Double Rounding: Nearly Lossless Adaptive Bit Switching for QAT

Haiduo Huang[1], **Zhenhua Liu**[2], **Tian Xia**[1], **Pengju Ren**[1*]

[1]Xi'an Jiaotong University
[2]Peking University
huanghd@stu.xjtu.edu.cn, liu.zhenhua@pku.edu.cn, tian_xia@xjtu.edu.cn, pengjuren@xjtu.edu.cn

Abstract

Model quantization is widely applied for compressing and accelerating deep neural networks (DNNs). However, conventional Quantization-Aware Training (QAT) focuses on training DNNs with uniform bit-width. The bit-width settings vary across different hardware and transmission demands, which induces considerable training and storage costs. Hence, the scheme of one-shot joint training multiple precisions is proposed to address this issue. Previous works either store a larger FP32 model to switch between different precision models for higher accuracy or store a smaller INT8 model but compromise accuracy due to using shared quantization parameters. In this paper, we introduce the ***Double Rounding*** quantization method, which fully utilizes the quantized representation range to accomplish nearly lossless bit-switching while reducing storage by using the highest integer precision instead of full precision. Furthermore, we observe a competitive interference among different precisions during one-shot joint training, primarily due to inconsistent gradients of quantization scales during backward propagation. To tackle this problem, we propose an Adaptive Learning Rate Scaling (**AdaScale**) technique that dynamically adapts learning rates for various precisions to optimize the training process. Additionally, we extend our *Double Rounding* to one-shot mixed precision training and develop a Hessian-Aware Stochastic Bit-switching (**HessBit**) strategy. Experimental results on the ImageNet-1K classification demonstrate that our methods have enough advantages to state-of-the-art one-shot joint QAT in both multi-precision and mixed-precision. We validate the feasibility of our method on detection and segmentation tasks, as well as on LLMs task.

Code — https://github.com/haiduo/Double-Rounding

Introduction

Model compression (Polino, Pascanu, and Alistarh 2018) has become a focal point for researchers due to constraints in computational resources and storage. Model quantization has emerged as a prominent solution, mapping floating-point values to integer representations to substantially reduce storage and computation costs without modifying the network architecture. Typically, a pre-trained model is quantized with a fixed bit-width tailored to a specific application scenario,

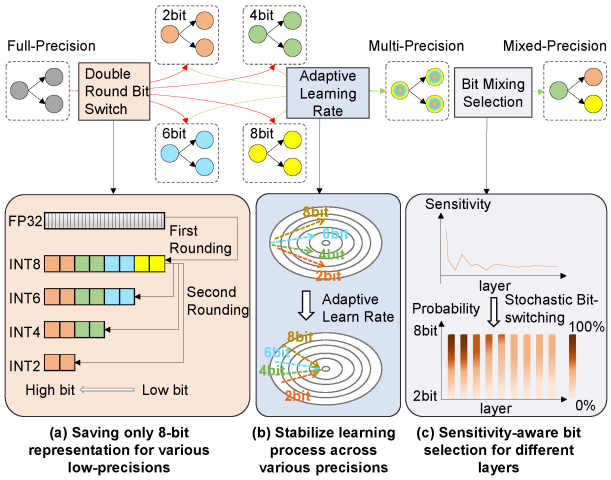


Figure 1: Illustration of our proposed lossless adaptive bit-switching framework.

followed by Quantization-Aware Training (QAT) to recover potential accuracy loss. However, deploying the same model across diverse scenarios with varying precision requirements often necessitates repeated retraining, resulting in significant computational and training overhead. To overcome this inefficiency, recent works have explored simultaneous training of multi-precision (Jin Qing 2020; Xu et al. 2022) or one-shot mixed-precision (Xu et al. 2023) models, where shared weight parameters across different precisions enable dynamic bit-width switching during inference.

However, bit-switching from high to low precision (or bit-width) often leads to notable accuracy degradation, primarily due to the inherent *Rounding* operation in quantization. Moreover, in multi-precision schemes, there exists intense competition during convergence between higher and lower precisions, which further complicates optimization. For mixed-precision schemes, prior approaches typically separate the training and search phases, resulting in substantial search and retraining overhead. These challenges collectively make bit-switching a particularly difficult problem. Motivated by this, we aim to develop a bit-switching quantization method that eliminates the need to store a full-precision model while enabling nearly lossless transitions from high to low bit-widths. To this

*Corresponding author

end, we introduce unified representation, normalized learning steps, and an adaptive probability distribution across different precisions, facilitating an efficient and stable learning process for both multi-precision and mixed-precision scenarios, as illustrated in Figure 1.

To address the bit-switching challenge, previous methods either store floating-point parameters (Yu et al. 2021; Kunyuan Du 2020; Xu et al. 2022; Sun et al. 2024) to prevent accuracy loss, or replace the *rounding* operation with *floor* (Jin Qing 2020; Bulat et al. 2021), which, however, often leads to accuracy degradation or even training collapse at lower bit-widths. In this work, we propose *Double Rounding*, which applies the *rounding* operation twice, as illustrated in Figure 1 (a). This design enables nearly lossless bit-switching and allows the model to be stored at the highest bit-width, rather than in full precision. In essence, the lower-precision weights are embedded within the higher-precision representation, thereby reducing storage overhead.

In addition, we observe that joint training of multiple precisions, especially involving 2-bit quantization, suffers from severe competition between high and low precisions, consistent with findings in (Tang et al. 2022; Xu et al. 2022). This phenomenon arises from two main factors: the optimal quantization intervals differ across precisions, and the gradients of quantization intervals for different precisions have varying magnitudes during training, despite sharing weights. To address this, we introduce Adaptive Learning Rate Scaling (AdaScale), which dynamically adjusts the learning rates for different precisions to ensure consistent update steps for quantization scales, as depicted in Figure 1 (b).

Furthermore, we extend our approach to efficient one-shot mixed-precision quantization based on *Double Rounding*. Unlike previous mixed-precision methods that require training a SuperNet, searching for optimal SubNets under constraints, and then retraining or fine-tuning—resulting in significant time and computational costs—we leverage the Hessian Matrix Trace (Dong et al. 2020) as a sensitivity metric to guide SuperNet optimization. We propose a Hessian-Aware Stochastic Bit-switching (HessBit) strategy, inspired by the Roulette algorithm (Dong et al. 2019a), which adaptively assigns bit-widths to each layer according to their sensitivity, as shown in Figure 1 (c). Sensitivity is also incorporated as a constraint during the search stage, allowing us to eliminate the need for additional retraining.

