Distribution-aware Adaptive Multi-bit Quantization

Sijie Zhao[1,2], Tao Yue[1], Xuemei Hu[1]
[1]School of Electronic Science and Engineering, Nanjing University, Nanjing, China
[2]Shenzhen Institute of Future Media Technology, Shenzhen 518071, China
sjzhao@smail.nju.edu.cn, yuetao@nju.edu.cn, xuemeihu@nju.edu.cn

Abstract

In this paper, we explore the compression of deep neural networks by quantizing the weights and activations into multi-bit binary networks (MBNs). A distribution-aware multi-bit quantization (DMBQ) method that incorporates the distribution prior into the optimization of quantization is proposed. Instead of solving the optimization in each iteration, DMBQ search the optimal quantization scheme over the distribution space beforehand, and select the quantization scheme during training using a fast lookup table based strategy. Based upon DMBQ, we further propose loss-guided bit-width allocation (LBA) to adaptively quantize and even prune the neural network. The first-order Taylor expansion is applied to build a metric for evaluating the loss sensitivity of the quantization of each channel, and automatically adjust the bit-width of weights and activations channel-wisely. We extend our method to image classification tasks and experimental results show that our method not only outperforms state-of-the-art quantized networks in terms of accuracy but also is more efficient in terms of training time compared with state-of-the-art MBNs, even for the extremely low bit width (below 1-bit) quantization cases.

1. Introduction

In the past decades, deep neural networks have achieved great successes in many fields [15, 21, 37, 7]. However, the requirements of huge memory footprint and computational resources impede the practical deployment of network based algorithms on resource-constraint devices, e.g., mobile phones, smart dresses and autopilot vehicles.

To facilitate deployability, network quantization methods quantize the weights and activations into low precision, greatly compressing the model size and reducing the required computational resources in inference [19, 39, 13, 5, 3]. These network quantization methods need to solve two main problems: (1) How to quantize the weights and activations with lower precision and higher accuracy? (2) How to allocate the bit-width to quantize different parts of weights and activations for optimal performance?

As for the former problem, three main types of quantization schemes are proposed, including fixed-point [19, 13], power of two [24, 38, 18] and binary/ternary [6, 16, 40, 28, 20, 26] quantization. Recently, the multi-bit quantization (MBQ) [20] that quantize weights and activations into the combination of multiple binary bases is proposed, which could achieve better performance and require less computational resources. Based upon MBQ strategy, various methods [20, 36, 32, 27] have been proposed to optimize the quantization problem in the training process to improve the accuracy of MBQ and have made significant improvements. However, how to derive the optimal quantization schemes, i.e., the multi-bit binary filters and the corresponding coordinates, efficiently during the iterations of training-aware quantization remains a knotty problem [27], suffering from the difficulty of the integer optimization (theoretically NP-hard). In this paper, inspired by Banner *et al*. [1] that the distribution can be utilized in the quantization, we explore the distribution of weights and dedicate to find the optimal quantization scheme of MBQ under the distribution assumption. By minimizing the expected mean square error , a lookup table based strategy is proposed to optimize the quantization schemes with very few computational cost during the iterations.

As for the latter one, except the globally unified bit-width allocation, mixed precision quantization which allocates different bit-width to different parts of the network, either layer-wise [9, 8, 10, 31] or channel-wise [22, 17, 5], is proposed to reduce the degradation of quantization, especially for low-bit cases. However, the bit-width allocation is quite challenging. Existing methods, e.g., deep reinforcement learning [22, 10, 31], Hessian information [9, 8], and pruning-based optimization [27], require considerable computation cost, hindering their application in practice. In this paper, through modeling the quantization effect upon the loss of network with Taylor expansion, we formulate a metric to evaluate the quantization sensitivity of weights, i.e., the loss variation of the network with the quantization of weights. Since only gradients of quantized weights are required, which could be directly obtained from the back-

ward propagation, the metric can be easily computed, and thus we can adaptively adjust the quantization bit-width of weights and activations in the training process without too much computational load.

In all, we make the following contributions:

- We introduce a distribution-aware multi-bit quantization (DMBQ) method for efficient and optimal MBQ quantization.
- We propose a first-order Taylor expansion based metric for evaluating the loss-sensitivity of the quantized weights and activations and introduce a loss-guided bit-width allocation (LBA) method.
- We demonstrate the effectiveness and efficiency of the proposed method through extensive comparisons with the state of the art methods.

2. Related Work

Multi-bit Quantization MBQ attracts much attentions for its powerful representation capability and high efficiency for inference. But due to a NP-hard integer optimization problem is involved, most MBQ suffering from the optimization difficulty in practice. Lin *et al*. [20] propose ABC-Net which obtain quantization scheme for weights by applying the least square optimization during each iteration. Xu *et al*. [32] propose a multi-bit alternating quantization scheme for LSTM and GRU networks, and binary search tree are employed to optimize binary bases efficiently. Zhang *et al*. [36] and Qu *et al*. [27] propose to optimize the binary bases and coordinates alternatively by minimizing construction error and quantization-induced loss respectively. All these methods suffer from the problem that there is no guarantee to find the optimal quantization scheme and heavy computational load is involved. In this paper, we propose a distribution-aware MBQ method that take the distribution prior of weights into the optimization of quantization. With a certain distribution assumption, the optimal quantization scheme can be searched over the distribution space beforehand in a brute-force way and the quantization scheme can be selected during the training process with a fast lookup table based strategy.

Mixed-precision Optimization Previous works [35, 5, 22] have observed that various weight layers/kernels exhibit different variances and hence contain different redundancy. Based on this observation, layer-wise quantization [31, 9, 8] or kernel-wise quantization [22, 17, 5] are proposed. Dong *et al*. [9, 8] propose to quantize weights with layer-wise mixed-precision using Hessian information. Power iteration is adopted to compute the top Hessian eigenvalue, which is used to set the relative bit-width among layers. Deep reinforcement learning (DRL) based methods [31, 22] are proposed to automatically select the layer-wise/kernel-wise bit-width of weights. Ma *et al*. [23] propose to adjust the hyper-parameter of network compression using Bayesian optimization (BO). Qu *et al*. [27] propose to optimize the bit-width allocation with a pruning-based optimization. High accuracy quantization could be achieved with these methods, while high computational burden and memory consumption are required, preventing them from practical applications. In this paper, through modeling the quantization effect upon loss with Taylor expansion, we propose a loss-sensitivity metric to guide the bit-width allocation of weights and activations in the training. Since only the gradients of quantized weights and activations are required, which could be directly obtained from backward propagation, we can realize a highly efficient mixed-precision optimization, in terms of both computation and memory.

