Nonuniform-to-Uniform Quantization: Towards Accurate Quantization via Generalized Straight-Through Estimation

Zechun Liu\textsuperscript{1,2,4} Kwang-Ting Cheng\textsuperscript{1} Dong Huang\textsuperscript{2} Eric Xing\textsuperscript{2,3} Zhiqiang Shen\textsuperscript{2,3}
\textsuperscript{1}Hong Kong University of Science and Technology \textsuperscript{2}Carnegie Mellon University \textsuperscript{3}Mohamed bin Zayed University of Artificial Intelligence \textsuperscript{4}Reality Labs, Meta Inc.
{zliubq, timcheng}@ust.hk epxing@cs.cmu.edu {dghuang,zhiqians}@andrew.cmu.edu

Abstract

The nonuniform quantization strategy for compressing neural networks usually achieves better performance than its counterpart, i.e., uniform strategy, due to its superior representational capacity. However, many nonuniform quantization methods overlook the complicated projection process in implementing the nonuniformly quantized weights/activations, which incurs non-negligible time and space overhead in hardware deployment. In this study, we propose Nonuniform-to-Uniform Quantization (N2UQ), a method that can maintain the strong representational ability of nonuniform methods while being hardware-friendly and efficient as the uniform quantization for model inference. We achieve this through learning the flexible in-equidistant input thresholds to better fit the underlying distribution while quantizing these real-valued inputs into equidistant output levels. To train the quantized network with learnable input thresholds, we introduce a generalized straight-through estimator (G-STE) for intractable backward derivative calculation w.r.t. threshold parameters. Additionally, we consider entropy preserving regularization to further reduce information loss in weight quantization. Even under this adverse constraint of imposing uniformly quantized weights and activations, our N2UQ outperforms state-of-the-art nonuniform quantization methods by 0.5 ~ 1.7% on ImageNet, demonstrating the contribution of N2UQ design. Code and models are available at: https://github.com/liuzechun/Nonuniform-to-Uniform-Quantization.

1. Introduction

Deep Neural Networks (DNNs) have demonstrated great success in various real-world applications [16, 41]. Despite their remarkable results, the large model size and high computational cost hinder pervasive deployment of DNNs, especially on resource-constrained devices. A number of approaches have been proposed to compress and accelerate DNNs, including channel pruning [27, 29], quantization [31, 47, 50], neural architecture search [6, 43], etc.

Among these methods, quantization-based methods have shown promising results in compressing the model size by representing weights with fewer bits, and faster inference by replacing computationally-heavy convolution operations with efficient bitwise operations [36, 47]. Despite these advantages, quantized DNNs still have a non-negligible performance gap from their full-precision counterparts, especially with extremely low-bit quantization. For example,
the 2-bit classic uniformly quantized ResNet-50 achieves 67.1% top-1 accuracy [50] on ImageNet dataset, a drop of 9.9% compared to a real-valued ResNet-50. This performance gap mainly results from the quantization error in representing real-valued weights and activations with a limited number of quantized levels and the inflexibility for uniform quantizers to adapt to different distributions of input values.

To better fit underlying distributions and mitigate quantization errors, several previous studies proposed nonuniform quantization by adjusting the quantization resolution according to the density of real-valued distribution [46,47]. However, the accuracy improvement of nonuniform quantization usually comes at the expense of hardware implementation efficiency [13,50]. Since the output of the nonuniform quantization are floating-point weights and activations, their multiplication can no longer be directly accelerated by the bitwise operation between binaries [22]. A common solution to addressing this issue has been building look-up tables (LUTs) to map floating-point values to binary digits [15,46], as shown in Fig. 1(a). This post-processing incurred by nonuniform quantization costs more hardware area and consumes additional energy compared to uniform quantization [1,13].

The goal of this study is to develop a new quantization method maintaining the hardware projection simplicity as uniform quantization and meanwhile offering the flexibility to achieve the merit of nonuniform quantization. Despite that it is desirable to enforce a quantizer’s outputs (i.e., quantized weights/activations) to have uniform quantization levels in order not to incur additional post-processing tasks, each of the output levels does not necessarily need to represent an equal range of the real-valued input. As shown in Fig. 1(b), we enforce the output quantization levels to be equidistant while learning the thresholds on the input values to incorporate more flexibility in fitting the underlying real-valued distributions for quantization. We name this quantizer design as Nonuniform-to-Uniform Quantizer (N2UQ).

However, it is challenging to optimize such a nonuniform-to-uniform quantizer, which can automatically learn to adapt the input thresholds through network training for higher precision. Because the gradient computation w.r.t. the threshold parameters is intractable, and cannot be resolved by the existing gradient estimation method for quantization, i.e., the straight-through estimator (STE) [2]. STE simply estimates the incoming gradient to a threshold operation to be equal to the outgoing gradient, which by definition is unable to incorporate the threshold difference in training neither to update the threshold with gradients.

