Mr.BiQ: Post-Training Non-Uniform Quantization based on Minimizing the Reconstruction Error

Yongkweon Jeon[1,*] Chungman Lee[1,*] Eulrang Cho[2,*,†] Yeonju Ro[3,*,†]
[1]Samsung Research [2]Korea University [3]The University of Texas at Austin
{dragwon.jeon, chungman.lee}@samsung.com ercho@korea.ac.kr yro@utexas.edu

Abstract

Post-training quantization compresses a neural network within few hours with only a small unlabeled calibration set. However, so far it has been only discussed and empirically demonstrated in the context of uniform quantization on convolutional neural networks. We thus propose a new post-training non-uniform quantization method, called Mr.BiQ, allowing low bit-width quantization even on Transformer models. In particular, we leverage multi-level binarization for weights while allowing activations to be represented as various data formats (e.g., INT8, bfloat16, binary-coding, and FP32). Unlike conventional methods which optimize full-precision weights first, then decompose the weights into quantization parameters, Mr.BiQ recognizes the quantization parameters (i.e., scaling factors and bit-code) as directly and jointly learnable parameters during the optimization. To verify the superiority of the proposed quantization scheme, we test Mr.BiQ on various models including convolutional neural networks and Transformer models. According to experimental results, Mr.BiQ shows significant improvement in terms of accuracy when the bit-width of weights is equal to 2: up to 5.35 p.p. improvement in CNNs, up to 4.23 p.p. improvement in Vision Transformers, and up to 3.37 point improvement in Transformers for NLP.

1. Introduction

As deep neural networks scale up rapidly to improve model accuracy, it becomes more challenging not only to reduce memory footprint but also to achieve low end-to-end latency in resource-constrained environments. To mitigate such challenges, many researchers have made a considerable amount of effort on advancing model compression techniques such as pruning [12, 24, 28], low-rank approximation [23, 37], knowledge distillation [9, 14], and quantization [4, 7, 11, 22, 29, 35, 40].

Quantization, among such compression techniques, is particularly effective in terms of reducing the model size and accelerating inference even on commodity hardware. By representing each parameter with lower bit-width, quantization can reduce the model size, and hence alleviate memory bottleneck issues. In addition, since quantization maintains dense formats of tensors, parallelism can be fully exploited without irregular data structures. These irregularities, induced by certain compression methods such as pruning, require much support for specialized hardware designs. As such, quantization can be implemented efficiently without a lot of support for specialized hardware designs and makes it practical to deploy quantized models on various hardware form factors.

By and large, quantization can be classified into two categories: quantization-aware training (QAT) [7, 47, 49] and post-training quantization (PTQ) [22, 29, 40, 44]. In general, QAT yields higher accuracy than PTQ because it directly aims to minimize the loss of the network. Nonetheless, QAT relies on the entire training dataset and requires a thorough hyperparameter search, which leads to the same amount of training time and overhead as the full-precision models. On the other hand, PTQ allows the quantization of pre-trained models with only a small calibration dataset or without any dataset, which enables us to compress models even when data access is restricted due to various reasons including privacy concerns. PTQ also does not require a comprehensive understanding of the model, and thus has been attracting attention recently.

Early works on post-training quantization are concentrated on minimizing the quantization error which is defined as the mean squared error between the original weights and the quantized weights (i.e., $\min \mathbb{E}[(\mathbf{w} - \mathbf{w}^q)^2]$). However, recent works [22, 29] tend to focus on minimizing the reconstruction error (i.e., $\min \mathbb{E}[(\mathbf{Wx} - \mathbf{W}^q\mathbf{x})^2]$) that can be derived from the second-order approximation of the loss by the Taylor series. Although minimizing the reconstruction error for PTQ is proven to be effective, PTQ has been discussed and empirically demonstrated only in the context of uniform quantization on convolutional neural networks

*Equal contribution. Correspondence to: *dragwon.jeon@samsung.com*
†This work was done when they were working at Samsung Research.

(CNNs). We also notice that the previous works study mapping scheme for weights and step-size learning for activations separately, and thus we may miss the opportunity to investigate any synergistic effects when those two are optimized jointly.

In this work, we propose Mr.BiQ[1], a post-training non-uniform quantization method that follows the form of multi-level binary (or binary code) while minimizing the reconstruction error. We extend general principles of post-training quantization to non-uniform quantization and demonstrate our proposed method on recently proposed Transformer models for vision tasks and natural language processing (NLP) tasks in addition to CNNs. In particular, we introduce a novel approach that enables optimization of quantization parameters (i.e., both binary-coding and the multi-level step size) jointly. Since the search space of uniform quantization is a subset of that of non-uniform quantization, our work provides a comprehensive search of quantized models and pushes the limit of quantization to significantly low bit-width (i.e., W2A4[2] or W2A8) with negligible accuracy drop.

