Retraining-free Model Quantization via One-Shot Weight-Coupling Learning

Chen Tang[1*] Yuan Meng[1*] Jiacheng Jiang[1] Shuzhao Xie[1] Rongwei Lu[1]
Xinzhu Ma[2] Zhi Wang[1†] Wenwu Zhu[1†]
[1]Tsinghua University [2]The Chinese University of Hong Kong

Abstract

Quantization is of significance for compressing the over-parameterized deep neural models and deploying them on resource-limited devices. Fixed-precision quantization suffers from performance drop due to the limited numerical representation ability. Conversely, mixed-precision quantization (MPQ) is advocated to compress the model effectively by allocating heterogeneous bit-width for layers. MPQ is typically organized into a searching-retraining two-stage process. Previous works only focus on determining the optimal bit-width configuration in the first stage efficiently, while ignoring the considerable time costs in the second stage and thus hindering deployment efficiency significantly. In this paper, we devise a one-shot training-searching paradigm for mixed-precision model compression. Specifically, in the first stage, all potential bit-width configurations are coupled and thus optimized simultaneously within a set of shared weights. However, our observations reveal a previously unseen and severe bit-width interference phenomenon among highly coupled weights during optimization, leading to considerable performance degradation under a high compression ratio. To tackle this problem, we first design a bit-width scheduler to dynamically freeze the most turbulent bit-width of layers during training, to ensure the rest bit-widths converged properly. Then, taking inspiration from information theory, we present an information distortion mitigation technique to align the behaviour of the bad-performing bit-widths to the well-performing ones. In the second stage, an inference-only greedy search scheme is devised to evaluate the goodness of configurations without introducing any additional training costs. Extensive experiments on three representative models and three datasets demonstrate the effectiveness of the proposed method. Code can be available on https://github.com/1hunters/retraining-free-quantization.

1. Introduction

Recent years have witnessed the tremendous achievements made by deep network-driven applications, ranging from classification [19, 22, 42], object detection [40, 41, 46], and segmentation [5, 20]. However, the huge number of parameters in these deep models poses intractable challenges for both training [4, 24, 32] and inference [16, 39]. To enable efficient deep learning on inference, several techniques are proposed, including pruning [31, 33], quantization [12, 47, 67], and neural architecture search [21, 50].

Ultra-low bit-width neural network quantization [12, 53, 67] is an appealing model compression technique to simplify the hardware complexity and improve the runtime efficiency of over-parameterized deep models. However, under severely limited numerical representation capabilities, performing such compression across the whole neural network usually incurs an unacceptable performance drop. Mixed-precision quantization (MPQ) [3, 15, 23, 47, 57], by allocating unequally bit-width for weight and activation tensors of each layer, can largely avoid accuracy degradation while maintaining the proper model size and runtime overhead (*e.g.* on-device latency). The underlying principle of MPQ is that layers contribute very differently to the final accuracy [3, 47, 57], so the compression algorithm should apply heterogeneous precision rather than a uniform one across the whole network. Besides, hardware starts to support mixed-precision computation [6, 57] in these years, which further pushes the study for mixed-precision compression.

Recently, MPQ has been extensively studied from several perspectives, *e.g.* through reinforcement learning [11, 57], differentiable methods [3, 60], and proxy-based approaches [6, 9]. They all try to solve one challenge, that says, how to find the optimal bit-width configuration for each layer, in an exponentially large $\mathcal{O}(n^{2L})$ space, where n is the number of bit-width candidates and L is the number of layers in the deep network. To this end, they organize a *searching-then-retraining pipeline*, in which the first stage aims to finish the bit-width allocation as fast as possible, and naturally becomes the focus of the research. Nevertheless, previous works tend to ignore the importance of the second stage, which in fact consumes considerable time cost for retraining the model to fit the obtained bit-width configurations (a.k.a, the policy). To recover the performance, LIMPQ [47] needs about 200 GPU-hours to re-

*Equal contributions. †Corresponding authors. Z. Wang is also with TBSI.

train a single ResNet18 policy. This impedes the real-world quantized mixed-precision model deployment—we might not have much time to retrain every policies for all devices.

Instead, we consider a new paradigm termed as *training-then-searching*, repositioning the resource-intensive training process to the forefront of the mixed-precision quantization pipeline. The initial stage focuses on the development of a weight-sharing quantization model, where all possible bit-width configurations are concurrently optimized within unified network of weights to fulfill extensive search requirements. Importantly, this weight-sharing model undergoes a singular training session and, notably, *requires no subsequent retraining following the search*. Subsequently, in the second stage, we present an inference-only search employing a bidirectional greedy scheme to judiciously determine the optimal bit-width for each layer.

The primary focus of this paper lies in the training of a high-quality weight-sharing quantization model, which highly relies on ingenious weight-coupling learning method with heterogeneous bit-widths. We identify a distinctive phenomenon inherent in weight-sharing quantization—referred to as the *bit-width interference problem*. This problem arises from the highly shared weights between bit-widths, the same weight could be quantized to very different discrete values for various bit-widths, so significantly superimposed quantization noise of various bit-widths leads to daunting optimization challenges, as we will discuss later. We illustrate the bit-width interference problem in Fig. 1, one can see that even the introduction of a single additional bit-width can cause the shared weight to frequently traverse quantization bound, resulting in training instability and large gradient variance.

To understand and circumvent the issue of weight-sharing quantization, we conduct a detailed analysis of the bit-width interference problem (Sec. 3.2). Building upon this understanding, we introduce a bit-width scheduler designed to freeze the bit-widths that contribute to weight interference, ensuring proper convergence for the remaining bit-widths. Furthermore, during dynamic training, we observe an information distortion phenomenon associated with the unfrozen bit-widths. To mitigate this distortion, we propose to align the behavior of poorly performing bit-widths with their well-performing counterparts. Extensive experiments demonstrate that these two complementary techniques not only unravel the intricacies of the bit-width interference problem but also provide meaningful performance improvements of weight-sharing quantization models. To summarize, our contributions are as follows:

- We identify and analyze the bit-width interference problem in weight-sharing quantization models, revealing its impact on optimization challenges, training stability, and convergence.
- To train the weight-sharing quantization model, we first design a novel bit-width scheduler that freezes interfering bit-widths during training, ensuring proper convergence and addressing instability caused by the introduction of additional bit-widths.
- We also propose a strategy inspired by information theory to align poorly performing bit-widths with their well-performing counterparts, mitigating information distortion during dynamic training and enhancing the overall performance.
- Extensive experiments on three representative models and three benchmarks demonstrate the effectiveness of proposed method. For example, under an average 4-bit constraint, our method leads on ResNet with a top accuracy of 71.0% at only 31.6G BitOPs and no retraining cost, compared to the second-best at 70.8% accuracy with higher 33.7G operations and 90 epochs of retraining.

2. Related Work

2.1. Neural Network Quantization

In this paper, we only consider the context in quantization-aware training, as it can achieve higher compression ratio than post-training quantization [25, 34]. Quantization can be generally categorized into two classes: fixed-precision quantization and mixed-precision quantization.