To circumvent this challenge, we revisit the earliest derivation of STE from the stochastic binarization [2] and derive a novel and more flexible backward approximation method for quantization. We name it as Generalized Straight-Through Estimator (G-STE). It degenerates to STE when all the input intervals are equal-sized, while for the scenarios that require nonuniform input thresholds, it automatically adapts the thresholds with gradient learning and provides a finer-grained approximation to the quantization function. Specifically, G-STE encodes the expectation of stochastic quantization into the backward approximation to the forward deterministic quantization functions, which naturally converts the intractable gradient computation w.r.t. the input threshold parameters to that w.r.t. the slopes, and encodes the influence from input threshold difference to the remaining network in the backward gradient computation.

Moreover, we propose the weight regularization that considers the overall entropy in a weight filter to further reduce the information loss arising from weight quantization. We extensively evaluate the effectiveness of the proposed N2UQ with the collective contributions of the threshold-learning quantizer via G-STE and the weight regularization on ImageNet with different architectures and different bit-width constraints. Under all deployment scenarios, N2UQ consistently improves the accuracy by a significant margin compared to the state-of-the-art methods, including both uniform and nonuniform quantization.

The contribution of this paper includes four aspects:
- We propose Nonuniform-to-Uniform Quantizer (N2UQ) for improving the quantization precision via learning input thresholds, while maintaining hardware-friendliness in implementation similar to uniform quantization.
- We propose Generalized Straight-Through Estimator (G-STE) to tackle intractable gradient computation w.r.t. input threshold parameters in N2UQ. G-STE calculates the expectation of the stochastic quantization as the backward approximation to the forward deterministic quantization.
- Based on entropy analysis, we propose a novel weight regularization considering the overall weight distribution for further preserving information in weight quantization.
- We demonstrate that even under the strict constraint of fixing the quantized weights and activations to be uniform and only learning input thresholds, N2UQ exceeds state-of-the-art nonuniform quantization method with 0.5~1.7% higher accuracy on ImageNet. Specifically, the 2-bit ResNet-50 model achieves 76.4% top-1 accuracy on ImageNet, reducing the gap to its real-valued counterpart to only 0.6%, demonstrating the effectiveness of N2UQ design.

2. Related Work

Model compression is a useful technology for deploying neural network models to mobile devices with limited storage and computational power [5,6], and has attracted increasing attention. Model compression methods can be categorized into several major categories, including quantization [31,47,50,53], pruning [11,27,29], knowledge distillation [17,40], compact network design [18,32,39,48], etc. This work is mainly focused on quantization.
Quantization can be further classified to uniform quantization [7, 8, 14, 19, 20, 50] and nonuniform quantization [26, 34, 46, 47]. Compared to uniform approaches, nonuniform quantization may achieve higher accuracy because it can better capture the underlying distributions by learning to allocate more quantization levels to important value regions [13]. For example, PoT [34] uses powers-of-two levels and ApoT [26] further proposed additive powers-of-two quantization. However, it is typically difficult to deploy nonuniform quantization efficiently on the general-purpose hardware [46], e.g., GPU and CPU. Since to accelerate nonuniform quantization with bitwise operations requires additional operations or designs like look-up tables (LUTs) for mapping between floating-point quantization outputs and their binary digit representations [15, 22], which is less efficient compared to uniform quantization. Our proposed N2UQ is motivated to combine the merits of uniform and nonuniform quantization by outputting uniformly quantized levels for hardware-friendly implementation while allowing effective input thresholds learning to fit the underlying distributions for higher accuracy.

Moreover, the quantization function is intrinsically a discontinuous step function and nearly always has zero gradients w.r.t. inputs. To circumvent this problem, early work proposed straight-through estimation (STE) [2] for gradient estimation, which is widely adopted in subsequent quantization methods [8, 12, 19, 46, 47, 50, 53]. However, STE simply backpropagates through the hard threshold function as if it had been the identity function, which is unable to incorporate non-linearities in quantizer’s input and output to fit various distributions. To deal with this restriction, previous works manually add non-linear functions before and/or after the STE-based quantizer. For example, LCQ [46] adds compressing and expanding function to STE-based quantizer to achieve nonuniform quantization. QIL [23] proposes to learn a power function before the STE-based quantizer for controlling quantization interval sizes. However, these methods relying on STE suffer from the discrepancy between quantizer’s thresholds and nonlinear function’s turning points [46] or the inflexibility of using a human-defined smooth function to control all the quantization intervals with a single power hyper-parameter [23]. To overcome this limitation, we proposed Generalized Straight-Through Estimator (G-STE) to incorporate flexible non-linearity learning inside the quantizer, as detailed in Sec. 3

3. Method

In this section, we first briefly introduce the objective of quantization in Sec. 3.1. Then we present our nonuniform-to-uniform quantization (N2UQ) in Sec. 3.2 with three components: threshold learning quantizer in the forward pass in Sec. 3.2.1, its backward approximation with Generalized Straight-Through Estimator (G-STE) in Sec. 3.2.2 and an additional weight regularization method in Sec. 3.2.3.