Note that, in terms of acceleration, multi-level binary as the form of quantized weights can be combined with various formats for activations (e.g., INT, bfloat16, and FP32) as well as binary code. When both weights and activations are binary-coded, matrix multiplications can be accelerated by *xnor-popcnt* operations [4]. If activations are maintained to be full-precision, we can exploit a dedicated computational kernel, *BiQGEMM* [16], available for commodity hardware. Furthermore, with fixed-point activations, the computation mainly requires integer adders which is effective in terms of chip area and power consumption [1, 8].

To sum up, we propose a new method for post-training multi-level binary quantization. Unlike the conventional approaches, Mr.BiQ recognizes both quantization parameters as learnable parameters and obtains quantized weights by multiplying them in a bottom-up fashion. Overall, we achieve state-of-the-art accuracy not only in CNNs but also in Transformer models: up to 5.35 p.p. improvement in CNNs (RegNetX-3.2GF [33]-W2A4), up to 4.23 p.p. improvement in Vision Transformers (DeiT-S [42]-W2A8), and up to 3.37 point improvement in Transformers for NLP (DistilBERT [39]-SQUAD v1.1-W2A8).

2. Preliminaries

In multi-level binary (or binary-coding) quantization (BiQ), multiple bits share the scaling factor $\alpha_i \in \mathbb{R}$ while each bit in the binary codes $\mathbf{b}_i \in \{-1, 1\}^n$ $(1 \leq i \leq q)$ determines the sign of the corresponding scaling factor. And a linear combination of $\{\alpha_i\}_{i=1}^q$ and $\{\mathbf{b}_i\}_{i=1}^q$ produces quantized weights $\mathbf{w}^q$. Thus, we have

$$Q(\mathbf{w}) = \mathbf{w}^q \in \mathcal{Q}^{BiQ} = \left\{ \sum^q x_i | x_i \in \{+\alpha_i, -\alpha_i\} \right\}^n, \tag{1}$$

where q is quantization bit-width.

AMQ [47] and LQ-Nets [49] proposed BiQ methods in a quantization-aware training (QAT) manner, where the quantizer Q decomposes $\mathbf{w} \in \mathbb{R}^n$ into the scale factors $\{\alpha_i\}_{i=1}^q$ and the binary coding vectors $\{\mathbf{b}_i\}_{i=1}^q$, such that $\mathbf{w}$ is approximated to be $\mathbf{w}^q = \sum_i^q \alpha_i^* \mathbf{b}_i^*$ as a result of minimizing the mean squared error:

$$\alpha_i^*, \mathbf{b}_i^* = \arg\min_{\alpha_i, \mathbf{b}_i} ||\mathbf{w} - \sum_{i=1}^q \alpha_i \mathbf{b}_i||^2. \tag{2}$$

Algorithm 1 Alternating Multi-bit Quantization [47]

```
Input: Full-precision weight $\mathbf{w} \in \mathbb{R}^n$, bit-width $q$, alternating cycles
    (AC) $T$
Output: $\alpha_i \in \mathbb{R}, \mathbf{b}_i \in \{-1,1\}^n,\ 1 \leq i \leq q$
1: procedure DECOMPOSITION($\mathbf{w}$, $q$, $T$)
2:     $\{\alpha_i, \mathbf{b}_i\}_{i=1}^q \leftarrow$ greedy_method ($\mathbf{w}$)          ▷ See Eq. (3)
3:     for $t \leftarrow 1$ to $T$ do
4:         $\{\alpha_i\}_{i=1}^q \leftarrow$ least_squares ($\mathbf{B}$, $\mathbf{w}$)        ▷ See Eq. (4)
5:         $\{\mathbf{b}_i\}_{i=1}^q \leftarrow$ binary_search ($\alpha_1, ..., \alpha_q$, $\mathbf{w}$)
```

