	4	**69.0±0.1**	**91.5±0.0**
MobileNetV2	Full-precision	32	75.1	94.0
	SAQ	2	70.6±0.1	92.1±0.1
	ASAQ	2	**71.0±0.2**	**92.2±0.0**
	SAQ	3	74.4±0.2	93.2±0.1
	ASAQ	3	**74.8±0.1**	**93.5±0.1**
	SAQ	4	75.6±0.2	93.7±0.1
	ASAQ	4	**75.9±0.1**	**94.0±0.1**

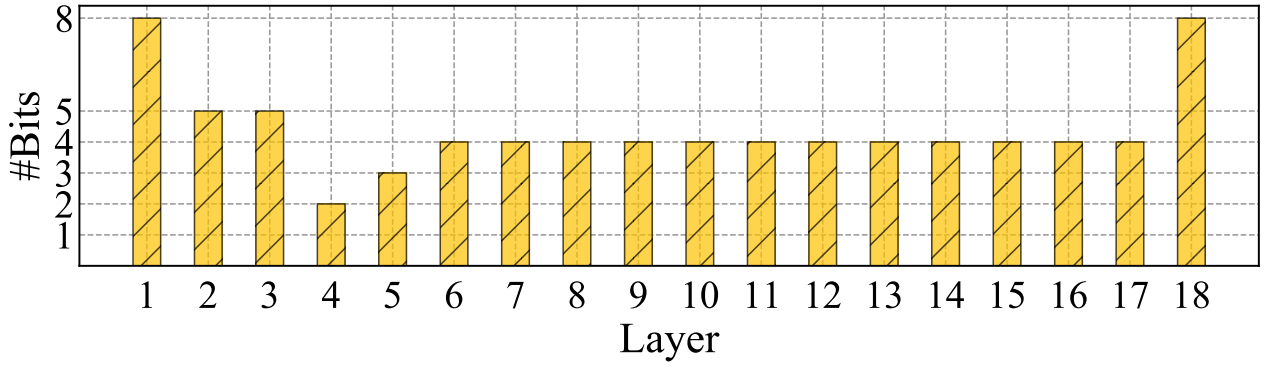


Figure A. Detailed configurations of the quantized ResNet-18 on ImageNet. The Top-1 accuracy, Top-5 accuracy and BOPs of the quantized ResNet-18 are 71.4%, 89.9% and 33.7G, respectively.

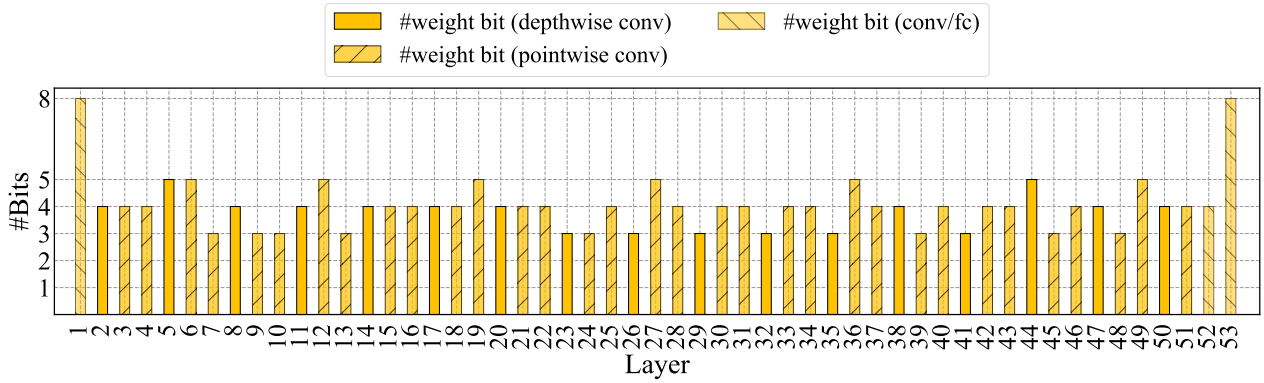
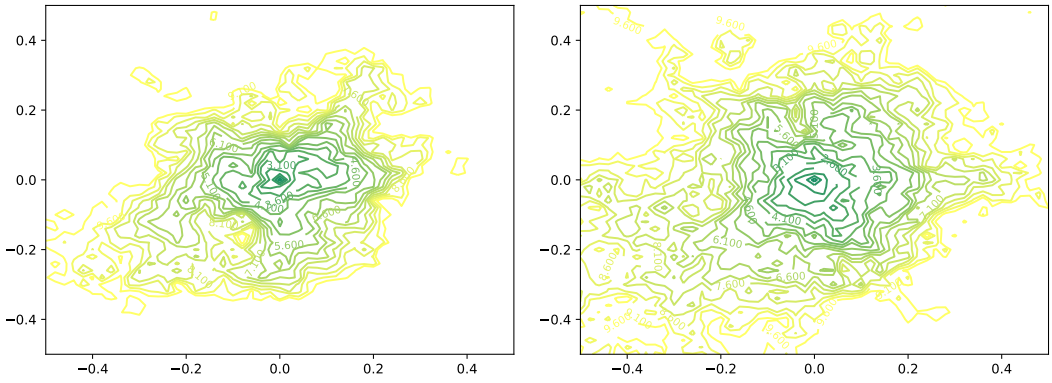