3.1. Preliminaries

Matrix multiplication is the most computation-costly operation in deep neural networks. By quantizing the weights and activations in the convolutional layers and fully connected layers to fixed bits, the matrix multiplication can be accelerated with bitwise operations [47, 50]:

\[
a^q \cdot w^q = \sum_{i=0}^{M-1} \sum_{j=0}^{K-1} 2^{i+j} \cdot \text{popcnt}[(a_i, w_j)],
\]

(1)

Here a and w are binary vectors of the quantized activation \(a^q\) and quantized weight \(w^q\). \(M\) and \(K\) denote the number of bits used in representing \(a^q\) and \(w^q\), respectively. While Eq. 1 in general holds true for uniform quantization, the situation becomes complicated when considering the nonuniform quantization.

3.2. Nonuniform-to-Uniform Quantization

To sort out the inner mechanism of quantization, we summarize two properties of the quantizer:

**Property 1 (Quantizer’s output).** The premise of quantized network being accelerated with Eq. 1 is that the quantizer’s output (i.e., quantized weights \(w^q\) and activations \(a^q\)) can be represented in OR mapped to binary digits within the fixed bits: 

\[
a^q = \sum_{i=0}^{M-1} a_i 2^i, \quad w^q = \sum_{j=0}^{K-1} w_j 2^j
\]

[50].

This property is straightforward since Eq. 1 takes place between binary vectors. While uniformly quantized \(a^q\) and \(w^q\) can be easily converted to binaries via linear mappings, it usually requires additional operations [26] or look-up tables (LUTs) [15, 46] for encoding ‘n-bit’ nonuniformly quantized \(a^q\) and \(w^q\) to \(n\)-bit binary digits, because outputs of \(n\)-bit nonuniform quantizer are actually \(2^n\) in-equidistant floating-point values, as illustrated in Fig. 1. This post-processing step incurs larger hardware area and more energy consumption as mentioned in hardware studies [1, 21].

**Property 2 (Quantizer’s input):** The uniformness in input and output levels of the quantization function \(x^q = F_Q(x^r)\) can be detached through a proper quantizer design, i.e., uniform quantization levels in output \(x^q\) do not necessarily need to represent uniform ranges from the input \(x^r\).

This property originates from the observation that the quantizer represents real-valued variables \(x^r\) with several quantized levels \(x^q\), while the quantized levels are fixed, the real-valued distributions are diverse. Thus, allowing learnable thresholds in dividing \(x^r\) to fixed quantized levels \(x^q\) theoretically can incorporate more representational flexibility for achieving higher precision.
threshold operation to be the outgoing gradients: \( x^r \) represent the input real-valued variables and the output quantized variables, respectively. The goal of N2UQ is to encode threshold learning in the quantizer to allow more flexibility, while output uniformly quantized weights and activations to accommodate fast bitwise operations without the post-processing step between quantization and matrix multiplication.

### 3.2.2 Backward Pass: Generalized Straight-Through Estimator (G-STE)

However, backpropagating through Eq. 2 is difficult because: (I) The derivatives of Eq. 2 w.r.t. input \( x_r \) are almost zero everywhere. (II) The gradient calculation w.r.t. threshold parameters is intractable.

For challenge (I), previous quantization works adopt straight-forward estimator (STE) to approximate the backward gradients in the quantization functions [8, 19, 28, 44, 46, 50]. STE simply assigns the incoming gradients to a threshold operation to be the outgoing gradients:

\[
\frac{\partial L}{\partial x^q_{i,l}} = \frac{\partial L}{\partial x^r_{i,l}}
\]

\( x^q_{i,l} \) and \( x^r_{i,l} \) denote the \( i^{th} \) quantized / real-valued variables in \( l^{th} \) layer, respectively. This simple approximation function works well for uniform quantizers. However, STE implicitly enforces the equal axis aspect ratio in the input and output intervals of the quantizer because it regards the quantization function as an identity function in the backward pass. This obstructs the quantizer design from allowing learnable input thresholds while fixing the output levels. Moreover, STE cannot deal with challenge (II), gradient computation w.r.t. learnable thresholds since STE basically just bypasses the gradients as if the quantization function does not exist. Thus, to come up with a finer-grained and more flexible backward approximation that can tackle the gradient computation w.r.t. input thresholds, we revisit STE in stochastic binarization and derive the proposed Generalized Straight-Through Estimator (G-STE).