Algorithm 1 describes a method to reduce the mean squared error as in Eq. (2). When the residue $\mathbf{r}_i$ denotes $\mathbf{w} - \sum_{k=0}^{i-1} \alpha_k \mathbf{b}_k$, for $i \geq 1$, we can obtain $\mathbf{b}_i$ and α_i sequentially as

$$\mathbf{b}_i = \text{sign}(\mathbf{r}_i) \text{ and } \alpha_i = \frac{\mathbf{r}_i^\top \mathbf{b}_i}{n}, \tag{3}$$

for $1 \leq i \leq q$ and $\alpha_0 = 0$, which is the greedy method (*Line 2* in Algorithm 1) [10]. Furthermore, scaling factors $\{\alpha_i\}_{i=1}^q$ can be refined with ordinary least squares (*Line 4*) [10]:

$$[\alpha_1, ..., \alpha_q] = ((\mathbf{B}^\top \mathbf{B})^{-1} \mathbf{B}^\top \mathbf{w})^\top, \tag{4}$$

where $\mathbf{B} = [\mathbf{b}_1, ..., \mathbf{b}_q] \in \{-1, 1\}^{n \times q}$. Then, the binary coding vectors $\{\mathbf{b}_i\}_{i=1}^q$ can be optimized by binary search, which re-calibrates the binary codes such that each weight is assigned to the nearest neighbor in $\{\sum^q x_i | x_i \in \{+\alpha_i, -\alpha_i\}\}$ (*Line 5*). This alternating process can be performed iteratively (*Line 3–5*) to further minimize the quantization error as in Eq. (2) [47].

Existing methods perform Algorithm 1 every step during training, and update full-precision weights $\mathbf{w}$ using straight through estimator (STE) ($\frac{\partial L}{\partial \mathbf{w}} = \frac{\partial L}{\partial \mathbf{w}^q}$) [2] (See the left of Figure 1).

[1]Post-training Multi-Level Binary Quantization based on Minimizing the Reconstruction error.
[2]2-bit Weights, 4-bit Activations

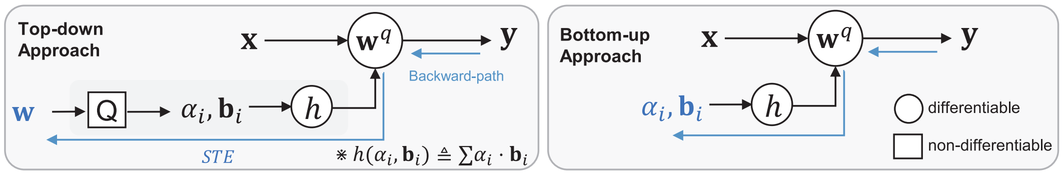


Figure 1. Comparing the proposed method (bottom-up) with conventional methods (top-down).

3. Mr.BiQ

We propose a novel post-training multi-level binary quantization (BiQ) method with a low computational cost. Unlike the conventional methods [47, 49], which decompose $\mathbf{w}$ into $\{\alpha_i\}_{i=1}^q$ and $\{\mathbf{b}_i\}_{i=1}^q$ (i.e., a top-down approach) with a least squares solution, our method optimizes scale factors and bit-code as learnable parameters that are to be combined to yield quantized weights $\mathbf{w}^q$ (i.e., a bottom-up approach, see Figure 1).

Given a pre-trained model, parameters $\{\alpha_i\}_{i=1}^q$ and $\{\mathbf{b}_i\}_{i=1}^q$ are initialized as illustrated in Algorithm 1 and then followed by our quantization techniques which aim to minimize the block-wise reconstruction error within a few epochs. A block can be defined as one or more consecutive layers and is usually set to a residual block. The objective function per-block to be optimized during our proposed quantization process is as follows:

$$\arg\min_{\mathbf{w}^q} \Delta\mathbf{z}^T \cdot \mathbf{H}^{(z)} \cdot \Delta\mathbf{z}, \tag{5}$$

where $\Delta\mathbf{z}$ is the perturbation of the block outputs and $\mathbf{H}^{(z)}$ is the Hessian matrix of the block outputs. However, computing the Hessian requires high computational cost, so we approximate it as $c \times \mathbb{I}$, where c is a constant and $\mathbb{I}$ is the identity matrix. Thus, the objective function becomes as follows:

$$\arg\min_{\mathbf{w}^q} \left|\left|\mathbf{z} - \mathbf{z}^q\right|\right|_F^2, \tag{6}$$

where $\mathbf{z}$ and $\mathbf{z}^q$ are the block outputs of the pre-trained full-precision network and the quantized network, respectively (i.e., $\Delta\mathbf{z} = \mathbf{z} - \mathbf{z}^q$). Thus, we can consider this process a sort of knowledge distillation [9, 14] in which the pre-trained model corresponds to a teacher while the quantized model becomes a student. Note that, we also used the empirical Fisher instead of the Hessian as done in BRECQ [22], but there were no significant differences in the results between utilizing the Fisher or not. In fact, the empirical Fisher approximation may negatively affect the results [19].