Fixed-Precision Quantization. Fixed-precision quantization involves assigning a uniform bit-width to all layers. Specifically, methods such as Dorefa [67] and PACT [7] employ a low-precision representation for weights and activations during forward propagation. They leverage the Straight-Through Estimation (STE) technique [1] to estimate the gradient of the piece-wise quantization function for backward propagation. LSQ [12] scales the weight and activation distributions by introducing learnable step-size scale factors for quantization functions.

Mixed-Precision Quantization. Mixed-Precision Quantization (MPQ) delves into the intricate aspects of low-precision quantization by recognizing the inherent variability in redundancy across different layers of the deep model. By allocating smaller bit-widths to layers with high redundancy, MPQ optimizes model complexity without causing a significant performance decline. The challenge, however, lies in determining the most suitable bit-width for each layer, considering that the bit-width selection is a discrete process, and the potential combinations of bit-width and layer (referred to as "policy") grow exponentially.

Strategies such as HAQ [57] and ReleQ [11] leverage reinforcement learning (RL) techniques to derive a bit-width allocator. SPOS [14], EdMIPS [3], BP-NAS [65], GMPQ [60] and SEAM [49] adopt differential neural architecture search (NAS) methods to learn the bit-width. However, these methods require to both search-from-scratch and train-from-scratch for the models when changing the

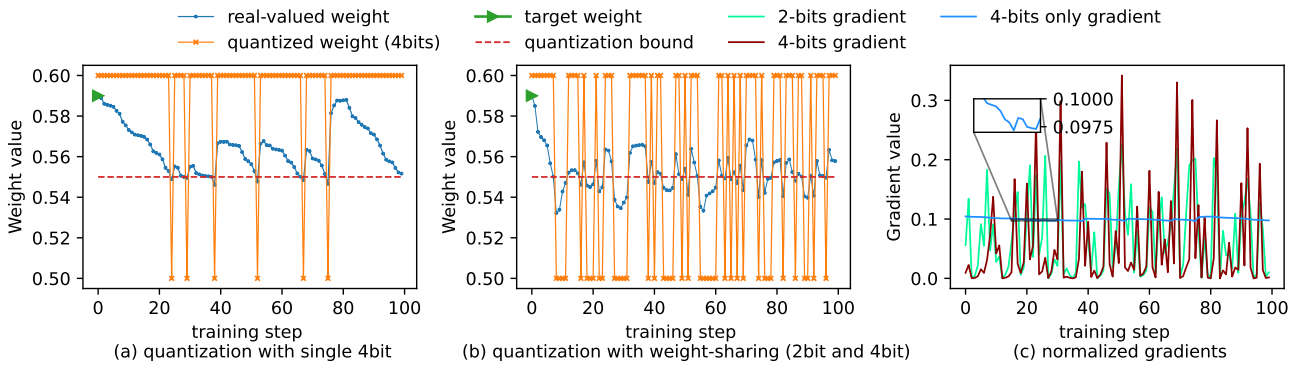

Figure 1. **(a)** 2D regression of *single* 4-bits quantization, **(b)** 2D regression of 4-bits quantization with an additional 2-bits (*i.e.,* weight-sharing quantization), and **(c)** the L2-normalized gradients of these two regressions. Compared with Fig. 1(a), the weight in Fig. 1(b) is more unstable due to the bit-width interference. Notably, the gradient of 4-bits also has a larger variance under weight-sharing.

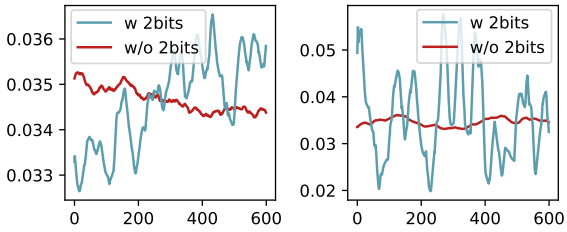

Figure 2. Distance between full-precision latent weights and quantized weights on MobileNetV2 of a point-wise conv layer. Left: 4-bits. Right: 6-bits.

search constraints. Unlike learning the optimal MPQ policy, HAWQ [9, 10] and MPQCO [6] use the Hessian information as the quantization sensitivity metrics to assist bit-width assignment. LIMPQ [47] proposes to learn the layer-wise importance within a single quantization-aware training cycle.

2.2. Deep Learning with Weight-Sharing

Weight-sharing is an effective and practical technique to reuse weight to deal with joint task learning or avoid the need to store multiple copies of weights. Generally speaking, there have been two relevant topics to weight-sharing with this work, covering neural architecture search (NAS) and dynamic neural network.

Neural Architecture Search. NAS [21, 45, 69] aims to automatically discover the well-performing topology of deep neural network in a vast search space, which composes of various operators (*e.g.* convolutional layers with different kernel-size or channels). To improve the search efficiency, recent works [14, 38, 50] both adopt the idea of weight-sharing to stuff the candidates into a shared and large meta-topology (a.k.a. the super-network). Weight-sharing allows to train directly the meta-topology and derive a sub-topology from the meta-topology to evaluate. This significantly shortens the evaluation time of the goodness for a given topology [30, 38]. Although certain MPQ research [3, 60] leverages this NAS-style searching process, they still pay significant time for retraining.

Dynamic Neural Network. Unlike conventional neural networks which are architecturally fixed during inference, dynamic neural networks [18] enable dynamic computational paths on demand according to various input samples or running environments. For example, runtime channel pruning [29, 59] dynamically activates channels of layers and layer skipping [43, 56, 58] adjusts depths of layers for different input images. To support the adaptive inference, the weight-sharing idea is applied to avoid substantial copies of weights. Therefore, they can both achieve a better accuracy-efficiency trade-off compared to their static counterparts. These have also been several works of dynamic bit-width neural networks [2, 26, 48, 62]. However, they either provide only the fixed-precision quantization that supports very limited bit-width configurations or have to drop the ultra-low bit-widths (*e.g.* 2 bits, 3bits, *etc.*) to guarantee the training convergence but lose the chance for achieving high compression ratio.

3. Methodology

3.1. Preliminary

Quantization. The uniform quantization function under b bits in quantization-aware training (QAT) maps the input full-precision activations and weights to the homologous quantized values $[0, 2^b - 1]$ and $[-2^{b-1}, 2^{b-1} - 1]$. The quantization functions $Q_b(\cdot)$ that quantize the input values x to quantized values $\hat{x}$ can be expressed as follows:

$$\hat{x} = Q_b(x;\gamma) = \lfloor \text{clip}(\frac{x}{\gamma}, \text{N}_{\text{min}}, \text{N}_{\text{max}}) \rceil \times \gamma_b, \frac{\partial \lfloor x \rceil}{\partial x} \triangleq 1, \tag{1}$$

Table 1. Accuracy of the weight-sharing quantization with/without low bit-width for MobileNetv2 (@80 epochs).