Figure B. Detailed configurations of the quantized MobileNetV2 on ImageNet. The Top-1 accuracy, Top-5 accuracy and BOPs of the quantized MobileNetV2 are 70.3%, 89.4% and 5.3G, respectively.

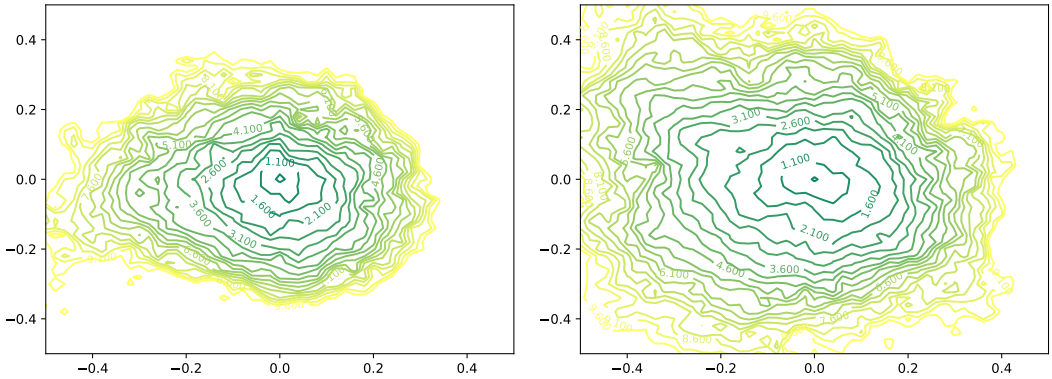
F. Extension to adaptive sharpness-aware quantization

Recently, adaptive sharpness-aware minimization (ASAM) [31] has been proposed to improve the performance of SAM by mitigating the effect of parameter re-scaling. Motivated by ASAM, we can extend our proposed SAQ to Adaptive Sharpness-Aware Quantization (ASAQ) to further improve the performance of the quantized models by introducing a normalization



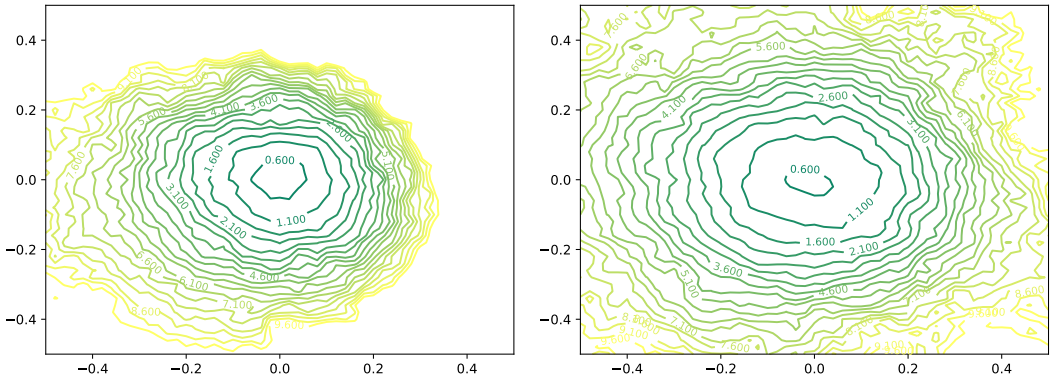
(a) SGD (b) SAQ

Figure C. The loss landscapes of the 2-bit ResNet-20 fine-tuned by different methods on CIFAR-100.



(a) SGD (b) SAQ

Figure D. The loss landscapes of the 3-bit ResNet-20 fine-tuned by different methods on CIFAR-100.



(a) SGD (b) SAQ

Figure E. The loss landscapes of the 4-bit ResNet-20 fine-tuned by different methods on CIFAR-100.

operation. To evaluate the performance, we apply ASAQ to quantize ResNet-20 and MobileNetV2 on CIFAR-100. We report the results in Table E. From the results, the quantized models fine-tuned by ASAQ outperform the SAQ counterparts for most cases. For example, for 2-bit MobileNetV2, the models obtained by ASAQ surpass those of SAQ by 0.4% in terms of the average Top-1 accuracy. These results show the effectiveness of the proposed ASAQ.