To explain a naïve approach optimizing quantization parameters in a bottom-up manner, we suppose that a quantized weight w^q is equals to $\alpha_1 b_1 + \alpha_2 b_2 + \alpha_3 b_3$, the order of α_i is given as $\alpha_1 > \alpha_2 > \alpha_3 \geq 0$, and g denotes $\frac{\partial\mathcal{L}}{\partial w^q}$. Then, we can compute the gradient with respect to b_i as $g \cdot \alpha_i$, which means the gradient to be accumulated into b_1 becomes the largest due to the magnitude of α_1. Such a large gradient for b_1, however, is not desirable because the change in b_3 (of the lower bit-position) should be more frequent than that in b_1 (of the higher bit-position) if we target smoothness in the amount of the overall change in w^q when the corresponding bit-code is updated. To be more specific, we would like to mutate the bit-code gradually from the lower bit-position (e.g., $111 \rightarrow 110$), which would be also observed in the changes of full-precision weights, not radically (e.g., $111 \rightarrow 011$). Such radical change may arise from a non-differentiable step function of which the output is binary parameter b_i. In practice, binary parameters could not converge during the optimization by the naïve approach.

To fundamentally address this issue, we reformulate full-precision weights $\mathbf{w}$ using the initial $\{\alpha_i\}_{i=1}^q$ and $\{\mathbf{b}_i\}_{i=1}^q$ before conducting post-training quantization, where $\{\mathbf{b}_i\}_{i=1}^q$ is translated to *softbit* vector, a differentiable form.

For brevity, we define several functions as follows:

- `base(`w`)` := the smaller one of the 2-nearest neighbors (2-nn) in $\mathcal{C}$ to w, where $\mathcal{C} = \{\sum^q x_i | x_i \in \{+\alpha_i, -\alpha_i\}\}$
- `scale(`w`)` := the distance between 2-nn in $\mathcal{C}$ to w
- `m(`w`)` := the distance between w and base(w)
- `sb=softbit(`w`)` := $clip(\frac{m(w)}{scale(w)}, 0, 1)$

By exploiting such functions, we can reformulate w as follows:

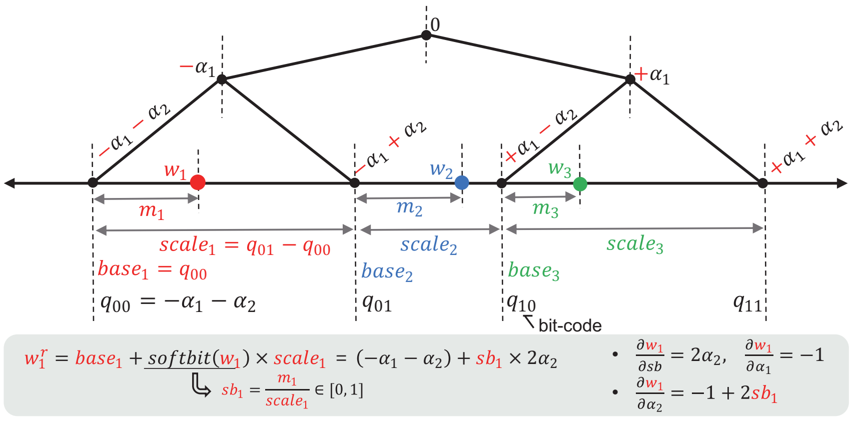
$$w \approx w^r = \texttt{base}(w) + \texttt{softbit}(w) \times \texttt{scale}(w) \tag{7}$$

which is identical to full-precision w except outliers of $|w| > \sum \alpha_i$. Figure 2 provides an example of reformulation when quantization bit-width is equal to 2. Note that `softbit(`w`)` can be considered the result of min-max scaling of w's for each section.

To encourage *softbit* to converge towards either 0 or 1, we use an adaptive rounding method proposed in AdaRound [29] where rectified sigmoid [26] is utilized as follows:

$$h(v) = clip(S(v)(\zeta - \gamma) + \gamma, 0, 1), \tag{8}$$

where $S(\cdot)$ denotes the sigmoid function, and ζ and γ are stretch parameters. We initialize v as $h^{-1}(\texttt{softbit}(w))$