	Top-1 Acc. (%) ***w/*** 2bits ($\downarrow$ sampling probability)	Top-1 Acc. (%) ***w/o*** 2bits
6 bit	69.2	70.4
4 bit	68.1	69.1

where $\lfloor \cdot \rceil$ is the rounding-to-nearest function, and γ is the scale factor. The Straight-Through Estimation (STE) is used to pass the gradients for back-propagation. The clip function ensures that the input values fall into the range $[\mathrm{N}_{\min}, \mathrm{N}_{\max}]$ [12, 67]. For ReLU activations, $\mathrm{N}_{\min} = 0$ and $\mathrm{N}_{\max} = 2^b - 1$. For weights, $\mathrm{N}_{\min} = -2^{b-1}$ and $\mathrm{N}_{\max} = 2^{b-1} - 1$. γ_b is the learnable scalar parameter used to adjust the quantization mappings, called the *step-size scale factor*. For a network, each layer has two distinct scale factors in the weights and activations quantizer.

Weight-Sharing for Mixed-Precision Quantization. We consider a model $f(\cdot) = f_{\mathbf{W}_{\mathrm{L}-1}} \circ ... \circ f_{\mathbf{W}_0}(\cdot)$ with L layers, and each layer has N bit-width choices. Let $\mathbf{W} := \{\mathbf{W}_l\}_{l=0}^{L-1}$ be the set of flattened weight tensors of these L layers. Therefore, the corresponding search space $\mathcal{A}$ with N^{2L} mixed-precision quantization policies $\{(b_l^{(w)}, b_l^{(a)})\}_{l=0}^{L-1}$ share the same **latent weights W**. To track the time-prohibitive training costs of traversing the whole search space, Monte-Carlo sampling is used to approximate the expectation term [47, 48, 66]. The overall optimization objective is formulated as follows

$$\begin{aligned} &\arg\min_{\mathbf{W}} \quad \mathbb{E}_{\mathcal{S}\sim\mathcal{A}}\left[\mathcal{L}(f(\mathbf{x}; \mathcal{S}, w^{(\mathcal{S})}), \mathbf{y})\right] \\ &\approx \arg\min_{\mathbf{W}} \frac{1}{K} \sum_{\mathcal{S}_k\sim \mathrm{U}(\mathcal{A})}^{K} \left[\mathcal{L}(f(\mathbf{x}; \mathcal{S}_k, \hat{\mathbf{W}}^{(\mathcal{S}_k)}), \mathbf{y})\right], \end{aligned} \tag{2}$$

where $\hat{\mathbf{W}}^{(\mathcal{S}_k)}$ is the quantized weights of k-th sampled policy $\mathcal{S}_k$ derived from the latent weights $\mathbf{W}$. This weight-sharing of layer l is achieved by

$$\hat{\mathbf{W}}^{(\mathcal{S}_k)} := \{\hat{\mathbf{W}}_l^{(\mathcal{S}_k)}\}_{l=0}^{L-1}, \quad \text{where} \quad \hat{\mathbf{W}}_l^{(\mathcal{S}_k)} = Q_{b_{l,\mathrm{w}}^{(k)} \in \mathcal{S}_k}(\mathbf{W}_l; \gamma) \tag{3}$$

according to Eq. (1), where $b_{l,\mathrm{w}}^{(k)} \in \mathcal{S}_k$ is the bit-width of weight of l-th layer in the policy $\mathcal{S}_k$.

3.2. Interference in Highly Coupled Weight-sharing

While training is possible, there still many challenges exist in weight-sharing in practice. For example, ABN [48] observes a large accuracy gap between the lower bit-widths and higher bit-widths. These works simply bypass this problem and remove the ultra-low bit-width until the training becomes stable, however, doing so does not achieve high compression ratios.

Here, we discuss the bit-width interference caused by highly coupled latent weights $\mathbf{W}$. Suppose we have $K = 2$ sampled policies $\{\mathcal{S}_0, \mathcal{S}_1\}$ at training step t in Eq. (2), and the bit-width of weight is sampled differently, namely $\hat{\mathbf{W}}^{(\mathcal{S}_0)} \neq \hat{\mathbf{W}}^{(\mathcal{S}_1)}$. $\forall b_{l,\mathrm{w}}^{(0)} \in \mathcal{S}_0, \forall b_{l,\mathrm{w}}^{(1)} \in \mathcal{S}_1 : b_{l,\mathrm{w}}^{(0)} < b_{l,\mathrm{w}}^{(1)}$.

Assumption 3.1 (Non-uniform Bit-width Convergence)**.** *Quantized weights $\hat{\boldsymbol{W}}_l^{(\mathcal{S}_1)} = Q_{b_{l,\mathrm{w}}^{(1)} \in \mathcal{S}_1}(\boldsymbol{W}_l; \gamma)$ of bit-width b_k at step t is nearly converged while the $\hat{\boldsymbol{W}}_l^{(\mathcal{S}_0)} = Q_{b_{l,\mathrm{w}}^{(0)} \in \mathcal{S}_0}(\boldsymbol{W}_l; \gamma)$ is not converged properly. The smaller and not fully converged bit-width in $\mathcal{S}_0$ will pose negative impact to the larger but well converged bit-width in $\mathcal{S}_1$.*

The situation in Assump. 3.1 is observed in weight-sharing network when the learning capacity gap between sub-networks is large enough [48, 66]. To further analyze the impact of $\mathcal{S}_0$, we can approximates the loss perturbation with the second-order Taylor expansion:

$$\begin{aligned} \Delta\mathcal{L} &= \sum_{n=1}^{N} \ell(f(\hat{\mathbf{W}}^{(\mathcal{S}_1)} + \Delta\mathbf{W}, \mathrm{x}^n), \mathrm{y}^n) - \sum_{n=1}^{N} \ell(f(\hat{\mathbf{W}}^{(\mathcal{S}_1)}, \mathrm{x}^n), \mathrm{y}^n) \\ &\approx \nabla_{\hat{\mathbf{W}}^{(\mathcal{S}_1)}} \ell(\hat{\mathbf{W}}^{(\mathcal{S}_1)}) \Delta\mathbf{W} + \Delta\mathbf{W}^{\mathsf{T}} \nabla^2_{\hat{\mathbf{W}}^{(\mathcal{S}_1)}} \ell(\hat{\mathbf{W}}^{(\mathcal{S}_1)}) \Delta\mathbf{W}, \end{aligned} \tag{4}$$

where $\ell(\cdot)$ is the cross-entropy loss function, $\Delta\mathbf{W} := \{\Delta\mathbf{W}_l^{(\mathcal{S}_1)}\}_{l=0}^{L-1}$ is the quantization noise introduced by policy $\mathcal{S}_1$ to policy $\mathcal{S}_0$ on each layer. It is noteworthy that the lower the bit-width, the larger the quantization noise introduced [23, 26, 68], caused by the large rounding and clipping error under very limited available discrete values, *e.g.* quantization error for 2 bits is roughly $5\times$ for 3 bits. Therefore, putting small bit-width (*e.g.* 2bit) into the weight-sharing will lead to large loss perturbation $\Delta\mathcal{L}$ and disturb the overall performance eventually (see Tab. 1). Accordingly, one can draw such conclusions in Eq. (4): *(i)* removing the low bit-width is the simplest way to erase the effects of quantization noise but loses the chance to compress the model with high compression ratio and *(ii)* one can track the loss perturbation through $\Delta\mathbf{W}$ to iteratively freeze the most unstable bit-width, which motivates our method described later in Sec. 3.3.