In all, we propose a distribution-aware adaptive MBQ method that could realize both high accuracy and efficiency.

3. Distribution-aware Adaptive MBQ

The proposed method gradually quantizes the neural network to an expected average bit-width in the training process and the overview of the proposed quantization method is shown in Fig. 1 and Alg. 1. Specifically, during each training epoch, according to the allocated bit-width from LBA, the weights are quantized with the optimal MBQ scheme selected by DMBQ based on a fast lookup table. The activations are quantized with uniform MBQ method. Through forward propagating the neural network with the quantized weights and activations, the loss of the training network is calculated and the weights are updated with back propagation. The loss sensitivity with respect to each channel of weights and activations is calculated with the gradients and accumulated along the iterations. At the end of each epoch, with the accumulated loss sensitivity of the whole epoch, the bit-width for different channels of weights and activations are adjusted with LBA. The quantization optimization is stopped until the target average bit-width of weights and activations are reached. The details of DMBQ and LBA are introduced in this section.

3.1. Distribution-aware MBQ

Given a full precision neural network with tensor-valued weights $\boldsymbol{W}$ with dimension of l, e.g. $\boldsymbol{W} \in \mathbb{R}^l$. MBQ methods quantize $\boldsymbol{W}$ into a linear combination of M binary filters $\boldsymbol{B}_1, \boldsymbol{B}_2, \ldots, \boldsymbol{B}_M \in \{-1, +1\}^l$, i.e. $\boldsymbol{W} \approx \hat{\boldsymbol{W}} = \sum_{k=1}^{M} \alpha_k \boldsymbol{B}_k$, where M is the bit-width of quantization, and $\alpha_1, \alpha_2, ..., \alpha_M$ are the corresponding float-point coordinates.

The goal of MBQ methods is to find the optimal quantization schemes $\boldsymbol{\alpha}$ ($\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_M]$) and $\boldsymbol{B}$ ($\boldsymbol{B} = [\boldsymbol{B}_1, \boldsymbol{B}_2, \ldots, \boldsymbol{B}_M]$) to minimize the quantization error, i.e.

$$\min_{\boldsymbol{\alpha}, \boldsymbol{B}} \| \boldsymbol{W} - \sum_{k=1}^{M} \alpha_k \boldsymbol{B}_k \|^2. \tag{1}$$

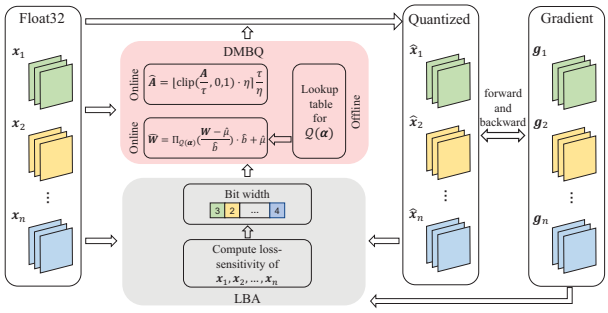

Figure 1. An overview of our proposed quantization method. We construct a lookup table beforehand which is used to efficiently quantize weights into MBQ in forward propagation. After backward propagation, the channel-wise loss-sensitivities are accumulated along with iterations and are used to adjust the bit-width at the end of each epoch.

Through quantizing both weights and activations into the optimal multi-bit form, the inference of the neural network could be realized in a highly efficient way compared with the floating point calculation.

Since searching for the optimal MBQ is NP-hard [27] and the global optimum can only be found through brute force searching, which is computationally intractable. Existing MBQ methods either propose to heuristically fix $\boldsymbol{B}$ and solve $\boldsymbol{\alpha}$ [20] or alternatively optimize $\boldsymbol{\alpha}$ and $\boldsymbol{B}$ during the training process [32, 36, 27], suffering from high computational cost and no guarantee to obtain global optimum in practice. In this paper, we denote the quantization as,

$$\hat{\boldsymbol{W}} = \Pi_{\mathcal{Q}(\boldsymbol{\alpha})}(\boldsymbol{W}), \tag{2}$$

where each element in $\boldsymbol{W}$ is projected by $\Pi_{\mathcal{Q}(\boldsymbol{\alpha})}(\cdot)$ onto a set of quantization levels $\mathcal{Q}(\boldsymbol{\alpha}) = \{\sum_{k=1}^{M} \alpha_k \beta_k\}, s.t.\ \beta_k \in \{-1, +1\}$. We denote $q_i(\boldsymbol{\alpha})$ as the i-th value in $\mathcal{Q}(\boldsymbol{\alpha})$ which is sorted in ascending order and each $q_i(\boldsymbol{\alpha})$ corresponds to a binary representation $\boldsymbol{\beta}_i = [\beta_1, \beta_2, ..., \beta_M]$. Once the optimal $\boldsymbol{\alpha}$ is obtained, $\hat{\boldsymbol{W}}$ can be derived by rounding $\boldsymbol{W}$ to the nearest $q_i(\boldsymbol{\alpha})$. Denoting the i-th rounding edge $s_i(\boldsymbol{\alpha})$ as,

$$s_i(\boldsymbol{\alpha}) = \begin{cases} -\infty & \text{if} \quad i = 1, \\ \frac{q_i(\boldsymbol{\alpha}) + q_{i+1}(\boldsymbol{\alpha})}{2} & \text{if} \quad 2 \le i \le 2^M, \\ +\infty & \text{if} \quad i = 2^M + 1. \end{cases} \tag{3}$$

Each parameter of $\boldsymbol{W}$ in $[s_i(\boldsymbol{\alpha}), s_{i+1}(\boldsymbol{\alpha})]$ will be quantized in to $q_i(\boldsymbol{\alpha})$. Since each $q_i(\boldsymbol{\alpha})$ corresponds to a binary representation $\boldsymbol{\beta}_i$, we can easily get binary filter $\boldsymbol{B}$ after quantization with the optimal $\boldsymbol{\alpha}$. Thus the quantization optimization problem (Eq. (1)) is then turned to how to efficiently find the optimal $\boldsymbol{\alpha}$.

Inspired by Banner *et al*. [1] that the distribution of weights can be modeled by laplace approximately, we explore the distribution of weights and dedicate to find the optimal quantization scheme of MBQ under the distribution assumption.