Related Works

Multi-precision quantization leverages one-shot joint Quantization-Aware Training (QAT) to enable a single model to support multiple bit-widths, dynamically adapting to hardware and storage constraints. Representative works include AdaBits (Jin, Yang, and Liao 2019), which first explored adaptive bit-switching but faced convergence issues at low bit-widths, TQ (Zhang et al. 2021) and BitWave (Shi et al. 2024), which exploit structured sparsity and dynamic dataflow, and Bit-Mixer (Bulat et al. 2021), which uses LSQ (Esser et al. 2019) but at the cost of discarding the lowest quantized state. Some methods, such as Any-precision (Yu et al. 2021) and MultiQuant (Xu et al. 2022), incorporate knowledge distillation to improve accuracy, with MultiQuant's "Online Adaptive Label" essentially being a form of self-distillation (Kim et al. 2021). While AdaBits and Bit-Mixer can store an 8-bit model for bit-switching, others require a 32-bit model; our *Double Rounding* method enables nearly lossless bit-switching by storing only the highest bit-width model, reducing training time by about 10% (Kunyuan Du 2020) compared to separate training.

One-shot mixed-precision prior works often rely on expensive reinforcement learning (Wang et al. 2019; Elthakeb et al. 2019), Neural Architecture Search (NAS) (Wu et al. 2018; Guo et al. 2020a; Shen et al. 2021), or partial prior knowledge (Liu, Cai, and Zhuang 2021; Yao et al. 2021) for bit-width allocation, which may not yield globally optimal solutions. In contrast, our Hessian-Aware optimization refines a SuperNet via gradient updates and enables efficient conditional SubNet search without retraining or fine-tuning. While Bit-Mixer (Bulat et al. 2021) and MultiQuant (Xu et al. 2022) also support layer-adaptive mixed-precision, the former uses a naive search and the latter requires extensive fine-tuning. Unlike NAS-based methods that alter network architecture, our approach optimizes a once-for-all SuperNet solely through quantization.

Methodology

Double Rounding

Conventional Quantization-Aware Training (QAT) (Jacob et al. 2017) typically produces a quantized model at a fixed bit-width, starting from a pre-trained FP32 model. During training, a pseudo-quantization node is inserted into each layer, which consists of two main operations: the quantization function $quant(x)$, mapping FP32 values to lower-bit integers, and the dequantization function $dequant(x)$, which reconstructs the floating-point value from the quantized integer. This process effectively simulates the quantization error introduced by compressing floating-point values into integers. Since quantization involves a non-differentiable $Rounding$ operation, the Straight-Through Estimator (STE) (Bengio, Léonard, and Courville 2013) is widely adopted to enable gradient-based optimization.

In the context of multi-precision quantization, however, bit-switching—especially from higher to lower bit-widths (e.g., 8-bit to 2-bit)—often incurs substantial accuracy degradation. Previous approaches have mainly adopted two strategies to address this: (1) performing bit-switching from a full-precision (32-bit) model to lower-bit models using multiple learnable quantization parameters, or (2) replacing the $Rounding$ operation with $Floor$, which unfortunately leads to further accuracy loss, particularly at very low bit-widths. To overcome these limitations, we propose *Double Rounding*, a nearly lossless bit-switching quantization method. By applying the $Rounding$ operation twice, our approach enables the model to be stored in the highest-bit (e.g., 8-bit) integer format, rather than full-precision, and supports seamless switching to other bit-widths. Figure 2 provides a detailed comparison between *Double Rounding* and other quantization schemes.

Distinct from AdaBits, which is based on the Dorefa (Zhou et al. 2016) quantization method with bit-width-dependent

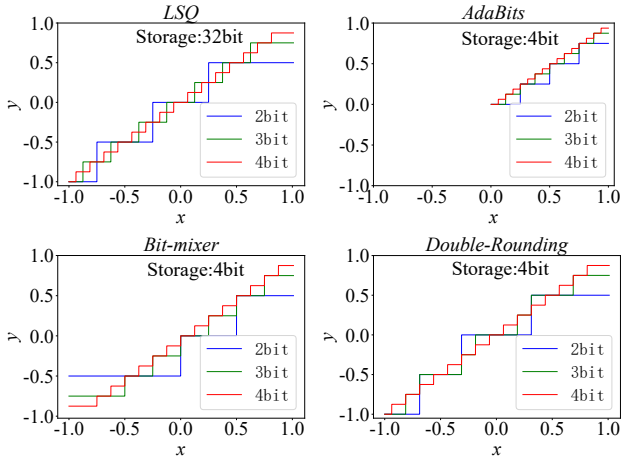


Figure 2: Comparison of four quantization schemes:(a) *LSQ* (Esser et al. 2019), (b) *AdaBits* (Jin Qing 2020), (c) *Bit-Mixer* (Bulat et al. 2021) and (d) Ours *Double Rounding*. In all cases $y = dequant(quant(x))$.

quantization scales, our *Double Rounding* learns the quantization scale online and does not require it to be fixed. Only a pair of shared quantization parameters—*scale* and *zero-point*—are needed. The quantization scales for different precisions are strictly related by a "Power of Two" relationship. Let h and l denote the highest and target lower bit-widths, respectively, and define $\Delta = h - l$. The *Double Rounding* process is formulated as:

$$\widetilde{W}_h = \text{clip}\left(\left\lfloor \frac{W - \mathbf{z}_h}{\mathbf{s}_h} \right\rceil, -2^{h-1}, 2^{h-1} - 1\right) \tag{1}$$

$$\widetilde{W}_l = \text{clip}\left(\left\lfloor \frac{\widetilde{W}_h}{2^{\Delta}} \right\rceil, -2^{l-1}, 2^{l-1} - 1\right) \tag{2}$$

$$\widehat{W}_l = \widetilde{W}_l \times \mathbf{s}_h \times 2^{\Delta} + \mathbf{z}_h \tag{3}$$

were, $\lfloor . \rceil$ denotes the *Rounding* function, and $\text{clip}(x, low, upper)$ restricts x to the interval $[low, upper]$. W is the FP32 weight, $\mathbf{s}_h \in \mathbb{R}$ and $\mathbf{z}_h \in \mathbb{Z}$ are the quantization *scale* and *zero-point* for the highest bit-width, respectively. $\widetilde{W}_h$ is the quantized weight at the highest bit-width, while $\widetilde{W}_l$ and $\widehat{W}_l$ are the quantized and dequantized weights at the lower bit-width.