**Lemma 1.** In binarization, straight-through estimation (STE) for gradient approximation to the forward deterministic binarization function can be derived from the expectation of the stochastic binarization function [19].

The concept of stochastic binarization and deterministic binarization are first proposed in [2, 9, 10]. In stochastic binarization, real-valued variables are binarized to \(-1/1\) stochastically according to their distances to \(-1/1\):

\[
\hat{x}^{b}_{i,l} = \begin{cases} 
-1 & \text{with probability } p = \text{clip}\left(\frac{1-x^{r}_{i,l}}{2}, 0, 1\right) \\
1 & \text{with probability } p = \text{clip}\left(\frac{1+x^{r}_{i,l}}{2}, 0, 1\right)
\end{cases}
\]

where \( \hat{x}^{b}_{i,l} \) denotes the stochastic binary variables. To update \( W_{i,j,l} \) (weights in layer \( l \) connecting neuron \( j \) in layer \( l-1 \) to neuron \( i \) in layer \( l \)), the expected gradients through the stochastic binarization function are computed:

\[
\mathbb{E}\left[\frac{\partial L}{\partial W_{i,j,l}}\right] = \mathbb{E}_{\hat{x}^{b}_{i,l}}\left[\frac{\partial L}{\partial \hat{x}^{b}_{i,l}}\mathbb{E}_{\hat{x}^{b}_{i,l}}\left[\frac{\partial \hat{x}^{b}_{i,l}}{\partial x^{r}_{i,l}}\frac{\partial x^{r}_{i,l}}{\partial W_{i,j,l}}\right]\right],
\]

where \( \mathbb{E} \), \( \mathbb{E}_{\hat{x}^{b}_{i,l}} \) and \( \mathbb{E}_{\hat{x}^{b}_{i,l}} \) respectively, the expectation over the whole network, only the stochastic binary variables and other parts except the stochastic binary variables. Specifically,

\[
\mathbb{E}_{\hat{x}^{b}_{i,l}}\left[\frac{\partial \hat{x}^{b}_{i,l}}{\partial x^{r}_{i,l}}\right] = \frac{\partial}{\partial x^{r}_{i,l}}\left(\mathbb{E}_{\hat{x}^{b}_{i,l}}\left[\hat{x}^{b}_{i,l}\right]\right)
\]

\[
= \frac{\partial}{\partial x^{r}_{i,l}}\left(\text{clip}(x^{r}_{i,l}, -1, 1)\right)
\]

This arrives at the common straight-through estimator for the binarization functions [3, 36, 51], which transmits the gradients identically near the threshold and disregards the gradients when the real-valued inputs are too far from the threshold (i.e., \( x^{r} > 1 \) or \( x^{r} < -1 \)). Meanwhile, the widely-adopted deterministic binarization function in forward pass [4, 9, 28, 30, 31] can be attained via setting a hard threshold on the probability (i.e., \( p = 0.5 \)) in Eq. 4:

\[
x^{b}_{i,l} = \begin{cases} 
-1 & p(\hat{x}^{b}_{i,l} = -1) > 0.5 \\
1 & p(\hat{x}^{b}_{i,l} = 1) \leq 0.5 \\
0 & p(\hat{x}^{b}_{i,l} = -1) \leq 0.5 \text{ and } p(\hat{x}^{b}_{i,l} = 1) > 0.5
\end{cases}
\]

To this end, we show that STE encodes the expectation of stochastic binarization in the backward approximation to the forward deterministic binarization functions.

**Lemma 2.** A quantization function \( x^{q} \) can be regarded as a summation of binarization functions \( x^{b} \) with different thresholds: \( x^{q} = x^{b}_{\text{threq}_1} + x^{b}_{\text{threq}_2} + \ldots + x^{b}_{\text{threq}_n} \).

This lemma is self-evident, as illustrated in Fig. 2 (b). From here, we extend the concept of stochastic binarization to quantization for deriving G-STE to solve the threshold learning problem in nonuniform-to-uniform quantizer.

We start with the first quantization segment. We denote the initial point as \( s \) and its length as \( a_1 \). For simplification, we omit the subscripts here. Following the concept
of stochastic binarization, in the range of $[s, s + a_1]$, real-valued variables can be quantized to 0/1 stochastically with the probability proportional to their distance to $s/s + a_1$:

$$\tilde{x}^q_{(0, 1)} = \begin{cases} 0 & \text{with probability } p = \text{clip} \left( \frac{x^r - s}{a_1}, 0, 1 \right) \\ 1 & \text{with probability } p = \text{clip} \left( \frac{x^r - s}{a_1}, 0, 1 \right) \end{cases}$$