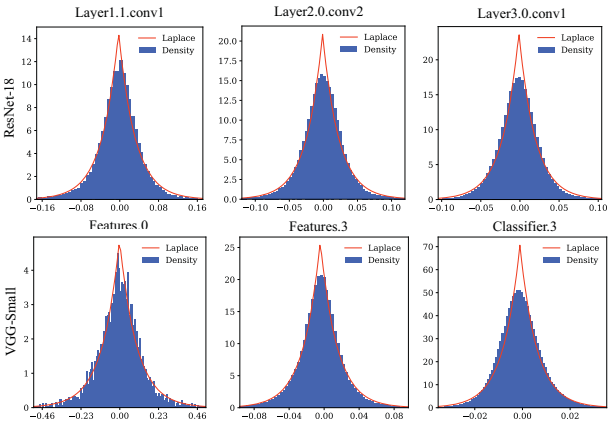

Figure 2. The distribution of layers in ResNet-18 [12] (top row) and VGG-small [30] (bottom row). It is obvious that the weights distribution can be approximated well using Laplace distribution.

To demonstrate the distribution assumption intuitively, Fig. 2 shows the weight histogram of several layers in two typical neural network, i.e., ResNet-18 [12] and VGG-small [30], as well as the fitted probability density function of Laplace distribution. We can see that the neural network weights can be approximated very well by using a Laplace distribution. Therefore, we consider a random variable X that subjects to Laplace distribution, i.e. $\mathcal{L}(\mu, b)$, where $\mu = \mathbb{E}(X)$ is the mean value and $b = \mathbb{E}(|X - \mu|)$ denotes the mean absolute deviation (MAD) of the Laplace distribution. Here, we replace Eq. (1) with the expectation of quantization error under the Laplace distribution assumption as the objective,

$$\min_{\boldsymbol{\alpha}} \mathbb{E}\big((X - \hat{X})^2\big) = \min_{\boldsymbol{\alpha}} \sum_{i=1}^{2^M} \int_{s_i(\boldsymbol{\alpha})}^{s_{i+1}(\boldsymbol{\alpha})} f(x)(x - q_i(\boldsymbol{\alpha}))^2 dx, \tag{4}$$

where $f(x)$ is the probability density function and $\hat{X}$ is the quantization of X. Since we can always normalize $\boldsymbol{W}$ during quantization as,

$$\hat{\boldsymbol{W}} = \Pi_{\mathcal{Q}(\boldsymbol{\alpha})}\big(\frac{\boldsymbol{W} - \hat{\mu}}{\hat{b}}\big) \cdot \hat{b} + \hat{\mu}, \tag{5}$$

where $\hat{\mu} = \mathbb{E}(\boldsymbol{W})$ and $\hat{b} = \mathbb{E}(|\boldsymbol{W} - \hat{\mu}|)$, we can only consider the quantization of standard case of Laplace distribution, i.e. $\mathcal{L}(0, 1)$. Substituting with $f(x) = \frac{1}{2}e^{-|x|}$ into Eq. (4) and we can get,

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \sum_{i=1}^{2^{M-1}} 2\Big(\Psi\big(q_i(\boldsymbol{\alpha}), s_{i+1}(\boldsymbol{\alpha})\big) - \Psi\big(q_i(\boldsymbol{\alpha}), s_i(\boldsymbol{\alpha})\big)\Big), \\ &\Psi(q, s) = \frac{e^s}{2}\big((s - q)^2 - 2(s - q) + 2\big), \end{aligned} \tag{6}$$

where $\Psi(q, s)$ is the primitive function of $f(x)(x - q)^2$ in the negative x-axis. However, $q_i(\boldsymbol{\alpha})$ and $s_i(\boldsymbol{\alpha})$ are not con-

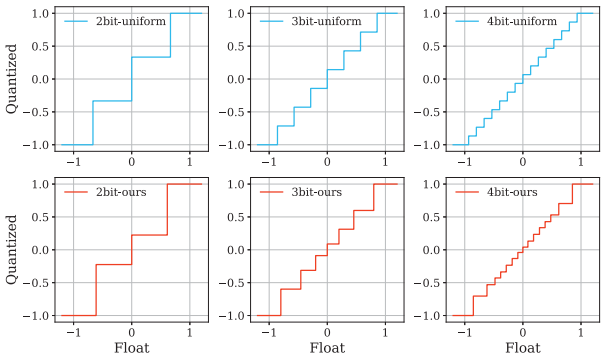

Figure 3. The quantization function of 2-bit, 3-bit, 4-bit using uniform quantization (top row) and DMBQ (bottom row). DMBQ has finer resolution near the origin thus can be more suitable for high quantization precision considering the distribution of weights.

tinuous w.r.t $\boldsymbol{\alpha}$, thus we cannot directly getting the closed-form solution for Eq. (6). As an alternative, we adopt brute force searching algorithm to find the optimal $\boldsymbol{\alpha}$.

With the brute force searching, we can get the optimal $\boldsymbol{\alpha}^{\text{best}}$ = [1.0], [1.009,1.591], [0.832,1.514,1.897], [0.838,1.324,1.619,1.879] for M = 1, 2, 3, 4 respectively. Note that here we do not take the cases where $M > 4$ into consideration, because 4-bit is well enough to represent the weights in practice. The experiments show that the accuracy of the network only degrades slightly compared to the full precision (FP) model after quantizing weights to 4-bit (refer to details in Sec. 4.4). We use $\boldsymbol{\alpha}^{\text{best}}$ to construct a lookup table for $\mathcal{Q}(\boldsymbol{\alpha})$ w.r.t bit-width for once. In Fig. 3, we plot the corresponding quantization function (normalized to [-1, 1]) of the proposed distribution-aware quantization and compared it with the uniform quantization cases. As shown, the optimized quantization points became denser when the distribution probability is higher, which is reasonable in reducing the quantization error.

In practice, we first calculate the statistical value $\hat{\mu}$ and $\hat{b}$ channel-wisely to normalize the weights. With the targeting bit-width to quantize, we could get the corresponding optimal $\mathcal{Q}(\boldsymbol{\alpha})$ by the constructed lookup table. Then weights can be quantized with Eq. (5), which requires no optimization during training and is much more efficient compared with existing optimization based multi-bit quantization methods [20, 36, 32, 27].

It is worth *noting* that the proposed DMBQ method can be easily migrated to other distribution types, like Gaussian, which makes it a general strategy for quantizing the weights following any specific distributions to MBQ form. For details of the Gaussian case, please refer to the supplementary materials.