The division of $\widetilde{W}_h$ by 2^{Δ} can be efficiently realized through hardware shift operations, which greatly accelerates inference. Moreover, our *Double Rounding* framework supports both shared and unshared quantization parameters for bit-switching, offering additional flexibility and the potential for further accuracy improvements when unshared parameters are used. In our symmetric quantization setting, we set $\mathbf{z}_h = 0$ for simplicity.

Unlike weights, which are fixed after training, activations are dynamic during inference. Therefore, the *scale* and *zero-point* for activations at different precisions can be learned independently to further enhance accuracy. Let X denote the full-precision activation, and $\widetilde{X}_b$ and $\widehat{X}_b$ be the quantized and dequantized activations at b-bit, respectively. The quantization process is given by:

$$\widetilde{X}_b = \text{clip}\left(\left\lfloor \frac{X - \mathbf{z}_b}{\mathbf{s}_b} \right\rceil, 0, 2^b - 1\right) \tag{4}$$

$$\widehat{X}_b = \widetilde{X}_b \times \mathbf{s}_b + \mathbf{z}_b \tag{5}$$

where $\mathbf{s}_b \in \mathbb{R}$ and $\mathbf{z}_b \in \mathbb{Z}$ are the quantization *scale* and *zero-point* for the b-bit activation, respectively. For ReLU activations, $\mathbf{z}_b$ is set to 0.

For one-shot joint training, the gradients of *Double Rounding* with respect to the quantization scale and zero-point are formulated as:

$$\frac{\partial \widehat{Y}}{\partial \mathbf{s}_h} \simeq \begin{cases} \left\lfloor \frac{Y - \mathbf{z}_h}{\mathbf{s}_h} \right\rceil - \frac{Y - \mathbf{z}_h}{\mathbf{s}_h} & \text{if } n < \frac{Y - \mathbf{z}_h}{\mathbf{s}_h} < p, \\ n \quad \text{or} \quad p & \text{otherwise} \end{cases} \tag{6}$$

$$\frac{\partial \widehat{Y}}{\partial \mathbf{z}_h} \simeq \begin{cases} 0 & \text{if } n < \frac{Y - \mathbf{z}_h}{\mathbf{s}_h} < p, \\ 1 & \text{otherwise} \end{cases} \tag{7}$$

where n and p are the lower and upper bounds of the integer quantization range $[N_{min}, N_{max}]$, Y denotes the FP32 weights or activations, and $\widehat{Y}$ is the corresponding dequantized value. For activations, the quantization *scale* and *zero-point* for each precision are learned independently, but the gradient formulation remains consistent with that of the weights.

Adaptive Learning Rate Scaling for Multi-Precision

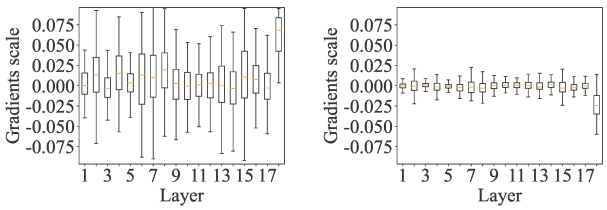
While our proposed *Double Rounding* method significantly advances multi-precision quantization, one-shot joint optimization across multiple precisions still faces a major challenge: strong competition between the highest and lowest bit-widths (Xu et al. 2022). During joint training, different precisions interact and interfere with each other, leading to notable discrepancies in their convergence rates, as illustrated in Figure 4 (a). Our experiments reveal that this competition is primarily caused by the *mismatch in the magnitudes of quantization scale gradients* between high-bit and low-bit quantization, as shown in Figure 3.

To address this issue, we propose Adaptive Learning Rate Scaling (AdaScale), a technique that dynamically adjusts the learning rates for different precisions to facilitate balanced optimization. AdaScale is inspired by the Layer-wise Adaptive Rate Scaling (LARS) (You, Gitman, and Ginsburg 2017) optimizer. Let λ denote the base learning rate for the current batch iteration. For each precision b, the learning rate λ_b is set as:

$$\lambda_b = \lambda \cdot \left(1 - \frac{1}{L}\sum_{i=1}^{L} \min\left(\left|\text{clip\_grad}(\nabla \mathbf{s}_b^i, 1.0)\right|, 1.0\right)\right) \tag{8}$$

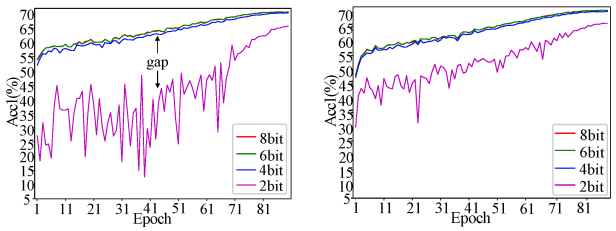
where, L is the number of layers, $\text{clip\_grad}(.)$ denotes gradient clipping to prevent explosion. $\nabla \mathbf{s}_b^i$ is the gradient of the quantization scale for layer i at precision b. AdaScale is applied exclusively to the quantization scale parameters, enabling adaptive learning rate adjustment across precisions. This ensures that quantization parameters for all bit-widths are optimized at a similar pace, effectively narrowing the convergence gap between high and low bits, see Figure 4 (b).

In the multi-precision setting, all candidate precisions share the same model weights during joint training. Conventionally, the shared weights are used for n forward passes



(a) 2-bit (b) 4-bit

Figure 3: The quantization scale gradients' statistics for the weights of ResNet18 on ImageNet-1K dataset, with outliers removed for clarity.



(a) w/o AdaScale (b) w. AdaScale

Figure 4: The multi-precision training processes of our *Double Rounding* on ResNet18 with ImageNet-1K dataset, without and with the AdaScale strategy.

per iteration (where n is the number of candidate bit-widths), the losses from each precision are accumulated, and gradients are computed before updating the shared parameters, see Algorithm 3 in the appendix. However, we observe that computing gradients and updating shared parameters separately for each precision loss—combined with AdaScale—yields better accuracy. Furthermore, we employ dual optimizers to update weight and quantization parameters simultaneously, and set the weight decay for quantization scales to zero to promote stable convergence, see Algorithm 1.