$\tilde{x}^q_{(0, 1)}$ is the stochastic quantization output within this segment. Similar to Eq. 6, the derivation of this quantization segment can be computed from the expectation of Eq. 8:

$$E \left[ \frac{\partial \tilde{x}^q_{(0, 1)}}{\partial x^r} \right] = \frac{\partial}{\partial x^r} E \left[ \tilde{x}^q_{(0, 1)} \right] = \frac{\partial}{\partial x^r} \left( 0 \times p(\tilde{x}^q = 0) + 1 \times p(\tilde{x}^q = 1) \right)$$

$$= \frac{\partial}{\partial x^r} \text{clip} \left( \frac{x^r - s}{a_1}, 0, 1 \right).$$

In this way, the influence of the threshold parameter $a_1$ to the network is decently encoded in the backward approximation function Eq. 9 by altering the slopes. Further in the forward pass, instead of using stochastic quantization which requires random seeds generating process [42], deterministic quantization are adopted and derived by setting the probability threshold to be $p = 0.5$,

$$x^q_{(0, 1)} = \begin{cases} 0 & p(\tilde{x}^q = 0) > 0.5 \\ 1 & p(\tilde{x}^q = 0) \leq 0.5 \end{cases} = \begin{cases} 0 & x^r < s + \frac{a_1}{2} \\ 1 & x^r \geq s + \frac{a_1}{2} \end{cases}.$$ 

Likewise, the forward deterministic quantization functions and the corresponding backward approximation functions for the remaining segments can be obtained from this probabilistic sense. Then, the nonuniform-to-uniform quantizer with its backward approximation for $n$-bit quantization can be naturally derived by summing up $2^n - 1$ segments, as illustrated in Fig. 2. We name this proposed backward gradient approximation method as Generalized Straight-Through Estimator (G-STE) and obtain the following theorem.

**Theorem 1** Generalized straight-through estimator:

$$\frac{\partial x^q}{\partial x^r} = E \left[ \frac{\partial \tilde{x}^q}{\partial x^r} \right] = \frac{\partial}{\partial x^r} E \left[ \tilde{x}^q \right] = \begin{cases} \frac{\partial}{\partial x^r} \left( \frac{x^r - d_i}{a_1} + i - 1 \right) & d_{i-1} \leq x^r < d_i \\ 0 & \text{otherwise} \end{cases}$$

for scaling the input $x^r$ before the quantization: $x^r := x^r \times \beta_1$ and the output $x^q$ after the quantization: $x^q := x^r \times \beta_2$, $\beta_1, \beta_2$ are initialized to 1, $a_i$ are initialized to $2/(2^n - 1)$ and enforced to be positive. The proposed nonuniform-to-uniform quantizer introduces only $2^n + 2$ extra parameters per layer compared to a classical uniform quantizer, which is negligible considering the large number of network weights.
3.2.3 Entropy Preserving Weight Regularization

Further, we propose weight regularization to encourage more information-carrying capacity in quantized weights. An important observation is that weights in real-valued networks are usually small in magnitude, e.g., $\sim 10^{-2}$ for weights in a pre-trained ResNet-18 network, but the quantized weights usually expand in the range of $[-1, 1]$. This mismatch in magnitude will cause quantized weights to collapse to a few quantization levels close to zero. The classic solution is using a $tanh$ function and dividing the maximum absolute weight value to rescale weights to $[-1, 1]$ [50]. However, these rescaled weights are likely to be dominant by a few extrema and still not fully occupy the range of $[-1, 1]$, as will further be illustrated in Fig. 3, which causes a huge information loss.

From the perspective of information theory, more information are preserved when quantized weights contain higher entropy. Thus, we are motivated to regularize the real-valued weights before the quantizer to obtain maximal entropy in quantized weights:

$$\max_p H(p) = \max_p \sum_{i=1}^{N} p_i \log(p_i), \text{s.t. } \sum_{i=1}^{N} p_i = 1$$

Here $p_i$ is the proportion of real-valued weights being quantized to $i^{th}$ quantization level and $N$ is the number of quantization levels in total. Based on the Lagrange multiplier, it is easy to obtain the optimal $p_i^* = \frac{1}{N}, i \in \{1, 2, \ldots, N\}$, meaning that, when the proportions of real-valued weights being quantized to multiple quantization levels are equal, the information entropy in the quantized weights reaches its maximum value.

Given the quantization function $F_Q = \text{round}(\text{clip}(w^r, -1, 1) + 1) \times 2^{n-1} + \frac{2}{2^n - 1} - 1$, we empirically solve that when real-valued weights are normalized to $W^r = 2^{(n-1)} ||W^r||_1 \cdot W^r$, the corresponding quantized weights $w^q = F_Q(w^r, n)$ are approximately uniformly distributed in all quantization levels. Here, $W^r$ is the real-valued weight filter, $||W^r||_1$ denotes the number of entries in $W^r$, more details of the derivation can be found in Appendix.