Activation Quantization In order to take advantage of bitwise operations for speedup, the activations also need to be quantized in multi-bit binary form. However, the distribution of activations is hard to model because of the rescaling and translation introduced by batch normalization and the clipping effect introduced by ReLU function. Therefore, we employ uniform quantization and convert it to the MBQ form for the propose of efficient training and inference.

Our quantization for activations mainly follows the method in [18], while being adapted to MBQ form for compatible with the proposed DMBQ method. Specifically, we divide the floating-point data by τ before clipping with the fixed interval $[0, 1]$ and multiply the quantized data by τ after rounding. τ is viewed as clipping value and updated by the gradient which is calculated by the automatic derivative mechanism. Here we denote the full precision activations as $A \in \mathbb{R}^l$. The quantization in the forward pass is,

$$\hat{\boldsymbol{A}} = \lfloor \text{clip}(\frac{\boldsymbol{A}}{\tau}, 0, 1) \cdot \eta \rceil \cdot \frac{\tau}{\eta}, \tag{7}$$

where $\eta = 2^N - 1$, $\text{clip}(x, r_1, r_2)$ returns r_1 for values below r_1, r_2 for values above r_2, and the values itself for values in the range of r_1 to r_2. $\lfloor x \rceil$ round x to the nearest integer, and $\hat{\boldsymbol{A}}$ is the quantized activations. We define $\boldsymbol{A}_z = \lfloor \text{clip}(\frac{\boldsymbol{A}}{\tau}, 0, 1) \cdot \eta \rceil$ which are integers in $[0, 2^N - 1]$ and $\boldsymbol{C}_1, \boldsymbol{C}_2, \ldots, \boldsymbol{C}_N \in \{0, 1\}^l$ as its binary filters. Thus $\boldsymbol{A}_z = \sum_{j=1}^{N} 2^{j-1} \boldsymbol{C}_j$, and the coordinates of $\boldsymbol{C}_i$ can be calculated as $\gamma_j = \frac{2^{j-1}\tau}{\eta}$ $(j = 1, 2, ..., N)$. The quantized activations can then be denoted as $\hat{\boldsymbol{A}} = \sum_{j=1}^{N} \gamma_j \boldsymbol{C}_j$ in the MBQ form.

After quantizing both weights and activations, the convolution can be calculated through

$$\begin{aligned} \text{conv}(\boldsymbol{A}, \boldsymbol{W}) &\approx \text{conv}(\sum_{j=1}^{N} \gamma_j \boldsymbol{C}_j, \sum_{k=1}^{M} \alpha_k \boldsymbol{B}_k) \\ &= \sum_{j=1}^{N} \sum_{k=1}^{M} \gamma_j \alpha_k \text{conv}(\boldsymbol{C}_j, \boldsymbol{B}_k). \end{aligned} \tag{8}$$

Thus the standard convolution in neural network can be computed by binary convolutions, which can greatly reduce the computational complexity and accelerate the inference.

3.2. Loss-guided Bit-width Allocation

Quantizing the whole network into the same bit-width is not the optimal strategy, since different part of network contains different degree of redundancy. In these paper, we introduce LBA to adaptively adjust the bit-width in channel-wise manner, taking the influence of network loss into account. We model the quantization process as disturbance and use Taylor expansion to derive the loss sensitivity introduced by quantization, i.e.

$$f(\boldsymbol{x}) \approx f(\hat{\boldsymbol{x}}) + (\boldsymbol{x} - \hat{\boldsymbol{x}})^T \boldsymbol{g}(\hat{\boldsymbol{x}}), \tag{9}$$

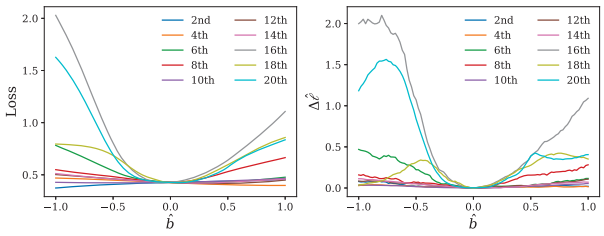

Figure 4. Left: the loss landscape for different layers of ResNet-20 on CIFAR-10 dataset. The landscape is plotted by disturbing weights along the Laplace MAD $\hat{b}$. Right: the corresponding $\Delta\ell'$ landscape of disturbed weights.

where $\boldsymbol{x}$ is the vector of channel-wise weights or hidden activations and is quantized to $\hat{\boldsymbol{x}}$, $f(\boldsymbol{x})$ represents the influence of $\boldsymbol{x}$ on the loss, and $\boldsymbol{g}(\hat{\boldsymbol{x}})$ is the gradient/derivative of $f(\cdot)$ at $\hat{\boldsymbol{x}}$. We make a simple transformation to Eq. (9), i.e.

$$\Delta\ell = |f(\boldsymbol{x}) - f(\hat{\boldsymbol{x}})| \approx |(\boldsymbol{x} - \hat{\boldsymbol{x}})^T \boldsymbol{g}(\hat{\boldsymbol{x}})|. \tag{10}$$

Eq. (10) shows the influential magnitude of the weights/activations before and after the quantization on the loss ℓ. Since the quantization is not differentiable, we adopt straight through estimator (STE) which is widely used in previous works [28, 34, 20]. $\boldsymbol{x}$ is replaced by $\hat{\boldsymbol{x}}$ in forward pass and the gradient $\boldsymbol{g}(\hat{\boldsymbol{x}})$ can be easily derived from backward propagation. The lower the value of $\Delta\ell$, the less sensitive the loss is to the small disturbance of $\boldsymbol{x}$, which indicates that less bit-width can be used to represent $\boldsymbol{x}$ for smaller quantization error. In addition, we propose to normalize $\Delta\ell$ by the number of elements in $\boldsymbol{x}$ to avoid the error introduced by the difference of elements number of different channels, i.e.

$$\Delta\ell' = \frac{|(\boldsymbol{x} - \hat{\boldsymbol{x}})^T \boldsymbol{g}(\hat{\boldsymbol{x}})|}{n}, \tag{11}$$

where n denotes the number of elements in $\boldsymbol{x}$. We denote $\Delta\ell'$ as the sensitivity of different channels, and use it as an indicator to allocate the bit-width.