One-Shot Mixed-Precision SuperNet

In contrast to multi-precision approaches where all layers share the same bit-width, mixed-precision SuperNet enables layer-wise adaptive bit-width assignment, offering finer granularity and flexibility. Traditional mixed-precision methods often separate the training and search phases, requiring an additional retraining or fine-tuning stage for the selected SubNets. This decoupled process typically incurs high search costs, as it relies on strategies such as greedy (Cai and Vasconcelos 2020; Bulat et al. 2021) or genetic algorithms (Guo et al. 2020b; Xu et al. 2022) to identify optimal SubNets. Recognizing that each layer exhibits different sensitivity (Dong et al. 2019b) (i.e., importance) to quantization, we introduce a Hessian-Aware Stochastic Bit-switching (HessBit) strategy for efficient one-shot mixed-precision training.

Our approach leverages the Hessian Matrix Trace (HMT) to quantify the sensitivity of each layer. We first estimate the HMT of a pre-trained model using approximately 1000 training images (Dong et al. 2020), as illustrated in Figure 6 (a). During training, the HMT values guide the bit-switching process: for layers with HMT above the average (i.e., sensitive layers), higher bit-widths are favored; for less sensitive layers, all candidate bit-widths are selected with equal probability. This stochastic bit assignment is implemented via our proposed Roulette algorithm (see Algorithm 4 in appendix). Specifically, if a layer's HMT exceeds the mean HMT, the probability distribution in Figure 5 (b) is used; otherwise, the distribution in Figure 5 (a) is applied.

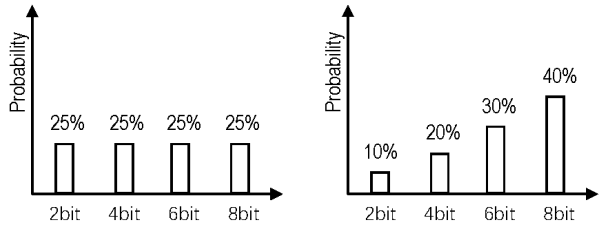
After the one-shot mixed-precision SuperNet is trained, we leverage Integer Linear Programming (ILP) (Ma et al. 2023) to efficiently search for optimal SubNets under various deployment constraints. Our ILP formulation integrates layer-wise sensitivity (quantified by the Hessian Matrix Trace, HMT) as a key factor in the objective, while also supporting constraints on FLOPs, latency, and parameter count. This approach enables us to directly extract a set of Pareto-optimal SubNets from the trained SuperNet (see Figure 6 (b)), eliminating the need for additional retraining and greatly reducing the overall search cost.

For a comprehensive understanding, we provide the detailed procedures for both the one-shot mixed-precision training and the SubNet search process in Algorithm 2 and Algorithm 5 in appendix, respectively. Notably, unlike multi-precision joint training, we replace standard BatchNorm (BN) layers with Transitional Batch-Norm (TBN) (Bulat et al. 2021) to mitigate distribution shifts between adjacent layers quantized to different bit-widths. To further enhance convergence, we propose a dynamic bit-switching threshold σ, which increases progressively with training epochs.

Algorithm 1: Our Multi-precision Training Procedure

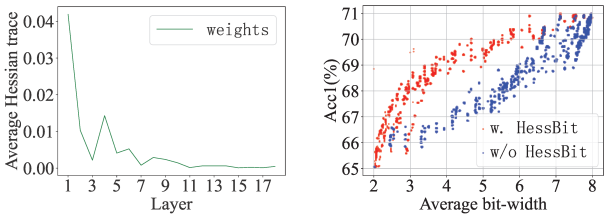
Require: Candidate bit-widths B; pretrained model M with FP32 weights W; quantization scales $\mathbf{s}$ (including $\mathbf{s}_w$ for weights and $\mathbf{s}_x$ for activations); BatchNorm layers $\{\text{BN}\}_{b=1}^{n}$; optimizers $optim_1(W, wd)$ and $optim_2(\mathbf{s}, wd = 0)$; base learning rate λ; weight decay wd; cross-entropy loss CE; training data D_{train}

for each epoch **do**
  Sample mini-batch $(\mathbf{x}, \mathbf{y})$ from D_{train}
  for each bit-width b in B **do**
    Forward pass for precision b:
    for each quantization layer **do**
      $\widehat{W}^b = dequant(quant(W, \mathbf{s}_w^b))$
      $\widehat{X}^b = dequant(quant(X, \mathbf{s}_x^b))$
      $O^b = Conv(\widehat{W}^b, \widehat{X}^b)$
    end for
    $\mathbf{o}^b = FC(W, O^b)$
    Update BatchNorm layer BN^b
    Compute loss: $\mathcal{L}^b = CE(\mathbf{o}^b, \mathbf{y})$
    Backward: $\mathcal{L}^b.\text{backward}()$
    Compute adaptive learning rate λ_b # see Eq. (8)
    Update weights and quantization scales: $optim_1.step(\lambda)$; $optim_2.step(\lambda_b)$ and then clear gradients: $optim_1.zero\_grad()$; $optim_2.zero\_grad()$
  end for
end for



(a) Unsensitive (b) Sensitive

Figure 5: The HessBit stochastic bit-switching process. (a) Probability distribution for unsensitive layers. (b) Probability distribution for sensitive layers.



(a) Hessian trace (b) Mixed precision

Figure 6: The Hessian trace and mixed-precision results of ResNet18 for {2,4,6,8}-bit.

Experiments

Image Classification

Setup. We evaluate our approach on the ImageNet-1K classification task using both standard (ResNet18/50) and lightweight (MobileNetV2) architectures, following common experimental protocols. Our experiments cover both multi-precision and mixed-precision joint quantization training. We consider two candidate bit-width sets: {8,6,4,2} and {4,3,2}, where each value denotes the quantization level for weights and activations. Consistent with prior work, batch normalization layers are not quantized, and the first and last layers remain in full precision. Multi-precision models are initialized from pre-trained FP32 models, while mixed-precision models are initialized from pre-trained multi-precision models. All models are trained for 90 epochs with the Adam optimizer (batch size 256), a cosine learning rate schedule (no warm-up), initial learning rate of 5e-4, and weight decay of 5e-5. Standard data augmentation is applied, including random cropping, resizing to 224×224, and random flipping. For evaluation, images are resized to 256×256 and center-cropped to 224×224. All experiments are conducted on a system equipped with an Nvidia Tesla V100 GPU.