To illustrate the bit-width interference in weight-sharing quantization, we use a 2D regression quantization experiment depicted in Fig. 1. Our optimization objective minimizes the empirical risk [8, 35]:

$$\arg\min_{w} \mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,1)}\left[\|\mathbf{x}w^* - \mathbf{x}Q_b(w, \gamma)\|_2^2\right], \tag{5}$$

where w^* represents the target weight and $\mathbf{x}$ is sampled from a normal distribution. In Fig. 1(a), the single-bit optimization showcases relatively stable quantized weights, occasionally crossing the quantization boundary due to gradient approximation of STE [1]. Comparatively, weight-sharing quantization exhibits more instability and the *weight moves closer to the quantization bound more frequently* (Fig. 1(b)), even with higher variance in gradients as in Fig. 1(c).

This interference extends beyond toy regression to modern neural networks, shown in Fig. 2 and Tab. 1. Furthermore, Fig. 2 demonstrates the distance between quantized weights of 6 and 4 bits in one training epoch. Removing the smallest bit-width (2 bits) notably stabilizes the higher bit-width curve. However, introducing extra small bit-widths induces significant random oscillations, signifying heightened model training instability.

3.3. Dynamic Bit-width Schedule

Eq. (4) reveals the decomposition of overall quantization noise $\Delta\mathbf{W}$ into layer-specific perturbation components, offering a metric to identify unstable layers. Therefore, *dynamically freezing the bit-width causing weight interference ensures proper convergence for remaining bit-widths during training*. However, direct use of Eq. (4) poses computational challenges, particularly in calculating the Hessian and quantization noise terms, prompting us to devise an alternative method.

We approximate layer perturbations by focusing on rounding errors due to their significant impact on overall performance [17, 35]. Rounding errors portray the distance between full-precision weights and their discrete quantization levels, and reach maximums when at the midpoint between two quantization levels (*i.e.*, the *quantization bound* in Fig. 1) because the possible quantization levels change. In other words, the closer to the quantization bounds, the more unstable the weights are, and therefore the unstable weights are more vulnerable to the weight-sharing. Therefore, tracking the round errors provide effective proxies for constructing our bit-width scheduler. For clarity, we first definite the Bit-width Representation Set (BRS) as follows:

Definition 3.1 (Bit-width Representation Set)**.** *For bit-width b under uniform weight quantization, the bit-width representation set $\Phi_b := \gamma \times \{-2^{b-1}, ..., 0, ..., 2^{b-1}-1\}$, representing 2^b decomposed values of discrete quantization levels according to Eq. (1).*

The midpoints between two adjacent elements in a BRS are quantization bounds, where they have a uniform distance γ. Given a pre-defined weight bit-width candidates $B^{(w)}$, we can accumulate bit-specific unstable weights for BRS of each bit-width of each layer's shared weights $\mathbf{W}_l$. Therefore, we calculate the unstable weight criterion $\hat{\Delta}\mathbf{W}^{\text{unstable}}$ by

$$\begin{aligned}\hat{\Delta}\mathbf{W}^{\text{unstable}} &\triangleq \{\hat{\Delta}\mathbf{W}_l^{\text{unstable}}\}_{l=0}^{L-1}, \text{where} \\ \hat{\Delta}\mathbf{W}_l^{\text{unstable}} &= \sum_{b\in B^{(w)}} \frac{1}{2^b}\frac{1}{\|\mathbf{W}_l\|_0} \cdot \\ &\sum_{q_b\in\Phi_b}\sum_{\mathbf{w}_{l,*}\in\mathbf{W}_l} \mathbb{1}_{|\mathbf{w}_{l,*}|\le\gamma\times\left(\frac{1-\epsilon}{2}+\frac{q_b}{\gamma}\right)},\end{aligned} \tag{6}$$

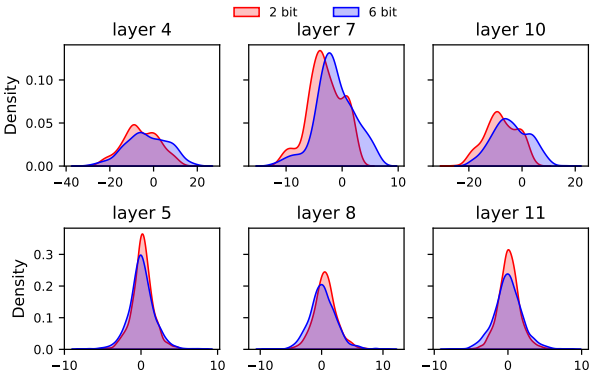

Figure 3. Output density at 2bit and 6bits. Small bit-width shows noteworthy information distortion.

where $\epsilon \in [0, 1]$ is to control the range of weights we care about. After that, we choose the frozen layer set Ω with a Top-$\mathcal{K}$ selector from the weight criterion,

$$\Omega \leftarrow \texttt{TopKToFreeze}(\hat{\Delta}\mathbf{W}^{\text{unstable}}; \mathcal{K}), \tag{7}$$

and the smallest bit-width of selected layers in Ω will be temporarily frozen periodically. In practice, we use a cosine scheduler to gradually decrease the value of $\mathcal{K}$ to guarantee that more unstable low bits will be frozen early to improve the convergence of more high bits.

3.4. Optimization during Dynamic Training

Information Distortion Mitigation. While freezing the bit-width of layers, we observe the outputs of the remaining small bit-widths of layers still exhibit a *information distortion* compared to their high precision counterparts, as shown in Fig. 3. Inspired by the information bottleneck principle [51, 52, 63], we expect if the smallest bit-width is sampled of a layer l, its outputs $\mathbf{O}_l^S$ can preserve the information of its large counterparts $\mathbf{O}_l^H$. However, directly optimizing this mutual information term $I(\mathbf{O}_l^S; \mathbf{O}_l^H)$ is infeasible, so we consider a feature alignment loss function to optimize their rectified Euclidean distance as follows:

$$\begin{aligned}\mathbb{E}\Big[\|\max\{Q, \frac{\mathbf{O}^S-\mu(\mathbf{O}^S)}{\sqrt{\sigma(\mathbf{O}^S)+\zeta}}\eta_{\mathbf{O}^S}+\xi_{\mathbf{O}^S}\} - \\ \max\{Q, \frac{\mathbf{O}^H-\mu(\mathbf{O}^H)}{\sqrt{\sigma(\mathbf{O}^H)+\zeta}}\eta_{\mathbf{O}^H}+\xi_{\mathbf{O}^H}\}\|\Big],\end{aligned} \tag{8}$$

where ζ is a small constant to avoid Divide-by-Zero errors, η and ξ are the learnable parameters for adapting the features, $\mu(\cdot)$ and $\sigma(\cdot)$ return the channel-wise mean and variance of input. Eq. (8) not only scales the features for better optimization but uses a $\max$ operator to avoid needless activations. See Fig. 4 for visualization with proposed Information Distortion Mitigation technique.