As shown in Fig. 4, for each layer, we add a gradually changing variable to weights while keep other layers and input the same, and then plot the landscape of loss and $\Delta\ell'$. Note that here the disturbance range is set from -$\hat{b}$ to $\hat{b}$, which is a relatively large disturbance range, i.e. even for the extremely low bit (i.e. 1 bit) quantization, the disturbance of nearly 86.5% of weights will not exceed this range (under Laplace assumption). It can be found that $\Delta\ell'$ is strongly positive correlated to the loss, especially when the disturbance is relatively small, which indicates $\Delta\ell'$ can describe the sensitivity of the loss w.r.t. the quantization of different parameters.

In practice, we first initialize the whole network to the same bit-width, and then progressively adjust the quantization bit-width with the guidance of loss sensitivity until the average bit-width of the network reaches the objective, e.g. weights with 1.0-bit and activations with 2.0-bit. Specifically, during the training-aware quantization, we compute $\Delta\ell'$ of each channel of weights/activations in each iteration, and accumulate it across iterations in one epoch to get the overall loss sensitivity over the whole dataset. At the end of each epoch, we first sort all the channels according to $\Delta\ell'$ in ascending order, then decrease the bit-width of the channels with smaller $\Delta\ell'$. The number of channels to decrease the bit-width is set according to a ratio r, e.g. $r = 15\%$ and the bit-width of the Top-rT (T is the total channel number) channels, sorted by $\Delta\ell'$ in ascending order, will be decreased by 1.

Specifically, in our method, when the bit-width is decreased to 0, the corresponding channels of weights are pruned from the original network, resulting in a more compact network structure. Furthermore, the proposed method allows to quantize the weights to extremely low bit-width, e.g. below 1.0-bit on average. The details of distribution-aware adaptive multi-bit quantization are concluded in Alg. 1.

Algorithm 1: Distribution-aware Adaptive MBQ

Require: Training data, FP weights $\{\boldsymbol{W}_c\}_{c=1}^{T_1}$, lookup table $\mathcal{T}$ for $\mathcal{Q}(\boldsymbol{\alpha})$.
Initialize: the bit-widths of weights $\{M_c\}_{c=1}^{T_1} \leftarrow 4$, the bit-widths of activations $\{N_c\}_{c=1}^{T_2} \leftarrow 4$.
for *i in epochs* **do**
  for *Sample in Training data* **do**
    Compute full precision activations $\boldsymbol{A}_c$.
    Construct $\mathcal{Q}(\boldsymbol{\alpha})$ by $\mathcal{T}$ w.r.t M_c.
    Compute $\hat{\boldsymbol{W}}_c$ and $\hat{\boldsymbol{A}}_c$ with Eq. (5) and Eq. (7).
    Forward propagate and compute the loss ℓ.
    Backward propagate $\frac{\partial \ell}{\partial \hat{\boldsymbol{W}}_c}$ and $\frac{\partial \ell}{\partial \hat{\boldsymbol{A}}_c}$.
    Compute $\Delta\ell'_{\mathrm{W,c}}$ for weights and $\Delta\ell'_{\mathrm{A,c}}$ for activations with Eq. (11).
    Accumulate $\Delta\ell'_{\mathrm{W,c}}$ and $\Delta\ell'_{\mathrm{A,c}}$ along iterations.
    Update $\boldsymbol{W}_c$ using $\frac{\partial \ell}{\partial \hat{\boldsymbol{W}}_c}$.
  end
  Sort $\{\Delta\ell'_{\mathrm{W,c}}\}_{c=1}^{T_1}$ and $\{\Delta\ell'_{\mathrm{A,c}}\}_{c=1}^{T_2}$ ascendingly.
  $M_l = M_l - 1$ if l in Top-rT_1 sorted $\{\Delta\ell'_{\mathrm{W,c}}\}_{c=1}^{T_1}$.
  $N_l = N_l - 1$ if l in Top-rT_2 sorted $\{\Delta\ell'_{\mathrm{A,d}}\}_{c=1}^{T_2}$.
end

4. Experiments

The proposed method is verified by conducting extensive experiments on CIFAR10 [14] and ILSVRC2012 [29] with VGG [30] and ResNet-18/20/34 [12]. Besides, the ablation study is present to illustrate the effectiveness of each component of the method.

4.1. Implementation Details

We implement the algorithm with Pytorch [25] and follow the hyper-parameter settings in [12, 36, 18, 28]. As commonly used in existing methods [18, 27, 11], pre-trained full-precision model are adopted to initialize the whole network. The first and last layers are uniformly quantized into 8-bit which can be easily converted into multi-bit binary form with Eq. (7). The weights in other layers are first initialized to 4-bit and fine-tuned for a few epochs (2 epochs in our implementation), then the average bit-width are gradually decreased to the objective bit-width by applying Alg. 1 during training. Note that here we choose to initialize the model to 4-bit, since we found that the accuracy of 4-bit model is already very close to the full-precision model, as shown in Sec. 4.4. The average bit-width is defined as $\frac{\sum_{i=1}^{N} b_i}{N}$, where N is the total number of weights or activations except for the first and last layers, and b_i is the bit-width of the i-th parameter.

4.2. Evaluations on Different Datasets

Evaluation on ILSVRC12 We apply the proposed method on ResNet-18/34 [12], and compare with a number of state-of-the-art quantization methods, e.g., BWN [28], TTQ [40], HWGQ [2], INQ [38], PACT [4], LQ-Net [36], DSQ [11], AutoQ [22], APoT [18], HAWQ [33] and ALQ [27], on ILSVRC12 [29] dataset with different bit-width settings. Most of them are non-uniform methods and the mixed-precision methods are marked as float-point number in precision column.

Table 1 shows the performance of different methods with different average bit-width on ResNet-18/34. Our method shows the best appealing results in all the cases. Specifically, as for ResNet-18, we achieve 0.3%, 1.2%, 0.3%, 1.4% and 0.1% Top-1 accuracy improvement compared with state-of-the-arts methods for 1.0/32, 2.0/32, 1.0/2.0, 2.0/2.0 and 3.0/3.0 bit-width settings, respectively. As for ResNet-34, we achieve 1.0% and 2.5% Top-1 accuracy improvement compared with the state-of-the-art methods. Note that, for the 2.0/32 bit-width setting of ResNet-18, only 0.2% Top-1 and Top-5 accuracy degradation compared with the full-precision model are introduced by our method, which further demonstrate the effectiveness of our method.