Results in Multi-Precision. Table 1 summarizes the Top-1 validation accuracy for multi-precision quantization across various bit-width configurations. Our *double-rounding* method, combined with AdaScale, consistently outperforms previous SOTA approaches. For the {8,6,4,2}-bit setting, our method achieves the best results on both ResNet18 and ResNet50. Specifically, on ResNet18, our approach surpasses Any-Precision by up to 2.83% (w8a8, with KD) and outperforms MultiQuant by up to 0.73% (w4a4, with KD). On MobileNetV2, since prior methods fail to converge with 2-bit included, we focus on the {8,6,4}-bit results for fair comparison, where our method also achieves the highest accuracy, outperforming AdaBits and other baselines. Notably, our method achieves 70.87% (w8a8, with KD) and 69.77% (w4a4, with KD) on ResNet18, and 76.98% (w8a8, with KD) and 76.52% (w4a4, with KD) on ResNet50.

For the {4,3,2}-bit configuration, our *double-rounding* approach continues to demonstrate clear advantages. On ResNet18, it achieves up to 1.02% (w2a2, with KD) and 1.2% (w2a2, no KD) higher accuracy than Bit-Mixer, and up to 1.12% improvement over ABN (with KD). On ResNet50, our method consistently outperforms Bit-Mixer across all bit-widths, achieving 76.42% (w4a4, with KD) and 73.28% (w2a2, with KD). It is also observed that the overall accuracy for the {4,3,2}-bit set is lower than that of the {8,6,4,2}-bit set, mainly due to increased information loss in the shared low-precision model, which makes optimization more challenging. Therefore, we recommend including 8-bit in multi-precision training. Additionally, learning separate quantization scales for each precision (for both weights and activations) can further improve accuracy, but this requires storing the model in 32-bit format, as shown in the 'Double Rounding*' rows on the ResNet50 experiment.

Results in Mixed-Precision. As shown in Table 2, Double Rounding achieves excellent accuracy on both ResNet18

Algorithm 2: One-shot Mixed-Precision SuperNet Training

Require: Candidate bit-widths B; HMT for each layer $t_l \in \{T\}_{l=1}^{L}$; average HMT $t_m = \frac{1}{L}\sum_{l=1}^{L} t_l$;

Initialize: Pretrained model M with FP32 weights W; quantization scales $\mathbf{s}$ for weights ($\mathbf{s}_w$) and activations ($\mathbf{s}_x$); TBN layers $\{TBN\}_{b=1}^{n^2}$; bit-switching threshold σ; optimizer $optim(W, \mathbf{s}, wd)$; learning rate λ; weight decay wd.

for each epoch **do**
  Update bit-switching threshold: $\sigma = \sigma \times \frac{epoch+1}{total\_epochs}$
  for each mini-batch $(\mathbf{x}, \mathbf{y})$ from D_{train} **do**
    for each candidate bit-width b in B **do**
      Forward pass: $forward(M, \mathbf{x}, \mathbf{y}, b, T, t_m)$
      for each quantization layer **do**
        Sample $r \sim U[0, 1]$
        if $r < \sigma$ **then**
          $b = Roulette(B, t_l, t_m)$ # See Algorithm 4 in Appendix
        end if
        $\widehat{W}^b = dequant(quant(W, \mathbf{s}_w^b))$
        $\widehat{X}^b = dequant(quant(X, \mathbf{s}_x^b))$
        $O^b = Conv(\widehat{W}^b, \widehat{X}^b)$
      end for
      $\mathbf{o}^b = FC(W, O^b)$
      Update TBN^b layer
      Compute loss: $\mathcal{L}^b = CE(\mathbf{o}^b, \mathbf{y})$
      Backward: $\mathcal{L}^b$.backward()
      Update weights and quantization scales and then clear gradients: $optim.step(\lambda)$; $optim.zero\_grad()$
    end for
  end for
end for