Fairness Weight Regularization. Low-bit weights are actually subsets of high-bit weights, when a layer is sampled with different bit-width, low-bit weights will receive additional updates from high-bit weights. In other words, low-bit weights are subjected to very aggressive weight regularization, which exacerbates their underfitting issues [66]. To ensure regularization fairness, we disable weight regularization for *weights of current smallest bit-width* during training.

3.5. Bidirectional Greedy Search

To find the optimal quantization policy $\mathcal{S}^*$, the existing MPQ methods can be formulated to a bi-level optimization problem [47]. In this paper, our well-trained weight-sharing model can serve as a good performance indicator to perform inference-only searching [55, 66]. This simplified procedure motivates us to devise a bidirectional greedy search scheme to determine the per-layer bit-width efficiently.

Consider a mixed-precision quantization policy, $\mathcal{S}^{(t)}$, implemented at step t, with L being the total number of layers. To evolve this policy to $\mathcal{S}^{(t+1)}$, rather than concurrently adjusting the bit-width of most layers (*e.g.*, employing reinforcement learning), a step-by-step approach is taken. Specifically, the bit-width of a single layer is adjusted at a time, either increasing or decreasing by a single bit-width to create a provisional policy, $\mathcal{S}_i^{(t)}$, where $i \in \{0, ..., 2L-1\}$. This method yields a search space of complexity $\mathcal{O}(2L)$ for each iteration. During each iteration, the permanent policy $\mathcal{S}^{(t+1)}$ is chosen in a greedy manner between these $2L$ policies, considering the trade-off between accuracy and efficiency, denoted as $J_i = \bar{\mathcal{L}}_{val}(\hat{\mathbf{W}}^{(\mathcal{S}_i^{(t)})}) + \lambda * \overline{\texttt{BitOps}}(\mathcal{S}_i^{(t)})$ for each layer:

$$\mathcal{S}^{(t+1)} \leftarrow \arg\min_i [\Theta],$$
$$\Theta \triangleq \{J_i | J_i = \bar{\mathcal{L}}_{val}(\hat{\mathbf{W}}^{(\mathcal{S}_i^{(t)})}) + \lambda * \overline{\texttt{BitOps}}(\mathcal{S}_i^{(t)})\}_{i=0}^{2L-1}, \tag{9}$$

where $\bar{\mathcal{L}}$ and $\overline{\texttt{BitOps}}$ are the min-max normalization loss and BitOPs to ensure their values fall into the interval $[0, 1]$, and λ is the hyper-parameter to control the trade-off, respectively. By this means, the solution $\mathcal{S}^*$ is reached when the BitOPs is satisfied at final step T, *i.e.*, $\mathcal{S}^* \leftarrow \mathcal{S}^{(T)}$, if $\texttt{BitOps}(\mathcal{S}^{(T)}) \leq C$.

4. Experiments

In this section, we conduct experiments on three lightweight models (*i.e.*, ResNet18, MobileNetv2, and EfficientNetLite-B0) and three datasets (*i.e.*, ImageNet, Pets, and CIFAR100). Please refer to the *Supplementary Materials* for more detailed experimental setups.

Table 2. Accuracy and efficiency results for ResNet. "Top-1 Acc." represents the Top-1 accuracy of the quantized model and full-precision model. "MP" means mixed-precision quantization. "Retrain Cost" denotes the epochs required to retrain the MPQ policy. "*": reproduces through the vanilla ResNet architecture [19]. The best results are bolded in each metric, the second best results are underlined.

Method	Bit-width (W/A)	Top-1 Acc. (%) ↑	BitOPs (G) ↓	Retrain Cost (Epoch) ↓
Baseline	32 / 32	70.5	-	-
PACT [7]	3 / 3	68.1	23.0	-
LSQ* [12]	3 / 3	69.4	23.0	90
EWGS [28]	3 / 3	69.7	23.0	100
EdMIPS [12]	3_{MP} / 3_{MP}	68.2	-	40
GMPQ* [60]	3_{MP} / 3_{MP}	68.6	22.8	40
DNAS [61]	3_{MP} / 3_{MP}	68.7	24.3	120
FracBits [64]	3_{MP} / 3_{MP}	69.4	22.9	150
LIMPQ [47]	3_{MP} / 3_{MP}	69.7	23.0	90
SEAM [49]	3_{MP} / 3_{MP}	70.0	23.0	90
Ours	2_{MP} / 3_{MP}	67.7	17.3	**0**
Ours	3_{MP} / 3_{MP}	**70.2**	23.3	**0**
PACT [7]	4 / 4	69.2	35.0	-
LSQ* [12]	4 / 4	70.5	35.0	90
EWGS [28]	4 / 4	70.6	35.0	100
MPQCO [6]	4_{MP} / 4_{MP}	69.8	-	30
DNAS [13]	4_{MP} / 4_{MP}	70.6	35.1	120
FracBits [64]	4_{MP} / 4_{MP}	70.6	34.7	150
LIMPQ [47]	4_{MP} / 4_{MP}	70.8	35.0	90
SEAM [49]	4_{MP} / 4_{MP}	70.8	<u>33.7</u>	90
Ours	4_{MP} / 4_{MP}	**71.0**	**31.6**	**0**

4.1. ImageNet Classification

ResNet. PACT demonstrates accuracy with 3-bits for both weights and activations, achieving 68.1%. LSQ reaches 69.4% accuracy but requires 90 retraining epochs. EdMIPS and GMPQ employ MPQ (3MP / 3MP) for 68.2% and 68.6% accuracy but still require considerable retraining costs. DNAS and FracBits adopt longer retraining epochs and yield better accuracy.

When increasing the bit-width to 4-bits, PACT achieves 69.2% accuracy with 35.0G BitOPs, while LSQ reaches 70.5% accuracy. DNAS and FracBits demonstrate a 4-bits MPQ with slightly different results, while LIMPQ and SEAM both achieve the highest accuracy but still need 90 retraining epochs. Notably, our method with varying bit-width configurations (2MP/3MP, 3MP/3MP, and 4MP/4MP). The 4MP/4MP configuration achieves the

Table 3. Accuracy and efficiency results for MobileNetV2. †: QBitOPT uses channel-wise quantization to retain performance.