0
$-\alpha_1$
$+\alpha_1$
$-\alpha_1 - \alpha_2$
$-\alpha_1 + \alpha_2$
$+\alpha_1 - \alpha_2$
$+\alpha_1 + \alpha_2$
w_1
w_2
w_3
m_1
m_2
m_3
$scale_1 = q_{01} - q_{00}$
$scale_2$
$scale_3$
$base_1 = q_{00}$
$base_2$
$base_3$
$q_{00} = -\alpha_1 - \alpha_2$
q_{01}
q_{10}
q_{11}
bit-code
$w_1^r = base_1 + \underline{softbit(w_1)} \times scale_1 = (-\alpha_1 - \alpha_2) + sb_1 \times 2\alpha_2$
$sb_1 = \frac{m_1}{scale_1} \in [0, 1]$
$\frac{\partial w_1}{\partial sb} = 2\alpha_2, \quad \frac{\partial w_1}{\partial \alpha_1} = -1$
$\frac{\partial w_1}{\partial \alpha_2} = -1 + 2sb_1$

Figure 2. Reformulation. When quantization bit-width is 2, there are four quantization points ($q_{ij} \in \mathcal{Q}^{BiQ}$, $i, j \in \{0, 1\}$) and three sections ($[q_{00}, q_{01}]$, $[q_{01}, q_{10}]$, and $[q_{10}, q_{11}]$), where subscripts denote the bit-code corresponding to the quantization point. Weights in each section share *base* and *scale*, but each weight has its own m. Reformulation allows *softbit* (sb) to be a real number between 0 and 1, and sb is encouraged to be either 0 or 1 during the optimization. Provided that sb of w_1 close to q_{00} is converged to 1 after the optimization, the bit-code of w_1 is changed from '00' to '01'.

and add the regularization term to the objective to enforce $h(v)$ to be either 0 or 1, which can be expressed as

$$\arg\min_{\mathbf{w}^q} \left|\left|\mathbf{z} - \mathbf{z}^q\right|\right|_F^2 + \lambda \sum_i 1 - \left|2h(v_i) - 1\right|^\beta. \qquad (9)$$

In Eq. (9), β is annealed during the optimization such that $h(v)$ gets closer to either 0 or 1 [29]. By reformulating w as in Eq. (7), we can compute the gradient of $\{\alpha_i\}_{i=1}^q$ and sb as does in the computation of gradient of a floating-point number. As shown in Figure 2, the scaling factors, the binary code, and *softbit* are involved when back-propagating gradients. Depending on the optimized sb, each element of $\mathbf{w}$ is assigned to one of the two nearest neighbors in $\{\sum^q x_i | x_i \in \{+\alpha_i, -\alpha_i\}\}$ which is revised by optimized $\{\alpha_i\}_{i=1}^q$.

Algorithm 2 summarizes how Mr.BiQ optimizes a block by minimizing the reconstruction error. Mr.BiQ only needs a small unlabeled calibration set $\mathcal{D}_c$ sampled from the training data set $\mathcal{D}_t$ (i.e., $\mathcal{D}_c \subset \mathcal{D}_t$ and $|\mathcal{D}_c| \ll |\mathcal{D}_t|$). After initialization of $\{\alpha_i\}_{i=1}^q$ and $\{\mathbf{b}_i\}_{i=1}^q$ (*Line 2* in Algorithm 2), Mr.BiQ reformulates $\mathbf{w}_b$ using initial $\{\alpha_i\}_{i=1}^q$ and the binary coding vectors is translated into *softbit* vector $\mathbf{sb} \in \mathbb{R}^n$ (*Line 3*). Then, Mr.BiQ optimizes both $\{\alpha_i\}_{i=1}^q$ and $\mathbf{sb}$ to minimize the reconstruction error (*Line 4–10*). After the optimization, $\mathcal{Q}^{BiQ}$ is updated based on the optimized $\{\alpha_i\}_{i=1}^q$ (*Line 11*) and each element of *softbit* vector $\mathbf{sb}$ is restored to the nearest bit-code (*Line 12*). The optimization is sequentially performed from the block closest to the input layer. To compensate accumulation of quantization error, the student block takes the quantized activations passed through the previous quantized blocks as the input (i.e., $\mathbf{a}^q$ in *Line 5*) which is different from the input of the teacher block (i.e., $\mathbf{a}$ in *Line 7*). The notation f_b^q indicates the quantized output of the block f_b (i.e., f_b^q=$Q(f_b)$, where Q is any quantizer.). In Section 4, we utilize LSQ [7] as the activation quantizer and the step size of activations can be optimized along with $\mathbf{w}_b^r$ in *Line 10*.