Model	Method	KD	Storage	Epoch	Bit Config	w8a8	w6a6	w4a4	w3a3	w2a2	FP
ResNet18	Hot-Swap (Sun et al. 2021)	✗	32bit	–	{8,6,4,2}	70.40	70.30	70.20	–	64.90	–
	L1 (Alizadeh et al. 2020)	✗	32bit	–	{8,6,4,2}	69.92	66.39	0.22	–	–	70.07
	KURE (Chmiel et al. 2020)	✗	32bit	80	{8,6,4,2}	70.20	70.00	66.90	–	–	70.30
	Double Rounding (**ours**)	✗	8bit	90	{8,6,4,2}	70.74	70.71	70.43	–	66.35	69.76
	Any-Precision (Yu et al. 2021)	✓	32bit	80	{8,6,4,2}	68.04	–	67.96	–	64.19	69.27
	CoQuant (Sun et al. 2024)	✓	8bit	100	{8,6,4,2}	67.90	67.60	66.60	–	57.10	69.90
	MultiQuant (Xu et al. 2022)	✓	32bit	90	{8,6,4,2}	70.28	70.14	69.80	–	66.56	–
	BitWave (Shi et al. 2024)	✓	8bit	90	{8,6,4,2}	70.16	70.08	69.62	–	66.15	69.76
	Double Rounding (**ours**)	✓	8bit	90	{8,6,4,2}	**70.87**	**70.79**	**70.53**	–	**66.84**	69.76
	Bit-Mixer (Bulat et al. 2021)	✗	4bit	160	{4,3,2}	–	–	69.10	68.50	65.10	69.60
	Vertical-layer (Wu et al. 2023)	✗	4bit	300	{4,3,2}	–	–	69.20	68.80	66.60	70.50
	Double Rounding (**ours**)	✗	4bit	90	{4,3,2}	–	–	69.73	69.20	66.30	69.76
	Q-DNNs (Kunyuan Du 2020)	✓	32bit	45	{4,3,2}	–	–	66.94	66.28	62.91	68.60
	ABN (Tang et al. 2022)	✓	4bit	160	{4,3,2}	–	–	68.90	68.60	65.50	–
	Bit-Mixer (Bulat et al. 2021)	✓	4bit	160	{4,3,2}	–	–	69.40	68.70	65.60	69.60
	BitWave (Shi et al. 2024)	✓	4bit	160	{4,3,2}	–	–	69.40	68.70	65.60	69.76
	Double Rounding (**ours**)	✓	4bit	90	{4,3,2}	–	–	**69.77**	**69.34**	**66.62**	69.76
ResNet50	Any-Precision (Yu et al. 2021)	✗	32bit	80	{8,6,4,2}	74.68	–	74.43	–	72.88	–
	Hot-Swap (Sun et al. 2021)	✗	32bit	–	{8,6,4,2}	75.60	75.50	75.30	–	71.90	–
	KURE (Chmiel et al. 2020)	✗	32bit	80	{8,6,4,2}	–	76.20	74.30	–	–	76.30
	Double Rounding (**ours**)	✗	8bit	90	{8,6,4,2}	76.51	76.28	75.74	–	72.31	76.13
	Any-Precision (Yu et al. 2021)	✓	32bit	80	{8,6,4,2}	74.91	–	74.75	–	73.24	75.95
	MultiQuant (Xu et al. 2022)	✓	32bit	90	{8,6,4,2}	76.94	76.85	76.46	–	73.76	76.13
	Double Rounding (**ours**)	✓	8bit	90	{8,6,4,2}	**76.98**	**76.86**	**76.52**	–	**73.78**	76.13
	Double Rounding (**ours**)	✗	4bit	90	{4,3,2}	–	–	75.81	75.24	71.62	76.13
	AdaBits (Jin Qing 2020)	✗	32bit	150	{4,3,2}	–	–	76.10	75.80	73.20	75.00
	Double Rounding* (**ours**)	✗	32bit	90	{4,3,2}	–	–	**76.42**	**75.82**	**73.28**	76.13
	Bit-Mixer (Bulat et al. 2021)	✓	4bit	160	{4,3,2}	–	–	75.20	74.90	72.70	–
	Double Rounding (**ours**)	✓	4bit	90	{4,3,2}	–	–	76.06	75.53	72.80	76.13
MobileNetV2	AdaBits (Jin Qing 2020)	✗	8bit	150	{8,6,4}	72.30	72.30	70.30	–	–	71.80
	KURE (Chmiel et al. 2020)	✗	32bit	80	{8,6,4}	–	70.00	59.00	–	–	71.30
	BitWave (Shi et al. 2024)	✗	8bit	90	{8,6,4}	72.23	71.35	69.84	–	–	71.14
	Double Rounding (**ours**)	✗	8bit	90	{8,6,4}	72.42	72.06	69.92	–	–	71.14
	MultiQuant (Xu et al. 2022)	✓	32bit	90	{8,6,4}	72.33	72.09	70.59	–	–	71.88
	Double Rounding (**ours**)	✓	8bit	90	{8,6,4}	**72.55**	**72.41**	**70.86**	–	–	71.14
	Double Rounding (**ours**)	✗	8bit	90	{8,6,4,2}	70.98	70.70	68.77	–	50.43	71.14
	Double Rounding (**ours**)	✓	8bit	90	{8,6,4,2}	71.35	71.20	69.85	–	**53.06**	71.14

Table 1: Top-1 accuracy (%) on ImageNet-1K for multi-precision quantization. "Bit Config" indicates the candidate bit-width set. "KD" denotes knowledge distillation. "–" indicates unavailable results. Bold numbers indicate the best results.

and ResNet50. On ResNet18, it outperforms Bit-Mixer by up to 0.83% (w4a4) and achieves the best results across all bit-widths, including a 0.12% gain over EQ-Net in the 3MP setting. On ResNet50, Double Rounding consistently surpasses previous methods such as Bit-Mixer and EQ-Net, showing clear advantages in both w4a4 and 3MP. These results demonstrate the effectiveness of our Hessian-based sensitivity-guided bit-width allocation for mixed-precision.

Object Detection and Segmentation

Setup. We further validate the generalization of our approach on object detection and instance segmentation tasks using Mask-RCNN (He et al. 2017) with pre-trained ResNet backbones on the MS-COCO 2017 dataset (118K training and 5K validation images). All experiments are conducted following the official PyTorch codebase, with AdamW optimizer, 26 training epochs, batch size 16, and default hyperparameters without additional tuning.

Results. As reported in Table 3, our Double Rounding method achieves highly competitive mAP results for both detection and segmentation across all bit-widths on both ResNet18 and ResNet50 backbones. Notably, Double Rounding consistently outperforms prior methods such as Bit-Mixer, MultiQuant, and BitWave, especially at lower bit-widths (e.g., w2a2), demonstrating strong robustness and generalization.

Ablation Studies

AdaScale vs. Conventional Multi-Precision. To assess the impact of AdaScale, we conduct ablation experiments without knowledge distillation (KD), as shown in Table 4. AdaScale consistently improves accuracy, especially at lower bit-widths. For MobileNetV2, which struggles to converge with {4,3,2}-bit as in prior works, we use {8,6,4}-bit for fair comparison (see Table 8 in appendix). Our method achieves stable convergence even with {8,6,4,2}-bit, highlighting the effectiveness of both *Double Rounding* and AdaScale.

Model	Method	KD	Training	Searching	Fine-tune	Epoch	w4a4	w3a3	w2a2	3MP	FP
ResNet18	Double Rounding (**ours**)	✗	HessBit	ILP	w/o	90	69.80	68.63	64.88	68.85	69.76
	Bit-Mixer (Bulat et al. 2021)	✓	Random	Greedy	w/o	160	69.20	68.60	64.40	62.90	69.60
	ABN (Tang et al. 2022)	✓	DRL	DRL	w.	160	69.80	69.00	65.20	67.70	–
	MultiQuant (Xu et al. 2022)	✓	LRH	Genetic	w.	90	–	67.50	–	69.20	69.76
	EQ-Net (Xu et al. 2023)	✓	LRH	Genetic	w.	120	–	69.30	65.90	69.80	69.76
	Double Rounding (**ours**)	✓	HessBit	ILP	w/o	90	**70.03**	**69.32**	**65.97**	**69.92**	69.76
ResNet50	Double Rounding (**ours**)	✗	HessBit	ILP	w/o	90	75.01	74.31	71.47	75.06	76.13
	Bit-Mixer (Bulat et al. 2021)	✓	Random	Greedy	w/o	160	75.20	74.80	72.10	73.20	–
	EQ-Net (Xu et al. 2023)	✓	LRH	Genetic	w.	120	–	74.70	72.50	75.10	76.13
	Double Rounding (**ours**)	✓	HessBit	ILP	w/o	90	**75.63**	**74.88**	**72.61**	**75.24**	76.13