Method	Bit-width (W/A)	Top-1 Acc. (%) ↑	BitOPs (G) ↓	Retrain Cost (Epoch) ↓
Baseline	32 / 32	72.6	-	-
LSQ [12]	3 / 3	65.2	3.4	90
QBR [17]	3 / 3	67.4	3.4	90
HMQ [15]	2_{MP} / 4_{MP}	64.5	-	50
QBitOPT† [37]	3_{MP} / 3_{MP}	65.7	-	30
NIPQ [44]	3_{MP} / 3_{MP}	62.3	-	43
Ours	3_{MP} / 3_{MP}	**67.8**	3.6	**0**
LSQ [12]	4 / 4	69.5	5.4	90
EWGS [28]	4 / 4	70.3	5.4	100
AdaBits [26]	4 / 4	70.4	5.4	**0**
QBR [17]	4 / 4	70.4	5.4	90
MPDNN [53]	3.75_{MP} / 4_{MP}	69.8	-	50
QBitOPT† [37]	4_{MP} / 4_{MP}	69.7	-	30
NIPQ [44]	4_{MP} / 4_{MP}	69.2	-	43
BayesianBits [54]	4_{MP} / 4_{MP}	69.0	5.9	40
GMPQ [17]	$\sim 4_{MP}$ / 4_{MP}	70.4	7.4	40
HAQ [57]	6_{MP} / 4_{MP}	69.5	8.3	30
Ours	4_{MP} / 4_{MP}	**70.7**	5.5	**0**

highest accuracy in the table at 71.0%, with competitive BitOPs (31.6G) and no retraining cost.

MobileNetV2. QBR demonstrates a competitive Top-1 accuracy of 67.4% with 3/3 bit-width and 3G BitOPs. QBitOPT adopts a performance-friendly channel-wise quantization and achieves 65.7% accuracy in the 3MP/3MP configuration and requires retraining split into 15 + 15 epochs [37], suggesting a more complex process. In the 4/4 bit-width category, QBR stands out with 70.4% accuracy and 5.4G BitOPs, demonstrating efficiency. GMPQ delivers 70.4% accuracy but requires 40 retraining epochs. HAQ achieves 69.5% accuracy but incurs higher BitOPs (8.3G) and demands 30 retraining epochs.

With 3_{MP}/3_{MP} bit-width, our method reaches 67.8% accuracy with 3.6G BitOPs and no retraining. Moreover, in the 4_{MP}/4_{MP} configuration, it excels with a Top-1 accuracy of 70.7% and competitive BitOPs (5.5G), all while eliminating retraining costs.

EfficientNet. LSQ achieves 69.7% accuracy with 4.2G BitOPs and requires 90 retraining epochs. In contrast, our 3MP/3MP method attains 70.4% accuracy with 4.5G BitOPs but eliminates the need for retraining, showcasing improved accuracy at a lower cost. QBitOPT achieves 70.0% accuracy under 3MP/3MP with 30 epochs for retraining. Our method at the same setting achieves 70.4% accuracy without any retraining, highlighting superior performance without complex retraining. While LSQ and NIPQ achieve 72.3% accuracy at 4/4 bit-width, they demand 90 retraining epochs. Our 4MP/4MP method surpasses both, achieving 73.2% accuracy with 6.9G BitOPs and no retraining. Our method consistently achieves comparable or superior accuracy with no retraining costs, demonstrating efficacy and simplicity in EfficientNet quantization.

Table 4. Accuracy and efficiency results for EfficientNetLite-B0. †: QBitOPT uses channel-wise quantization to retain performance.

Method	Bit-width (W/A)	Top-1 Acc. (%) ↑	BitOPs (G) ↓	Retrain Cost (Epoch) ↓
Baseline	32 / 32	75.4	-	-
LSQ [12]	3 / 3	69.7	4.2	90
NIPQ [44]	3_{MP} / 3_{MP}	66.5	-	43
QBitOPT† [37]	3_{MP} / 3_{MP}	70.0	-	30
MPQDNN [53]	3_{MP} / 3_{MP}	68.8	-	50
Ours	3_{MP} / 3_{MP}	**70.4**	4.5	**0**
LSQ [12]	4 / 4	72.3	6.8	90
NIPQ [44]	4_{MP} / 4_{MP}	72.3	-	43
QBitOPT† [37]	4_{MP} / 4_{MP}	**73.3**	-	30
Ours	4_{MP} / 4_{MP}	73.2	6.9	**0**

Table 5. Accuracy and efficiency results for ResNet with knowledge distillation.

Method	Bit-width (W/A)	Top-1 Acc. (%) ↑	BitOPs (G) ↓	Retrain Cost (Epoch) ↓
Baseline	32 / 32	70.5	-	-
GMPQ [60]	3_{MP} / 3_{MP}	69.5	22.8	90
SEAM [49]	3_{MP} / 3_{MP}	70.7	23.0	90
EQNet [62]	3_{MP} / 3_{MP}	69.8	-	**0**
SDQ [23]	3_{MP} / 3	70.2	25.1	90
Ours	3_{MP} / 3_{MP}	**70.9**	23.9	**0**
NIPQ [44]	4_{MP} / 4_{MP}	71.2	34.2	40
SDQ [23]	4_{MP} / 3	**71.7**	33.4	90
Ours	4_{MP} / 4_{MP}	71.6	**31.6**	**0**

4.2. Ablation Study

Efficientness with KD. In comparison to the existing methods in Tab. 5 when knowledge distillation (KD) is enabled with a ResNet101 teacher model, our method exhibits compelling advantages. GMPQ achieves a respectable 69.5% accuracy with 3_{MP} bit-width but requires 90 retraining epochs. Our method surpasses it significantly,

Table 6. Effectiveness of proposed dynamic bit-width schedule scheme and information distortion mitigation (IDM) training technique. To save costs, we train the weight-sharing model 80 epochs.

Dynamic Bit Schedule	IDM Training	4 Bit Top-1 Acc. (%)
✗	✗	68.3
✓	✗	69.1 (**+0.8%**)
✓	✓	69.5 (**+1.2%**)

achieving a 70.9% accuracy without retraining. Similarly, SEAM marginally improves accuracy to 70.7%, but our method still outperforms with 70.9% accuracy and no retraining costs. EQNet stands out with zero retraining epochs but falls significantly short of our method in accuracy (69.8%). SDQ shows varied performance, but our method consistently outperforms it, particularly with 3_{MP} / 3_{MP} and 4_{MP} / 4_{MP} bit-width configurations, achieving higher accuracy and requiring no retraining compared to SDQ's 90 retraining epochs.

Effectiveness of Proposed Techniques. Tab. 6 investigates the impact of a dynamic bit-width schedule and our information distortion mitigation (IDM) training technique on the weight-sharing model. It presents three experimental scenarios: without both dynamic bit scheduling and IDM training resulting in 68.3% Top-1 accuracy, dynamic bit scheduling alone with an improvement to 69.1%, and the combination of both techniques achieving the highest Top-1 accuracy of 69.5%. The results suggest that both dynamic bit scheduling and IDM training contribute positively to the model's performance, and their combination yields the most significant improvement. Moreover, our IDM training technique significantly mitigates information distortion, as shown in Fig. 4.

4.3. Transfer Learning

We transfer the quantized weights for downstream benchmarks to verify the generalization ability of the proposed method. We directly use the pretrained checkpoints on ImageNet and then finetune the classifiers. As shown in Tab. 7, our method achieves the same accuracy as a full-precision model at 4-bits with smaller model complexity, which further confirms the superiority of the proposed method.