4. Experiments

To verify the effectiveness of the proposed Nonuniform-to-Uniform Quantization (N2UQ), we conduct experiments on the ImageNet dataset. We first introduce the dataset and training strategy in Sec. 4.1, followed by the comparison to state-of-the-art quantization methods in Sec. 4.2. We then analyze the effect of each proposed component of N2UQ in Sec. 4.3. Visualization results on how N2UQ captures fine-grained underlying distributions are presented in Sec. 4.4.

4.1. Experimental Settings

Dataset The experiments are carried out on the ImageNet-2012 classification dataset [37], with 1.2 million training images and 50,000 validation images in 1000 classes.

Training Strategy We follow the training scheme in [12, 23, 46, 50] to use real-valued PyTorch pre-trained models as initialization for corresponding quantized networks. We use Adam optimizer [24] with a linear learning rate decay scheduler. The initial learning rate is set to $2.5e-3$ for weight parameters and batch size is set to 512. We set weight decay to be 0 following [30, 31]. The models are trained for 128 epochs with the same knowledge distillation scheme as LSQ [12]. We adopt the basic data augmentation as ResNet [16]. Training images are randomly resized and cropped to $224 \times 224$ pixels and randomly flipped horizontally. Test images are center-cropped to $224 \times 224$ pixels. For fair comparison, the floating-point (FP) model results reported in Table 1 are initialized and fine-tuned with the same settings as quantized models.

Learnable Parameters For the learnable parameters, $s$ are initialized to 0, $a_i$ are initialized to $2/(2^n - 1)$ and enforced to be greater than $1e-3$. The learnable scaling parameters $\beta_1, \beta_2$ are initialized to 1. All these learnable parameters use $1/10$ of the learning rate as that for weight parameters.

Network Structure We adopt Pre-Activation structures (i.e., NonLinear-Conv-BN structure) and use RPReLU [30] as non-linear function. All convolution and fully-connected layers are quantized with N2UQ except for the first and the last one.

4.2. Comparison with State-of-the-Art Methods

Table 1 summarizes the accuracy of the proposed N2UQ on ResNet. Compared to uniform quantization, N2UQ allows more flexibility in learning input thresholds, and the results demonstrate that N2UQ surpasses uniform methods [8, 12, 14, 50] by a large margin. Moreover, the accuracy improvements become more significant with larger bit-width, as more flexibility can be incorporated with threshold learning. For nonuniform quantization, the state-of-the-art method LCQ [46] intrinsically uses the same quantizer as the classic uniform quantizer [8, 50] and learns two additional non-linear functions to reshape the real-valued input and quantized output. Instead, the proposed N2UQ completely integrates the non-linearity learning inside the quantizer, which avoids the discrepancy between the non-linear functions’ turning points and the quantizers’ thresholds. Thus, the proposed N2UQ achieves up to 1.7% higher accuracy than LCQ, showing that N2UQ is well designed to incorporate threshold learning into the quantizers for higher flexibility and better performance. Moreover, compared to nonuniform quantization, N2UQ not only saves the parameters by fixing the output levels, but also results in uniformly quantized weights and activations for computing the quantized matrix multiplication, which prevents the post-processing step of building look-up tables (LUTs) and mapping the floating-point levels to binaries, thus is much more hardware-friendly [13, 20, 22]. For the MobileNetV2
Table 1. Accuracy comparison to the state-of-the-art quantization methods with ResNet structure on ImageNet dataset. Note that W/A denote the bit-width of weights and activations, respectively. FP denotes the top-1 accuracy of the full-precision models.