Evaluation on CIFAR10 We also implement our method with ResNet-20 [12] and a small version of VGG (VGG-small) [30] on CIFAR10 [14]. We compare the performance of our method with the state-of-the-arts, i.e., BWN [28], HWGQ [2], LQ-Net [36], DSQ [11], ApoT [18] and ALQ [27], with different average bit-width settings. The structure of VGG-small is obtained from the source code of ALQ [27]. In order to quantize the fully-connected layer to mixed-precision, we directly divide the weights into n groups and progressively reduce the bit-width of those groups with LBA. In this experiment, we simply set n to be the number of the output features of weights.

Table 1. Comparison of different quantization methods with different bit-width settings (ResNet18/34 on ILSVRC12).

Method	Prec (W/A)	Size(MB)	Top-1	Top-5
ResNet-18				
FP	32/32	46.7	70.3	89.5
APoT [18]	3/3	4.6	69.9	89.2
HAWQ [33]	-/-	6.1	68.6	-
AutoQ [22]	3.7/3.2	5.7	67.5	-
Ours	**3.0/3.0**	**4.7**	**70.0**	**89.4**
TTQ [40]*	2/32	4.9	66.6	87.2
INQ [38]	3/32	4.4	68.1	88.4
LQ-Net [36]*	2/32	4.9	68.0	88.0
ALQ [27]	2.0/32	3.4	68.9	-
Ours	**2.0/32**	**3.4**	**70.1**	**89.3**
BWN [28]*	1/32	3.5	60.8	83.0
HWGQ [2]*	1/32	3.5	61.3	-
DSQ [11]*	1/32	3.5	63.7	-
ALQ [27]	1.0/32	1.8	65.6	-
Ours	**1.0/32**	**1.8**	**65.9**	**87.1**
PACT [4]*	2/2	4.9	64.4	-
LQ-Net [36]*	2/2	4.9	64.9	85.9
DSQ [11]*	2/2	4.9	65.2	-
AutoQ [22]	2.2/3.0	3.6	66.4	-
ALQ [27]	2.0/2	3.4	66.4	-
Ours	**2.0/2.0**	**3.4**	**67.8**	**88.1**
PACT [4]*	1/2	3.5	62.9	-
LQ-Net [36]*	1/2	3.5	62.6	84.3
ALQ [27]	1.0/2	1.8	63.2	-
Ours	**1.0/2.0**	**1.8**	**63.5**	**85.5**
ResNet-34				
FP	32/32	87.1	73.7	91.3
LQ-Net [36]*	2/2	7.5	69.8	89.1
DSQ [11]*	2/2	7.4	70.0	-
ALQ [27]	2.0/2	6.3	71.1	-
Ours	**2.0/2.0**	**6.3**	**72.1**	**90.7**
HWGQ [2]*	1/2	4.8	64.3	85.7
LQ-Net [36]*	1/2	4.8	66.6	86.9
ALQ [27]	1.0/2	3.4	67.3	-
Ours	**1.0/2.0**	**3.4**	**69.8**	**89.2**

*Both the first and last layers are unquantized.

Table. 2 shows the performance comparison on CIFAR10 [14]. For the ResNet-20 and VGG-small, it is obvious that the proposed method obtains the best performance in all cases. Furthermore, our method can even quantize the network under 1-bit, e.g. VGG with 0.7-bit, with acceptable accuracy degradation, which could considerably decrease the computation and memory footprint. For the cases that ResNet-20 with 2.0/32 bit-width, and VGG-small with 1.0/2.0 bit-width, we can even obtain better accuracy than the full-precision model. [1]

[1]This accuracy promotion are also observed in [18], which could be explained by the regularization effect of the quantization.

Table 2. Comparison of different quantization methods with different bit-width settings (ResNet-20/VGG-small on CIFAR10).

Method	Prec (W/A)	Top-1
ResNet-20		
FP	32/32	92.4
LQ-Net [36]	2/32	91.8
Ours	**2.0/32**	**92.5**
BWN [28]	1/32	90.1
LQ-Net [36]	1/32	90.1
DSQ [11]	1/32	90.2
Ours	**1.0/32**	**91.4**
LQ-Net [36]	2/2	90.2
APoT [18]	2/2	91.0
Ours	**2.0/2.0**	**91.7**
LQ-Net [36]	1/2	88.4
Ours	**1.0/2.0**	**90.4**
VGG-small		
FP	32/32	93.8
BWN [28]	1/32	90.1
LQ-Net [36]	2/32	93.8
ALQ [27]	0.7/32	92.0
Ours	**0.7/32**	**93.7**
HWGQ [2]	1/2	92.5
LQ-Net [36]	1/2	93.4
Ours	**1.0/2.0**	**93.9**

Table 3. Comparison of training time with state-of-the-art MBQ methods.

Method	Prec(W/A)	Time
FP	32/32	1.00×
LQ-Net [36]	2/32	1.40×
ALQ [27]	2.0/32	2.46×
DMBQ + LBA	**2.0/32**	**1.16**×
LQ-Net [36]	2/2	2.30×
LQ-Net [36]	3/3	3.70×
DMBQ	**4/4**	**1.14**×
DMBQ + LBA	**2.0/2.0**	**1.22**×

4.3. Training Time Compared with Other MBNs

Previous MBQ based methods [36, 27] propose to solve the coordinates through an iterative optimization, requiring heavy computation during training. In this section, we compare the training time of our method with the state-of-the-art MBQ methods, e.g. LQ [36] and ALQ [27], on ResNet-18. The training time of LQ-Net is obtained from the original paper. The training time of ALQ is measured by the source code which is provided by the authors. Specifically, we accumulate the training time of quantization and divide it by the training time of FP model which is trained with same epochs. For fair comparison, we measure the training time of different methods with the same strategy.

As shown in Table 3, benefiting from the look-up table strategy of DMBQ and the efficient calculation of loss sensitivity metric in LBA with backward propagation, the training time is dramatically reduced compared to the previous MBNs, i.e., LQ-Net and ALQ. Note that for the case of quantizing the bit width to 2.0/32 with mixed-precision, our method can speedup the training time by more than 2× compared with the state-of-the-art method ALQ. Compared to the full-precision model, our method only increase the training time slightly thus can be more suitable for practical applications.

Table 4. Comparison between baseline (GP), layer-wise precision (LP) and channel-wise precision (CP).