5. Conclusion

In this paper, we introduce a novel one-shot training-searching paradigm for mixed-precision model compression. More specifically, traditional approaches focus on bit-width configurations but overlook significant retraining costs. We identified and addressed bit-width interference issues by introducing a dynamic scheduler and an information distortion mitigation technique. Together with an inference-only greedy search scheme, our method can significantly reduce the costs of mixed-precision quantization. Experiments on three commonly used benchmarks across various network architectures validate the effectiveness and efficiency of the proposed method in compressing models. Overall, our method offers a promising solution for deploying compressed models without compromising performance on resource-limited devices.

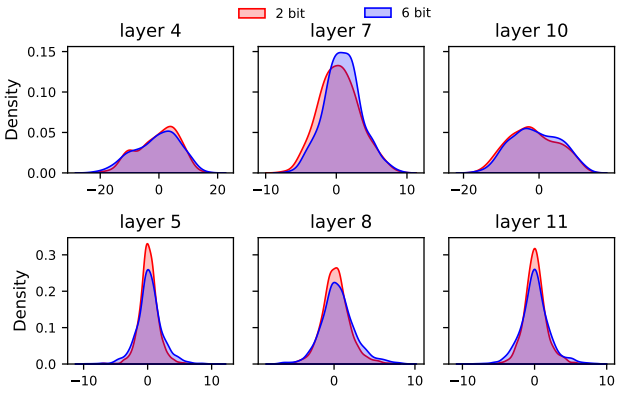

Figure 4. Output density at 2bit and 6bits with our IDM training. Compared with Fig. 3, information distortion of the small bit-widths is significantly mitigated.

Table 7. Performance of transfer learning using the pretrained weights on ImageNet.

Model	Bit-width (W/A)	CIFAR100 [27] Top-1 Acc. (%)	Pets [36] Top-1 Acc. (%)
ResNet18	32 / 32	79.4	88.9
	4_{MP} / 4_{MP}	79.5 (**+0.1%**)	88.7 (-0.2%)
	3_{MP} / 3_{MP}	78.7 (-0.7%)	87.9 (-2.0%)
MobileNetV2	32 / 32	78.9	86.0
	4_{MP} / 4_{MP}	79.0 (**+0.1%**)	86.1 (**+0.1%**)
	3_{MP} / 3_{MP}	78.2 (-1.7%)	84.1 (-1.9%)

Acknowledgment

This work was supported by the National Key Research and Development Program of China No. 2023YFF0905502, National Natural Science Foundation of China (Grant No. 62250008), Beijing National Research Center for Information Science and Technology (BNRist) under Grant No. BNR2023TD03006 and Beijing Key Lab of Networked Multimedia, and Shenzhen Science and Technology Program (Grant No. RCYX20200714114523079 and No. JCYJ20220818101014030).

References

[1] Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville. Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation. *CoRR*, 2013. 2, 4

[2] Adrian Bulat and Georgios Tzimiropoulos. Bit-mixer: Mixed-precision networks with runtime bit-width selection. In *Proc. of ICCV*, 2021. 3

[3] Zhaowei Cai and Nuno Vasconcelos. Rethinking Differentiable Search for Mixed-precision Neural Networks. In *Proc. of CVPR*, 2020. 1, 2, 3

[4] Cheng Chen, Yichun Yin, Lifeng Shang, Xin Jiang, Yujia Qin, Fengyu Wang, Zhi Wang, Xiao Chen, Zhiyuan Liu, and Qun Liu. bert2bert: Towards reusable pretrained language models. In *Proc. of ACL*, 2022. 1

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 2017. 1

[6] Weihan Chen, Peisong Wang, and Jian Cheng. Towards Mixed-precision Quantization of Neural Networks via Constrained Optimization. In *Proc. of ICCV*, 2021. 1, 3, 6

[7] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. PACT: Parameterized Clipping Activation for Quantized Neural Networks. *CoRR*, 2018. 2, 6

[8] Alexandre Défossez, Yossi Adi, and Gabriel Synnaeve. Differentiable model compression via pseudo quantization noise. *Transactions on Machine Learning Research*, 2022. 4

[9] Zhen Dong, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. HAWQ: Hessian AWare Quantization of Neural Networks With Mixed-precision. In *Proc. of ICCV*, 2019. 1, 3

[10] 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 *Proc. of NeurIPS*, 2020. 3

[11] Ahmed T. Elthakeb, Prannoy Pilligundla, Fatemehsadat Mireshghallah, Amir Yazdanbakhsh, and Hadi Esmaeilzadeh. ReLeQ : A Reinforcement Learning Approach for Automatic Deep Quantization of Neural Networks. *IEEE Micro*, 2020. 1, 2

[12] Steven K. Esser, Jeffrey L. McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S. Modha. Learned Step Size quantization. In *Proc. of ICLR*, 2020. 1, 2, 4, 6, 7

[13] 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 *Proc. of ICCV*, 2019. 6

[14] Zichao Guo, Xiangyu Zhang, Haoyuan Mu, Wen Heng, Zechun Liu, Yichen Wei, and Jian Sun. Single Path One-shot Neural Architecture Search with Uniform Sampling. In *Proc. of ECCV*, 2020. 2, 3

[15] Hai Victor Habi, Roy H. Jennings, and Arnon Netzer. HMQ: Hardware Friendly Mixed Precision Quantization Block for CNNs. In *Proc. of ECCV*, 2020. 1, 7

[16] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015. 1

[17] Tiantian Han, Dong Li, Ji Liu, Lu Tian, and Yi Shan. Improving low-precision network quantization via bin regularization. In *Proc. of ICCV*, 2021. 5, 7

[18] Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 3

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proc. of CVPR*, 2016. 1, 6

[20] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proc. of ICCV*, 2017. 1

[21] Andrew Howard, Ruoming Pang, Hartwig Adam, Quoc V. Le, Mark Sandler, Bo Chen, Weijun Wang, Liang-Chieh Chen, Mingxing Tan, Grace Chu, Vijay Vasudevan, and Yukun Zhu. Searching for MobileNetV3. In *Proc. of ICCV*, 2019. 1, 3

[22] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *CoRR*, 2017. 1

[23] Xijie Huang, Zhiqiang Shen, Shichao Li, Zechun Liu, Xianghong Hu, Jeffry Wicaksana, Eric P. Xing, and Kwang-Ting Cheng. SDQ: Stochastic Differentiable Quantization with Mixed Precision. In *Proc. of ICML*, 2022. 1, 4, 7

[24] Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Dehao Chen, Mia Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V Le, Yonghui Wu, et al. Gpipe: Efficient training of giant neural networks using pipeline parallelism. *Proc. of NeurIPS*, 2019. 1

[25] Itay Hubara, Yury Nahshan, Yair Hanani, Ron Banner, and Daniel Soudry. Accurate post training quantization with small calibration sets. In *Proc. of ICML*, 2021. 2

[26] Qing Jin, Linjie Yang, and Zhenyu Liao. Adabits: Neural network quantization with adaptive bit-widths. In *Proc. of CVPR*, 2020. 3, 4, 7

[27] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 8

[28] Junghyup Lee, Dohyung Kim, and Bumsub Ham. Network quantization with element-wise gradient scaling. In *Proc. of CVPR*, 2021. 6, 7