<table>
<thead>
<tr>
<th>Network</th>
<th>Method</th>
<th>Bit-width (W/A)</th>
<th>Top-1 Accuracy (%)</th>
<th>Bit-width (W/A)</th>
<th>Top-5 Accuracy (%)</th>
<th>Bit-width (W/A)</th>
<th>Top-1 Accuracy (%)</th>
<th>Bit-width (W/A)</th>
<th>Top-5 Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResNet-18</td>
<td>PACT [8]</td>
<td>2/2</td>
<td>64.4</td>
<td>85.6</td>
<td>3/3</td>
<td>68.1</td>
<td>88.2</td>
<td>4/4</td>
<td>69.2</td>
</tr>
<tr>
<td></td>
<td>DoReFa-Net [50]</td>
<td>2/2</td>
<td>64.7</td>
<td>84.4</td>
<td>3/3</td>
<td>67.5</td>
<td>87.6</td>
<td>4/4</td>
<td>68.1</td>
</tr>
<tr>
<td></td>
<td>LQ-Nets [47]</td>
<td>2/2</td>
<td>64.9</td>
<td>85.9</td>
<td>3/3</td>
<td>68.2</td>
<td>87.9</td>
<td>4/4</td>
<td>69.3</td>
</tr>
<tr>
<td></td>
<td>DSQ [14]</td>
<td>2/2</td>
<td>65.2</td>
<td>–</td>
<td>3/3</td>
<td>68.7</td>
<td>–</td>
<td>4/4</td>
<td>69.6</td>
</tr>
<tr>
<td></td>
<td>FAQ [33]</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>DSQ [14]</td>
<td>2/2</td>
<td>65.7</td>
<td>–</td>
<td>3/3</td>
<td>69.2</td>
<td>–</td>
<td>4/4</td>
<td>70.1</td>
</tr>
<tr>
<td></td>
<td>QIL [23]</td>
<td>2/2</td>
<td>66.9</td>
<td>–</td>
<td>3/3</td>
<td>69.6</td>
<td>–</td>
<td>4/4</td>
<td>70.5</td>
</tr>
<tr>
<td></td>
<td>DSQ [14]</td>
<td>2/2</td>
<td>65.7</td>
<td>–</td>
<td>3/3</td>
<td>69.2</td>
<td>–</td>
<td>4/4</td>
<td>70.1</td>
</tr>
<tr>
<td></td>
<td>LQ-Nets [47]</td>
<td>2/2</td>
<td>69.8</td>
<td>89.1</td>
<td>3/3</td>
<td>71.9</td>
<td>90.2</td>
<td>4/4</td>
<td>72.9</td>
</tr>
<tr>
<td></td>
<td>DSQ [14]</td>
<td>2/2</td>
<td>70.0</td>
<td>–</td>
<td>3/3</td>
<td>72.5</td>
<td>–</td>
<td>4/4</td>
<td>72.8</td>
</tr>
<tr>
<td></td>
<td>FAQ [33]</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>4/4</td>
<td>73.3</td>
</tr>
<tr>
<td></td>
<td>QIL [23]</td>
<td>2/2</td>
<td>70.6</td>
<td>–</td>
<td>3/3</td>
<td>73.1</td>
<td>–</td>
<td>4/4</td>
<td>73.7</td>
</tr>
<tr>
<td></td>
<td>DSQ [14]</td>
<td>2/2</td>
<td>70.9</td>
<td>89.7</td>
<td>3/3</td>
<td>73.4</td>
<td>91.1</td>
<td>4/4</td>
<td>73.8</td>
</tr>
<tr>
<td></td>
<td>FAQ [33]</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>4/4</td>
<td>73.3</td>
</tr>
<tr>
<td></td>
<td>DSQ [14]</td>
<td>2/2</td>
<td>71.0</td>
<td>–</td>
<td>3/3</td>
<td>73.1</td>
<td>–</td>
<td>4/4</td>
<td>73.7</td>
</tr>
<tr>
<td></td>
<td>LQ-Nets [47]</td>
<td>2/2</td>
<td>73.7</td>
<td>91.5</td>
<td>3/3</td>
<td>75.8</td>
<td>92.7</td>
<td>4/4</td>
<td>76.7</td>
</tr>
<tr>
<td></td>
<td>DSQ [14]</td>
<td>2/2</td>
<td>73.8</td>
<td>91.4</td>
<td>3/3</td>
<td>75.4</td>
<td>92.4</td>
<td>4/4</td>
<td>76.6</td>
</tr>
<tr>
<td></td>
<td>FAQ [33]</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>DSQ [14]</td>
<td>2/2</td>
<td>75.1</td>
<td>–</td>
<td>3/3</td>
<td>76.3</td>
<td>–</td>
<td>4/4</td>
<td>76.6</td>
</tr>
<tr>
<td></td>
<td>N2UQ (Ours)</td>
<td>2/2</td>
<td>69.4</td>
<td>88.4</td>
<td>3/3</td>
<td>71.9</td>
<td>90.5</td>
<td>4/4</td>
<td>72.9</td>
</tr>
</tbody>
</table>

Table 2. Accuracy comparison with MobileNet structure.

<table>
<thead>
<tr>
<th>Network</th>
<th>Method</th>
<th>Top1 Acc (%)</th>
<th>Top5 Acc (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MobileNet-V2</td>
<td>DSQ [14]</td>
<td>64.8</td>
<td>–</td>
</tr>
<tr>
<td>(FP: 72.0)</td>
<td>LLSQ [49]</td>
<td>67.4</td>
<td>88.0</td>
</tr>
<tr>
<td></td>
<td>LCQ [46]</td>
<td>70.8</td>
<td>89.7</td>
</tr>
<tr>
<td></td>
<td>PROFIT [35]</td>
<td>71.6</td>
<td>90.4</td>
</tr>
<tr>
<td>N2UQ (Ours)</td>
<td>72.1</td>
<td>90.6</td>
<td></td>
</tr>
</tbody>
</table>

structure, N2UQ also achieves the state-of-the-art result as shown in Table 2. It is remarkable that N2UQ can compress the real-valued network to 4-bit without compromising or even improving the accuracy, indicating that deep neural network structures have redundancy and N2UQ is well-designed to fit underlying distributions and act as a good regularization to prevent overfitting as well [15].