Algorithm 2 Mr.BiQ

```
Input: Full-precision weights $\mathbf{w}_b \in \mathbb{R}^n$ in a block $f_b$, bit-width $q$,
       Sampled data set $\mathcal{D}_c$
Output: Quantized block $f_b(\cdot; \mathbf{w}_b^q)$
 1: procedure MR.BIQ($\mathbf{w}_b$, $q$, $\mathcal{D}_c$)
 2:     $\{\alpha_i, \mathbf{b}_i\}_{i=1}^q \leftarrow$ Decomposition($\mathbf{w}_b, q, 50$)   ▷ Algorithm 1
 3:     $\mathbf{w}_b^r \leftarrow$ reformulation($\mathbf{w}_b, \alpha_1, \ldots, \alpha_q$)   ▷ See Eq.(7)
 4:     for each input $\mathbf{x} \in \mathcal{D}_c$ do
 5:         $\mathbf{a}^q \leftarrow f_{b-1}^q(\ldots f_2^q(f_1^q(\mathbf{x}; \mathbf{w}_1^q); \mathbf{w}_2^q) \ldots; \mathbf{w}_{b-1}^q)$
 6:         $\mathbf{z}^q \leftarrow f_b(\mathbf{a}^q; \mathbf{w}_b^r)$   ▷ Student block
 7:         $\mathbf{a} \leftarrow f_{b-1}(\ldots f_2(f_1(\mathbf{x}; \mathbf{w}_1); \mathbf{w}_2) \ldots; \mathbf{w}_{b-1})$
 8:         $\mathbf{z} \leftarrow f_b(\mathbf{a}; \mathbf{w}_b)$   ▷ Teacher block
 9:         $\mathcal{L} \leftarrow ||\mathbf{z}^q - \mathbf{z}||_F^2$   ▷ See Eq.(6) and (9)
10:         $\mathcal{L}$.backward()   ▷ Update $\mathbf{w}_b^r$ and the step size of $\mathbf{a}^q$
11:     $\mathcal{Q}^{BiQ} \leftarrow$ update($\alpha_1, \ldots, \alpha_q$)
12:     $\{\mathbf{b}_i\}_{i=1}^q \leftarrow$ Restore($\mathbf{sb}$)
```

4. Experimental Results

We evaluate our proposed method by testing it on recently proposed Transformer models for vision tasks and natural language processing (NLP) tasks in addition to convolutional neural networks (CNNs). With randomly sampled 1K images from ImageNet (ILSVRC12) [36], we quantize various CNNs including ResNet [13], MobileNetV2 [38], RegNet [33] and MnasNet [41]. Also, we quantize transformer models for vision tasks such as ViT [6] and DeiT [42]. As for the NLP tasks, we evaluate MNLI-matched, MRPC (from GLUE benchmark [43]) and SQuAD 1.1 [34] for on both BERT [5] and Distil-

Table 1. Ablation Study (top-1 accuracy (%))

Methods	#Bits (W/A)	ResNet-18	ResNet-50	MobileNetV2	RegNetX-600MF	RegNetX-3.2GF	MnasNet
Full Prec.	32/32	71.08	77.00	72.49	73.71	78.36	76.68
Alpha-only	2/32	37.20	48.21	27.33	35.64	53.34	33.45
Bit-only		66.83	72.12	58.83	65.45	73.06	60.12
Mr.BiQ		**67.92**	**73.10**	**62.96**	**67.24**	**74.89**	**70.12**
Alpha-only	3/32	65.23	72.61	65.79	68.09	74.52	71.84
Bit-only		69.71	75.52	69.26	71.45	76.91	74.17
Mr.BiQ		**70.17**	**75.83**	**70.57**	**72.11**	**77.63**	**75.15**
Alpha-only	4/32	68.88	75.31	70.66	71.88	76.97	75.45
Bit-only		70.41	76.19	71.28	72.79	77.65	75.97
Mr.BiQ		**70.76**	**76.42**	**72.11**	**73.15**	**78.24**	**76.17**

BERT [39].

Throughout this section, we mainly compare Mr.BiQ with BRECQ [22], a post-training quantization framework, which has shown the state-of-the-art result among integer-based PTQ approaches while it has shown comparable results to QAT approaches. Such a framework includes AdaRound [29] for weight quantization and *learned step size quantization* (LSQ) [7] for activation quantization. While BRECQ selects asymmetric quantization for both weights and activations, Mr.BiQ utilizes symmetric quantization for both of them. In all experiments, weights and activations are quantized channel-wise and layer-wise, respectively. We also quantize activation uniformly using LSQ [7] as did in BRECQ. Note that activation quantization optimizes only the step size. Based on the step size optimized offline, activations can be allocated to the nearest quantization point at inference by rounding off.