[29] Ji Lin, Yongming Rao, Jiwen Lu, and Jie Zhou. Runtime neural pruning. *Proc. of NeurIPS*, 2017. 3

[30] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018. 3

[31] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the Value of Network Pruning. In *Proc. of ICLR*, 2019. 1

[32] Rongwei Lu, Yutong Jiang, Yinan Mao, Chen Tang, Bin Chen, Laizhong Cui, and Zhi Wang. Dagc: Data-volume-aware adaptive sparsification gradient compression for dis-

tributed machine learning in mobile computing. *arXiv preprint arXiv:2311.07324*, 2023. 1

[33] Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In *Proc. of CVPR*, 2019. 1

[34] Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. In *Proc. of ICML*, 2020. 2

[35] Markus Nagel, Marios Fournarakis, Yelysei Bondarenko, and Tijmen Blankevoort. Overcoming oscillations in quantization-aware training. In *Proc. of ICML*, 2022. 4, 5

[36] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *Proc. of CVPR*, 2012. 8

[37] Jorn Peters, Marios Fournarakis, Markus Nagel, Mart van Baalen, and Tijmen Blankevoort. Qbitopt: Fast and accurate bitwidth reallocation during training. In *Proc. of ICCV*, 2023. 7

[38] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In *Proc. of ICML*, 2018. 3

[39] Antonio Polino, Razvan Pascanu, and Dan Alistarh. Model compression via distillation and quantization. In *Proc. of ICLR*, 2018. 1

[40] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 1

[41] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Proc. of NeurIPS*, 2015. 1

[42] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In *Proc. of CVPR*, 2018. 1

[43] Jianghao Shen, Yue Wang, Pengfei Xu, Yonggan Fu, Zhangyang Wang, and Yingyan Lin. Fractional skipping: Towards finer-grained dynamic cnn inference. In *Proc. of AAAI*, 2020. 3

[44] Juncheol Shin, Junhyuk So, Sein Park, Seungyeop Kang, Sungjoo Yoo, and Eunhyeok Park. Nipq: Noise proxy-based integrated pseudo-quantization. In *Proc. of CVPR*, 2023. 7

[45] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *Proc. of ICML*, 2019. 3

[46] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *Proc. of CVPR*, 2020. 1

[47] Chen Tang, Kai Ouyang, Zhi Wang, Yifei Zhu, Wen Ji, Yaowei Wang, and Wenwu Zhu. Mixed-precision Neural Network Quantization via Learned Layer-wise Importance. In *Proc. of ECCV*, 2022. 1, 3, 4, 6

[48] Chen Tang, Haoyu Zhai, Kai Ouyang, Zhi Wang, Yifei Zhu, and Wenwu Zhu. Arbitrary bit-width network: A joint layer-wise quantization and adaptive inference approach. In *Proc. of ACM MM*, 2022. 3, 4

[49] Chen Tang, Kai Ouyang, Zenghao Chai, Yunpeng Bai, Yuan Meng, Zhi Wang, and Wenwu Zhu. Seam: Searching transferable mixed-precision quantization policy through large margin regularization. In *Proc. of ACM MM*, 2023. 2, 6, 7

[50] Chen Tang, Li Lyna Zhang, Huiqiang Jiang, Jiahang Xu, Ting Cao, Quanlu Zhang, Yuqing Yang, Zhi Wang, and Mao Yang. Elasticvit: Conflict-aware supernet training for deploying fast vision transformer on diverse mobile devices. In *Proc. of ICCV*, 2023. 1, 3

[51] Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 ieee information theory workshop (itw)*, 2015. 5

[52] Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000. 5

[53] 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. *arXiv preprint arXiv:1905.11452*, 2019. 1, 7

[54] Mart Van Baalen, Christos Louizos, Markus Nagel, Rana Ali Amjad, Ying Wang, Tijmen Blankevoort, and Max Welling. Bayesian bits: Unifying quantization and pruning. *Proc. of NeurIPS*, 2020. 7

[55] Dilin Wang, Meng Li, Chengyue Gong, and Vikas Chandra. Attentivenas: Improving neural architecture search via attentive sampling. In *Proc. of CVPR*, 2021. 6

[56] Jue Wang, Ke Chen, Gang Chen, Lidan Shou, and Julian McAuley. Skipbert: Efficient inference with shallow layer skipping. In *Proc. of ACL*, 2022. 3

[57] Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. HAQ: Hardware-aware Automated Quantization With Mixed Precision. In *Proc. of CVPR*, 2019. 1, 2, 7

[58] Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. Skipnet: Learning dynamic routing in convolutional networks. In *Proc. of ECCV*, 2018. 3

[59] Yulong Wang, Xiaolu Zhang, Xiaolin Hu, Bo Zhang, and Hang Su. Dynamic network pruning with interpretable layerwise channel selection. In *Proc. of AAAI*, 2020. 3

[60] Ziwei Wang, Han Xiao, Jiwen Lu, and Jie Zhou. Generalizable Mixed-precision Quantization via Attribution Rank Preservation. In *Proc. of ICCV*, 2021. 1, 2, 3, 6, 7

[61] Bichen Wu, Yanghan Wang, Peizhao Zhang, Yuandong Tian, Peter Vajda, and Kurt Keutzer. Mixed Precision Quantization of ConvNets via Differentiable Neural Architecture Search. *CoRR*, 2018. 6

[62] Ke Xu, Lei Han, Ye Tian, Shangshang Yang, and Xingyi Zhang. Eq-net: Elastic quantization neural networks. In *Proc. of ICCV*, 2023. 3, 7

[63] Sheng Xu, Yanjing Li, Mingbao Lin, Peng Gao, Guodong Guo, Jinhu Lü, and Baochang Zhang. Q-detr: An efficient low-bit quantized detection transformer. In *Proc. of CVPR*, 2023. 5

[64] Linjie Yang and Qing Jin. FracBits: Mixed Precision Quantization via Fractional Bit-widths. In *Proc. of AAAI*, 2021. 6

[65] Haibao Yu, Qi Han, Jianbo Li, Jianping Shi, Guangliang Cheng, and Bin Fan. Search What You Want: Barrier Panelty NAS for Mixed Precision Quantization. In *Proc. of ECCV*, 2020. 2

[66] Jiahui Yu, Pengchong Jin, Hanxiao Liu, Gabriel Bender, Pieter-Jan Kindermans, Mingxing Tan, Thomas Huang, Xiaodan Song, Ruoming Pang, and Quoc Le. Bignas: Scaling up neural architecture search with big single-stage models. In *Proc. of ECCV*, 2020. 4, 6

[67] Shuchang Zhou, Zekun Ni, Xinyu Zhou, He Wen, Yuxin Wu, and Yuheng Zou. DoReFa-Net: Training Low Bitwidth Convolutional Neural Networks with Low Bitwidth Gradients. *CoRR*, 2016. 1, 2, 4

[68] Yiren Zhou, Seyed-Mohsen Moosavi-Dezfooli, Ngai-Man Cheung, and Pascal Frossard. Adaptive quantization for deep neural network. In *Proc. of AAAI*, 2018. 4

[69] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016. 3