4.3. Ablation Study

To investigate the effect of each component on the final performance, we conduct thorough ablation studies on the 2-bit quantized ResNet-18 network on ImageNet. As shown in Table 3, the proposed weight regularization and threshold learning quantizer with G-STE improve the accuracy by 1.9% and 3.0%, respectively over the baseline [50]. By using both techniques, our proposed N2UQ achieves 69.7% accuracy, narrowing the accuracy gap to the corresponding real-valued ResNet-18 to only 2.1%.

In Table 4, based on the proposed threshold learning quantizer with G-STE, we make further ablation study on weight regularization. Compared to the proposed method of calculating the regularization factor from the overall weight statistics, learning regularization factors with gradients introduces unstableness since gradients are noisy, which affects the training negatively, as shown in 4th row of Table 4. Moreover, the previous weight norm [38] designed for training real-valued network is not aware of weight quantization, but only scales the weights participating in matrix multiplication, which turns out unsuitable for quantized network, as shown in 3rd row of Table 4. In comparison, the entropy-based regularization considers the knowledge of weight dis-
4.4. Visualization

To further understand the proposed N2UQ, we illustrate the initial weight distributions and the learned parameters in quantized networks. In Fig. 3 (a), given the real-valued pre-trained weights as initialization, the baseline method [50] of using tanh function and dividing the maximum weight value to normalize the weights is likely to be biased by the extrema. Specifically, for the 2-bit case, only 6 entries in the weight filter are quantized to 0.5 and 20 entries are quantized to -0.5 in the baseline method. In contrast, the proposed entropy-based regularization method produces more evenly distributed quantized weights, which evidently enhances the amount of information that quantized weights carry. In Fig. 3 (b), the classic uniform quantization has fixed input thresholds, making it hard to adjust itself to fit different underlying distributions. In comparison, N2UQ learns the input thresholds with neural network training. As shown in the right dashed box of Fig. 3 (b), when the real-valued distribution is dense around zero and sparse in the tail, the quantization function learns smaller intervals for the dense area and larger intervals for the sparse part, which better represents the distribution and mitigates the quantization error.

Table 3. Effects of different components of N2UQ on the final performance of a 2-bit quantized ResNet-18 network.

<table>
<thead>
<tr>
<th>Method</th>
<th>Top-1 Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline [50] (our implementation)</td>
<td>65.9</td>
</tr>
<tr>
<td>+ Threshold Learning Activation Quantizer with G-STE</td>
<td>68.9</td>
</tr>
<tr>
<td>+ Entropy Preserving Weight Regularization</td>
<td>67.8</td>
</tr>
<tr>
<td>+ Both (N2UQ)</td>
<td>69.7</td>
</tr>
<tr>
<td>Corresponding real-valued network</td>
<td>71.8</td>
</tr>
</tbody>
</table>

Table 4. Comparison among different weight regularization schemes for a 2-bit quantized ResNet-18 on ImageNet, based on the proposed threshold learning quantizer with G-STE.

<table>
<thead>
<tr>
<th>Method</th>
<th>Top-1 Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Regularization</td>
<td>68.9</td>
</tr>
<tr>
<td>Weight Norm [38]</td>
<td>67.4</td>
</tr>
<tr>
<td>Learnable Scaling Factor</td>
<td>68.6</td>
</tr>
<tr>
<td>Entropy Preserving Weight Regularization</td>
<td>69.7</td>
</tr>
</tbody>
</table>

5. Conclusions

We have introduced Nonuniform-to-Uniform Quantization, abbreviated as N2UQ, for learning input thresholds in the uniform quantizer. Compared to nonuniform quantization, N2UQ enjoys the advantage of hardware-friendliness since the uniform outputs do not require additional post-processing before the bitwise operations. Meanwhile, N2UQ contains stronger representational capability than the classical uniform quantization by making the input thresholds learnable. To tackle the intractable gradient computation w.r.t. input thresholds, we derive the generalized straight-through estimator (G-STE) from the essence of stochastic quantization. Further, we propose entropy preserving weight regularization to further reduce quantization error. With these contributions, N2UQ achieves 0.8~1.7% higher accuracy than previous state-of-the-art nonuniform methods under the harsh condition of having uniform output quantized levels. Particularly, the 2-bit N2UQ ResNet-50 is just 0.6% shy of its real-valued counterpart.

Acknowledgments

This research was partially supported by ACCESS - AI Chip Center for Emerging Smart Systems, sponsored by InnoHK funding, Hong Kong SAR. We would like to thank Turing AI Computing Cloud [45] and HKUST iSING Lab for providing us computation resources on their platform.
References