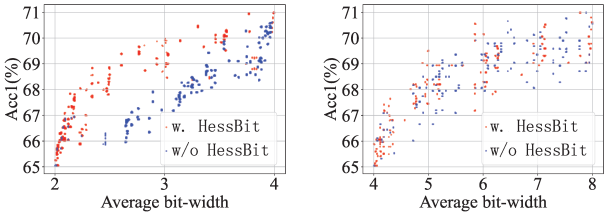
Table 2: Top-1 accuracy (%) on ImageNet-1K for mixed-precision quantization ({4,3,2}-bit). "MP" denotes average bit-width for mixed-precision. "–" denotes unavailable results. "FP" denotes full-precision model. Bold numbers indicate the best results.

Backbone	Method	Bit Config	Object Detection					Instance Segmentation				
			FP	w8a8	w6a6	w4a4	w2a2	FP	w8a8	w6a6	w4a4	w2a2
ResNet18	Bit-Mixer (Bulat et al. 2021)	{8,6,4,2}	27.3	27.4	26.8	25.9	19.8	25.6	25.7	25.0	24.3	19.0
	MultiQuant (Xu et al. 2022)	{8,6,4,2}	27.3	27.6	27.0	26.2	20.8	25.6	25.7	25.1	24.5	20.6
	BitWave (Shi et al. 2024)	{8,6,4,2}	27.3	27.5	26.9	25.7	19.2	25.6	25.6	24.8	24.1	18.5
	Double Rounding (**ours**)	{8,6,4,2}	27.3	**27.9**	**27.3**	**26.7**	**21.7**	25.6	**25.9**	**25.2**	**24.8**	**21.0**
ResNet50	Bit-Mixer (Bulat et al. 2021)	{8,6,4,2}	37.9	36.9	36.3	34.2	24.8	34.6	33.6	32.2	31.2	23.5
	MultiQuant (Xu et al. 2022)	{8,6,4,2}	37.9	36.7	36.1	34.5	25.6	34.6	33.4	32.1	31.5	24.8
	BitWave (Shi et al. 2024)	{8,6,4,2}	37.9	36.8	36.0	33.8	24.2	34.6	33.4	31.8	30.8	22.8
	Double Rounding (**ours**)	{8,6,4,2}	37.9	**37.2**	**36.8**	**35.1**	**26.8**	34.6	**33.9**	**32.6**	**31.9**	**25.3**

Table 3: Multi-precision results for object detection and instance segmentation on COCO. Bold numbers indicate the best results.

Model	AdaScale	w8a8	w6a6	w4a4	w2a2	FP
ResNet20	w/o	92.17	92.20	92.17	89.67	92.30
	w.	92.25	92.32	92.19	90.19	92.30
ResNet18	w/o	70.05	69.80	69.32	65.83	69.76
	w.	70.74	70.71	70.43	66.35	69.76
ResNet50	w/o	76.18	76.08	75.64	70.28	76.13
	w.	76.51	76.28	75.74	72.31	76.13
MobileNetV2	w/o	70.55	70.65	68.08	45.00	71.14
	w.	70.98	70.70	68.77	50.43	71.14

Table 4: Ablation studies on multi-precision {8,6,4,2}-bit: ResNet20 (CIFAR-10) and other models (ImageNet-1K).



(a) {4,3,2}-bit (b) {8,4}-bit

Figure 7: Comparison of HessBit and baseline for mixed-precision on ResNet18.

Pareto Frontier for Mixed-Precision Configurations. We further evaluate the HessBit strategy by searching for mixed-precision configurations with different bit-lists. Figure 7 shows the Pareto frontiers for {4,3,2}-bit and {8,4}-bit on ResNet18. Each point represents a SubNet sampled via ILP without retraining or fine-tuning. The red points (HessBit) consistently outperform the blue baseline at the same bit-width, demonstrating the effectiveness of our approach.

Additional experimental results can be found in Appendix Section C, including Separate-Precision quantization (Table 5), a comparison of training costs between Multi-Precision and Separate-Precision (Table 7), and Multi-Precision quantization results on LLMs (Table 6).

Conclusion

We propose a unified framework for efficient multi-precision and mixed-precision quantization. Our *Double Rounding* method enables joint training across multiple bit-widths with a single set of integer weights, supporting efficient and nearly lossless bit-switching. AdaScale adaptively scales learning rates to close the convergence gap between high- and low-precision branches, improving overall accuracy. The Hessian-Aware Stochastic Bit-switching (HessBit) strategy, combined with an efficient Integer Linear Programming-based search, enables one-shot mixed-precision SuperNet training and effective Pareto frontier approximation. Together, these techniques offer a practical and flexible solution for model compression under various storage and computational constraints.

Acknowledgements

This work was supported in part by National Natural Science Foundation of China No. 62088102, No.62302381 and No.52441602. The authors are with the National Key Laboratory of Human-Machine Hybrid Augmented Intelligence, National Engineering Research Center of Visual Information and Applications, and Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University, Xi'an, Shaanxi, China.

References

Alizadeh, M.; Behboodi, A.; van Baalen, M.; Louizos, C.; Blankevoort, T.; and Welling, M. 2020. Gradient L1 regularization for quantization robustness. *arXiv preprint arXiv:2002.07520*.

Bengio, Y.; Léonard, N.; and Courville, A. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*.

Bulat, A.; Tzimiropoulos, A., Georgios; Tzimiropoulos, L., Tzimiropoulosd; and Liu, B. 2021. Bit-Mixer: Mixed-precision networks with runtime bit-width selection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5188–5197.

Cai, Z.; and Vasconcelos, N. 2020. Rethinking differentiable search for mixed-precision neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2349–2358.

Chmiel, B.; Banner, R.; Shomron, G.; Nahshan, Y.; Bronstein, A.; Weiser, U.; et al. 2020. Robust quantization: One model to rule them all. *Advances in neural information processing systems*, 33: 5308–5317.