Model	Method	Prec (W)	Top-1
VGG small	GP/LP	1	92.4
	CP	**0.7**	**93.7**
ResNet-18	GP	2	68.5
	LP	2.0	69.6
	CP	**2.0**	**70.1**
ResNet-18	GP/LP	1	64.5
	CP	**1.0**	65.9

4.4. Ablation Study

Loss-guided Bit-width Allocation In this experiment, we analyze the effectiveness of the proposed LBA. We quantize the weights to global-precision using DMBQ and take it as the baseline, and we quantize the weights to layer-wise precision (LP) and channel-wise precision (CP) using DMBQ and LBA for comparison. The bit-width of LP model is adjusted in the same way with CP model, except that it is based on the gradient-based quantization sensitivity metric defined layer-wisely. The activations are kept as floating-point values for fair comparison. Both the baseline and the corresponding mixed-precision model are initialized by the same pre-trained model and trained with the same epochs and learning rate schedule.

Table 4 shows the comparison between the global-precision (GP) and mixed-precision (LP and CP). It can be observed that there is a large gap between global-precision models and channel-wise precision models, e.g. the 2.0-bit channel-wise precision ResNet18 only has 0.2% degradation of Top-1 accuracy while the 2-bit global-precision one has 1.8% degradation, compared with the Top-1 accuracy of full-precision (70.3%). The LP models have better performance than GP models but lower than CP models. Note that quantizing weights to 1.0-bit with layer-wise precision will degenerate into 1-bit global-precision, since we cannot directly quantize a whole layer to 0-bit. While the proposed LBA with channel-wise precision could enable to quantize to extremely low average bit width (i.e. under 1.0-bit) which can further remove the redundancy of the neural network (e.g. 0.7bit-VGG with 93.7% Top-1 accuracy).

We also plot the average bit-width and normalized distribution of different bit-width across layers of those three

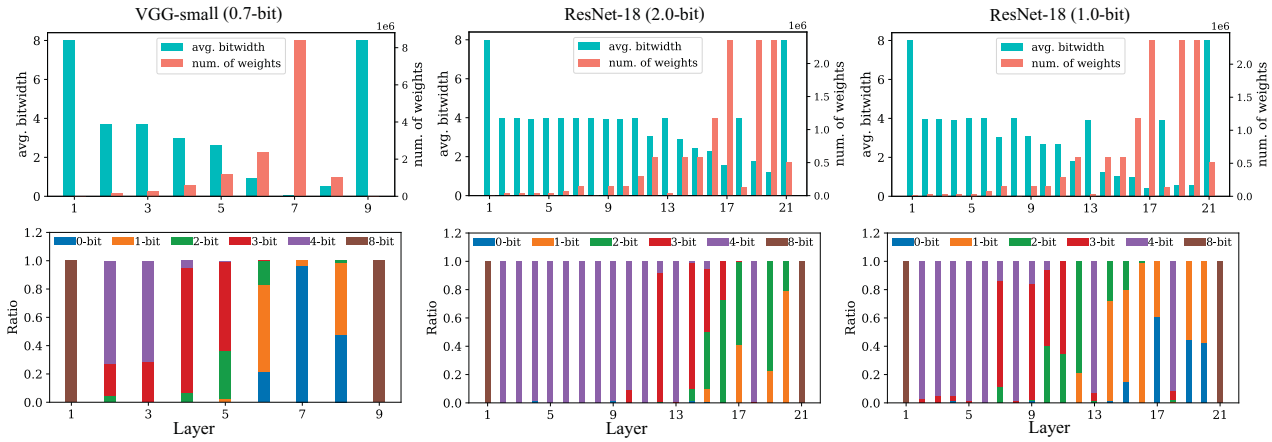

Figure 5. The statistical characteristics of CP models. Top: The average bit-width and the number of weights of each layer. Bottom: Normalized distribution of different bit-width of each layer.

channel-wise (CP) model in Fig. 5. In VGG-small, the first fully-connected layer (i.e. the 7-th layer) has the most of the parameters and the average bit-width of this layer is only 0.03-bit, where most of the parameters are pruned (bit-width are reduced to 0-bit in LBA). In addition, there is only a slight accuracy loss (i.e. 0.1%) of 0.7-bit VGG compared to floating-point model, which indicates high redundancy exists in this layer and is eliminated by the proposed LBA. In ResNet-18, it can be found that the average bit-width is roughly decreased across the layers except for the shortcut layers in ResNet18 (i.e. the 8th, 13rd, 18th layers), which indicates that the shortcut layer plays an important role in data flow and require more precision to keep the whole accuracy. It can also be found that different networks have different sensitivity with respect to accuracy (e.g. the accuracy degradation of 0.7/32-bit VGG-small is much smaller than that of 1.0/32-bit ResNet-20), which indicates that to achieve a better trade-off between accuracy and efficiency, different models should be flexibly quantized to different bit-width based on its network capacity.

Distribution-aware MBQ To demonstrate the effectiveness of DMBQ, we design two experiments for comparison: 1) uniform quantization, denoted as uniform. The weights are uniformly quantized under the maximum absolute value of each channel (i.e. $[-\max|\boldsymbol{W}_{\mathrm{i}}|, \max|\boldsymbol{W}_{\mathrm{i}}|]$ where i means the i-th channel). 2) DMBQ, the proposed MBQ method. Those experiments are implemented on ResNet-18 with ImageNet dataset. Activations are quantized using the method described in Sec. 3.1 and no LBA is introduced for clear comparison. Table. 5 summarizes the results of ResNet-18 with different methods. As shown, DMBQ could achieve 2.4%/1.8%, 0.7%/0.4%, 0.2%/0.1% Top-1/Top-5 accuracy improvement than uniform quantization for 2-bit, 3-bit, 4-bit respectively. Besides, the lower the bit width, the higher the accuracy gain, which further demonstrate the effectiveness of the proposed DMBQ methods.

Table 5. Comparison of uniform quantization and DMBQ.

Method	Prec (W/A)	Top-1	Top-5
FP	32	70.3	89.5
Uniform	2/2	62.7	84.6
	3/3	68.5	88.4
	4/4	70.0	89.3
DMBQ	2/2	65.1	86.4
	3/3	69.2	88.8
	4/4	70.2	89.4

5. Conclusion

In this paper, we propose a novel distribution-aware quantization method (DMBQ) and loss-guided bit-width allocation (LBA) for multi-bit binary networks. Through incorporating the distribution prior of weights, we searched the optimal MBQ with brute force searching beforehand and realize efficient MBQ in the training process based on a lookup-table based strategy. Based upon DMBQ, we propose LBA to adaptively reduce the bit-width of weights and activations with the guidance of the loss sensitivity metric. Experimental results show that our method not only outperforms state-of-the-art quantized networks in terms of accuracy but also is more efficient in terms of training time compared with state-of-the-art MBNs.