For all CNNs evaluations, we quantize the weights of the first and the last layer into 8 bits as did in [22]. For Transformer models (i.e., ViT [6], DeiT [42], BERT [5], and DistilBERT [39]), we do not quantize the input of the softmax layer and the normalization layer as did in [25, 32, 50]. We measure the accuracy of 20 runs with randomly sampled datasets, and then the average values and the standard deviations are obtained. Moreover, we evaluate the *t*-test to identify whether the figures are statistically different or not. Please refer to Appendix for more details on other specific experimental setups.

4.1. Ablation Study

In Table 1, we first compare three different schemes to optimize quantization parameters: scaling factors (labeled ”Alpha-only”), binary codes (”Bit-only”), and both of them (”Mr.BiQ”). To minimize the block-wise reconstruction error, each block is optimized for 20K steps with a batch size of 32, except in the case of ”Alpha-only” optimization for which 1k-step is enough to converge successfully. From our evaluation, learning only scaling factors (”Alpha-only”) is enough to achieve nearly full accuracy at 4-bit quantization while the accuracy reaches a saturation point at 3-bit in most schemes. Thus, ”Alpha-only” may be a good choice if we use more than 3-bit for quantization since it is simple and takes less time than others. At ultra-low bit-width (i.e., 2-bit), however, improvement in accuracy is severely bound to the initial bit-code, which implies that the bit-flipping is required in order to further improve the accuracy of the models. Indeed, ”Bit-only” shows a great improvement in accuracy and reaches nearly the same accuracy as prior works; but there is still room for improvement of the accuracy by optimizing both the multi-level step size and the bit-code together. Mr.BiQ is such an algorithm to allow both of them to be optimized jointly and presents the best accuracy among the three quantization schemes for multi-level binary quantization.

4.2. Comparison on Convolutional Neural Nets

Weight-Only Quantized Models As the baseline, we test ”Data-free” quantization, a naïve approach, which minimizes the quantization error without any retraining, fine-tuning, or calibration. In other words, ”Data-free” performs Algorithm 1 once from the pre-trained model. We also implement existing methods [47, 49] in a post-training quantization manner (labeled ”Top-down”). By setting the objective as minimizing the block-wise reconstruction error, ”Top-down” decomposes weights into $\{\alpha_i\}_{i=1}^q$ and $\{\mathbf{b}_i\}_{i=1}^q$, and updates weights using STE during the optimization. In addition, the results of BRECQ are included in the comparison. As shown in Table 2, Mr.BiQ outperforms other methods especially when weights are quantized into 2-bit. The results also indicate that our bottom-up approach (i.e., Mr.BiQ) is clearly better than the conventional BiQ method (”Top-down”) at least in post-training. In Table 2, BRECQ uses the Fisher information matrix while Mr.BiQ does not.

Table 2. The evaluation on Weight-Only Quantized Models (top-1 accuracy (%))

Methods		#Bits (W/A)	ResNet-18	ResNet-50	MobileNetV2	RegNetX-600MF	RegNetX-3.2GF	MnasNet
Full Prec.		32/32	71.08	77.00	72.49	73.71	78.36	76.68
BiQ	Data-free	2/32	0.16	0.12	0.07	0.11	0.11	0.11
	Top-down		63.45	68.67	50.35	58.32	67.87	57.47
	Mr.BiQ		**67.92**$_{\pm0.11}$	**73.10**$_{\pm0.11}$	**62.96**$_{\pm0.18}$	**67.24**$_{\pm0.10}$	**74.89**$_{\pm0.10}$	**70.12**$_{\pm0.17}$
INT	BRECQ* [22]		66.30$_{\pm0.12}$	72.40$_{\pm0.12}$	59.67$_{\pm0.13}$	65.83$_{\pm0.13}$	73.88$_{\pm0.14}$	67.13$_{\pm0.13}$
BiQ	Data-free	3/32	5.50	15.75	0.53	4.95	20.65	5.52
	Top-down		68.41	74.19	66.67	69.03	75.21	72.25
	Mr.BiQ		70.17$_{\pm0.08}$	75.83$_{\pm0.07}$	70.57$_{\pm0.10}$	72.11$_{\pm0.10}$	77.63$_{\pm0.08}$	75.15$_{\pm0.09}$
INT	BRECQ* [22]		69.81$_{\pm0.05}$	75.61$_{\pm0.09}$	69.50$_{\pm0.12}$	71.48$_{\pm0.07}$	77.22$_{\pm0.04}$	74.58$_{\pm0.08}$
BiQ	Data-free	4/32	55.69	58.19	29.32	36.67	66.58	54.47
	Top-down		69.70	75.63	70.15	71.30	76.77	75.21
	Mr.BiQ		70.76$_{\pm0.06}$	76.42$_{\pm0.06}$	72.11$_{\pm0.07}$	73.15$_{\pm0.07}$	78.24$_{\pm0.05}$	76.17$_{\pm0.06}$
INT	BRECQ* [22]		70.70$_{\pm0.07}$	76.29$_{\pm0.04}$	71.66$_{\pm0.04}$	73.02$_{\pm0.09}$	78.04$_{\pm0.04}$	76.00$_{\pm0.02}$

[*] The figures are taken from [22].