Dong, Y.; Ni, R.; Li, J.; Chen, Y.; Su, H.; and Zhu, J. 2019a. Stochastic quantization for learning accurate low-bit deep neural networks. *International Journal of Computer Vision*, 127: 1629–1642.

Dong, Z.; Yao, Z.; Arfeen, D.; Gholami, A.; Mahoney, M. W.; and Keutzer, K. 2020. Hawq-v2: Hessian aware trace-weighted quantization of neural networks. *Advances in neural information processing systems*, 33: 18518–18529.

Dong, Z.; Yao, Z.; Gholami, A.; Mahoney, M. W.; and Keutzer, K. 2019b. Hawq: Hessian aware quantization of neural networks with mixed-precision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 293–302.

Elthakeb, A.; Pilligundla, P.; Mireshghallah, F.; Yazdanbakhsh, A.; Gao, S.; and Esmaeilzadeh, H. 2019. Releq: an automatic reinforcement learning approach for deep quantization of neural networks. In *NeurIPS ML for Systems workshop, 2018*.

Esser, S. K.; McKinstry, J. L.; Bablani, D.; Appuswamy, R.; and Modha, D. S. 2019. Learned step size quantization. *arXiv preprint arXiv:1902.08153*.

Guo, Z.; Zhang, X.; Mu, H.; Heng, W.; Liu, Z.; Wei, Y.; and Sun, J. 2020a. Single path one-shot neural architecture search with uniform sampling. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, 544–560. Springer.

Guo, Z.; Zhang, X.; Mu, H.; Heng, W.; Liu, Z.; Wei, Y.; and Sun, J. 2020b. Single path one-shot neural architecture search with uniform sampling. In *European conference on computer vision*, 544–560. Springer.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 2961–2969.

Jacob, B.; Kligys, S.; Chen, B.; Zhu, M.; Tang, M.; Howard, A. G.; Adam, H.; and Kalenichenko, D. 2017. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. *CoRR*, abs/1712.05877.

Jin, Q.; Yang, L.; and Liao, Z. 2019. Towards efficient training for neural network quantization. *arXiv preprint arXiv:1912.10207*.

Jin Qing, L. Z., Yang Linjie. 2020. Adabits: Neural network quantization with adaptive bit-widths. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2146–2156.

Kim, K.; Ji, B.; Yoon, D.; and Hwang, S. 2021. Self-knowledge distillation with progressive refinement of targets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 6567–6576.

Kunyuan Du, H. G., Ya Zhang. 2020. From Quantized DNNs to Quantizable DNNs. *CoRR*, abs/2004.05284.

Liu, J.; Cai, J.; and Zhuang, B. 2021. Sharpness-aware quantization for deep neural networks. *arXiv preprint arXiv:2111.12273*.

Ma, Y.; Jin, T.; Zheng, X.; Wang, Y.; Li, H.; Wu, Y.; Jiang, G.; Zhang, W.; and Ji, R. 2023. Ompq: Orthogonal mixed precision quantization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 9029–9037.

Polino, A.; Pascanu, R.; and Alistarh, D. 2018. Model compression via distillation and quantization. In *International Conference on Learning Representations*.

Shen, M.; Liang, F.; Gong, R.; Li, Y.; Li, C.; Lin, C.; Yu, F.; Yan, J.; and Ouyang, W. 2021. Once Quantization-Aware Training: High Performance Extremely Low-Bit Architecture Search. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 5340–5349.

Shi, M.; Jain, V.; Joseph, A.; Meijer, M.; and Verhelst, M. 2024. BitWave: Exploiting Column-Based Bit-Level Sparsity for Deep Learning Acceleration. In *2024 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, 732–746. IEEE.

Sun, Q.; Li, X.; Ren, Y.; Huang, Z.; Liu, X.; Jiao, L.; and Liu, F. 2021. One model for all quantization: A quantized network supporting hot-swap bit-width adjustment. *arXiv preprint arXiv:2105.01353*.

Sun, X.; Panda, R.; Chen, C.-F. R.; Wang, N.; Pan, B.; Oliva, A.; Feris, R.; and Saenko, K. 2024. Improved Techniques for Quantizing Deep Networks with Adaptive Bit-Widths. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 957–967.

Tang, C.; Zhai, H.; Ouyang, K.; Wang, Z.; Zhu, Y.; and Zhu, W. 2022. Arbitrary Bit-width Network: A Joint Layer-Wise Quantization and Adaptive Inference Approach.

Wang, K.; Liu, Z.; Lin, Y.; Lin, J.; and Han, S. 2019. Haq: Hardware-aware automated quantization with mixed precision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8612–8620.

Wu, B.; Wang, Y.; Zhang, P.; Tian, Y.; Vajda, P.; and Keutzer, K. 2018. Mixed precision quantization of convnets via differentiable neural architecture search. *arXiv preprint arXiv:1812.00090*.

Wu, H.; He, R.; Tan, H.; Qi, X.; and Huang, K. 2023. Vertical Layering of Quantized Neural Networks for Heterogeneous Inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(12): 15964–15978.

Xu, K.; Feng, Q.; Zhang, X.; and Wang, D. 2022. MultiQuant: Training Once for Multi-bit Quantization of Neural Networks. In Raedt, L. D., ed., *IJCAI*, 3629–3635. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Xu, K.; Han, L.; Tian, Y.; Yang, S.; and Zhang, X. 2023. Eq-net: Elastic quantization neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 1505–1514.

Yao, Z.; Dong, Z.; Zheng, Z.; Gholami, A.; Yu, J.; Tan, E.; Wang, L.; Huang, Q.; Wang, Y.; Mahoney, M.; et al. 2021. Hawq-v3: Dyadic neural network quantization. In *International Conference on Machine Learning*, 11875–11886. PMLR.

You, Y.; Gitman, I.; and Ginsburg, B. 2017. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*.

Yu, H.; Li, H.; Shi, H.; Huang, T. S.; and Hua, G. 2021. Any-precision deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 10763–10771.

Zhang, S. Q.; McDanel, B.; Kung, H.; and Dong, X. 2021. Training for multi-resolution inference using reusable quantization terms. In *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, 845–860.

Zhou, S.; Wu, Y.; Ni, Z.; Zhou, X.; Wen, H.; and Zou, Y. 2016. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*.