6. Acknowledgments

This work was supported by National Science Foundation of China (No. 61971465 and No. 61671236), Fundamental Research Funds for the Central Universities of China (No. 0210-14380145 and No. 0210-14380155), Shenzhen Science and Technology Project under Grant (GGFW2017040714161462, JCYJ20180226181021364).

References

[1] Ron Banner, Yury Nahshan, and Daniel Soudry. Post training 4-bit quantization of convolutional networks for rapid-deployment. In *Proceedings of Advances in Neural Information Processing Systems*, pages 7950–7958, 2019.

[2] Zhaowei Cai, Xiaodong He, Jian Sun, and Nuno Vasconcelos. Deep learning with low precision by half-wave gaussian quantization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5918–5926, 2017.

[3] Jungwook Choi, Pierce I-Jen Chuang, Zhuo Wang, Swagath Venkataramani, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. Bridging the accuracy gap for 2-bit quantized neural networks (qnn). *arXiv preprint arXiv:1807.06964*, 2018.

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

[5] Yoni Choukroun, Eli Kravchik, Fan Yang, and Pavel Kisilev. Low-bit quantization of neural networks for efficient inference. In *2019 IEEE International Conference on Computer Vision Workshop*, pages 3009–3018, 2019.

[6] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to + 1 or -1. *arXiv preprint arXiv:1602.02830*, 2016.

[7] Li Deng, Geoffrey Hinton, and Brian Kingsbury. New types of deep neural network learning for speech recognition and related applications: An overview. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8599–8603. IEEE, 2013.

[8] Zhen Dong, Zhewei Yao, Yaohui Cai, Daiyaan Arfeen, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Hawq-v2: Hessian aware trace-weighted quantization of neural networks. *arXiv preprint arXiv:1911.03852*, 2019.

[9] Zhen Dong, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Hawq: Hessian aware quantization of neural networks with mixed-precision. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 293–302, 2019.

[10] Ahmed T Elthakeb, Prannoy Pilligundla, FatemehSadat Mireshghallah, Amir Yazdanbakhsh, Sicun Gao, and Hadi Esmaeilzadeh. ReLeQ: an automatic reinforcement learning approach for deep quantization of neural networks. *arXiv preprint arXiv:1811.01704*, 2018.

[11] 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 *Proceedings of the IEEE International Conference on Computer Vision*, pages 4852–4861, 2019.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[13] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2704–2713, 2018.

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

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

[16] Fengfu Li, Bo Zhang, and Bin Liu. Ternary weight networks. *arXiv preprint arXiv:1605.04711*, 2016.

[17] Rundong Li, Yan Wang, Feng Liang, Hongwei Qin, Junjie Yan, and Rui Fan. Fully quantized network for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2810–2819, 2019.

[18] Yuhang Li, Xin Dong, and Wei Wang. Additive powers-of-two quantization: A non-uniform discretization for neural networks. In *Proceedings of International Conference on Learning Representations*, 2020.

[19] Darryl Lin, Sachin Talathi, and Sreekanth Annapureddy. Fixed point quantization of deep convolutional networks. In *Proceedings of International Conference on Machine Learning*, pages 2849–2858, 2016.

[20] Xiaofan Lin, Cong Zhao, and Wei Pan. Towards accurate binary convolutional neural network. In *Proceedings of Advances in Neural Information Processing Systems*, pages 345–353, 2017.

[21] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[22] Qian Lou, Feng Guo, Minje Kim, Lantao Liu, and Lei Jiang. Autoq: Automated kernel-wise neural network quantization. In *Proceedings of International Conference on Learning Representations*, 2019.

[23] Xingchen Ma, Amal Rannen Triki, Maxim Berman, Christos Sagonas, Jacques Cali, and Matthew B Blaschko. A bayesian optimization framework for neural network compression. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 10274–10283, 2019.

[24] Daisuke Miyashita, Edward H Lee, and Boris Murmann. Convolutional neural networks using logarithmic data representation. *arXiv preprint arXiv:1603.01025*, 2016.

[25] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[26] Haotong Qin, Ruihao Gong, Xianglong Liu, Mingzhu Shen, Ziran Wei, Fengwei Yu, and Jingkuan Song. Forward and backward information retention for accurate binary neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2250–2259, 2020.

[27] Zhongnan Qu, Zimu Zhou, Yun Cheng, and Lothar Thiele. Adaptive loss-aware quantization for multi-bit networks. In

Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 7988–7997, 2020.

[28] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *Proceedings of the European Conference on Computer Vision*, pages 525–542. Springer, 2016.

[29] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[30] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[31] Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. Haq: Hardware-aware automated quantization with mixed precision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8612–8620, 2019.

[32] Chen Xu, Jianqiang Yao, Zhouchen Lin, Wenwu Ou, Yuanbin Cao, Zhirong Wang, and Hongbin Zha. Alternating multi-bit quantization for recurrent neural networks. In *Proceedings of International Conference on Learning Representations*, 2018.

[33] Zhewei Yao, Zhen Dong, Zhangcheng Zheng, Amir Gholami, Jiali Yu, Eric Tan, Leyuan Wang, Qijing Huang, Yida Wang, Michael W Mahoney, et al. Hawqv3: Dyadic neural network quantization. *arXiv preprint arXiv:2011.10680*, 2020.

[34] Penghang Yin, Jiancheng Lyu, Shuai Zhang, Stanley Osher, Yingyong Qi, and Jack Xin. Understanding straight-through estimator in training activation quantized neural nets. *arXiv preprint arXiv:1903.05662*, 2019.

[35] Linghua Zeng, Zhangcheng Wang, and Xinmei Tian. Kcnn: Kernel-wise quantization to remarkably decrease multiplications in convolutional neural network. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 4234–4242, 2019.

[36] Dongqing Zhang, Jiaolong Yang, Dongqiangzi Ye, and Gang Hua. LQ-nets: Learned quantization for highly accurate and compact deep neural networks. In *Proceedings of the European Conference on Computer Vision*, pages 365–382, 2018.

[37] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017.

[38] Aojun Zhou, Anbang Yao, Yiwen Guo, Lin Xu, and Yurong Chen. Incremental network quantization: Towards lossless cnns with low-precision weights. *arXiv preprint arXiv:1702.03044*, 2017.

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

[40] Chenzhuo Zhu, Song Han, Huizi Mao, and William J Dally. Trained ternary quantization. *arXiv preprint arXiv:1612.01064*, 2016.