Table 3. The evaluation on Fully-Quantized Models (top-1 accuracy(%))

	#Bits (W/A)	ResNet-18	ResNet-50	MobileNetV2	RegNetX-600MF	RegNetX-3.2GF	MnasNet
Full Prec.	32/32	71.08	77.00	72.49	73.71	78.36	76.68
Mr.BiQ	2/4	**66.61**$_{\pm0.10}$	**71.38**$_{\pm0.15}$	**57.27**$_{\pm0.22}$	**64.15**$_{\pm0.12}$	**72.50**$_{\pm0.11}$	**65.48**$_{\pm0.18}$
BRECQ* [22]		64.80$_{\pm0.08}$	70.29$_{\pm0.23}$	53.34$_{\pm0.15}$	59.31$_{\pm0.49}$	67.15$_{\pm0.11}$	63.01$_{\pm0.35}$
AdaQuant* [15]		0.21	0.12	0.10	-	-	-
Mr.BiQ	4/4	**69.68**$_{\pm0.08}$	**75.17**$_{\pm0.08}$	**68.97**$_{\pm0.09}$	**71.18**$_{\pm0.09}$	**76.65**$_{\pm0.10}$	**73.39**$_{\pm0.13}$
BRECQ* [22]		69.60$_{\pm0.04}$	75.05$_{\pm0.09}$	66.57$_{\pm0.67}$	68.33$_{\pm0.28}$	74.21$_{\pm0.19}$	73.56$_{\pm0.24}$
Bit-Split* [44]		67.56	73.71	-	-	-	-
AdaQuant* [15]		67.5	73.7	34.95	-	-	-
ZeroQ* [3]		21.71	2.94	26.24	28.54	12.24	3.89
LAPQ* [31]		60.3	70.0	49.7	57.71	55.89	65.32

[*] The figures are taken from [22].

Fully-Quantized Models Table 3 evaluates the accuracy of the models quantized by various approaches, when both weights and activations are quantized. Likewise, the results show that Mr.BiQ outperforms other methods. Note that, even if there is no accuracy drop due to activation quantization in "Top-down" (if so, the results of "Top-down" after activation quantization is the same as "Top-down" in Table 2), the performance is inferior to Mr.BiQ of Table 3 which includes activation quantization.

Table 4. The *t*-tests between Mr.BiQ and BRECQ [22] (*p*-value)

#Bits (W/A)	*t*-test	ResNet-18	ResNet-50	RegNetX -600MF	RegNetX -3.2GF
2/4	Student	1.14E-33	1.46E-29	3.25E-28	9.84E-44
	Welch	7.64E-35	4.25E-31	4.74E-34	9.89E-48
4/4	Student	2.98E-19	9.91E-23	1.17E-31	2.66E-42
	Welch	2.37E-19	9.13E-23	6.28E-32	3.98E-43

We also perform the *t*-test to identify whether the figures listed in Table 3 have statistical significance. To conduct the test, we reproduce the results of BRECQ using their open-source code[3]. Because we use the different sampled datasets in each experiment, we evaluate an unpaired *t*-test such as Student's and Welch's *t*-test [45]. When the sample means from evaluations of Mr.BiQ and BRECQ are denoted by $\tilde{X}1$ and $\tilde{X}2$, we set the null hypothesis and the alternative hypothesis as $H_0 : \tilde{X}1 = \tilde{X}2$ and $H_a : \tilde{X}1 > \tilde{X}2$, respectively. Table 4 shows the *p*-values of each evaluation when quantization bit-width is W2A4 or W4A4. The results indicate that the null hypothesis is rejected in favor of the alternative hypothesis. In other words, the performance gap between Mr.BiQ and BRECQ is not likely to have occurred by chance.

4.3. Comparison on Transformer Models

We measure the accuracy on Transformer models not only for vision tasks (such as ViT-Base (ViT-B), ViT-Large (ViT-L), DeiT-Small (DeiT-S), and DeiT-Base (DeiT-B)) but also for NLP tasks (such as BERT and DistilBERT).

[3]https://github.com/yhhhli/BRECQ

Table 5. Comparison on Vision Transformers (top-1 accuracy(%))

	W/A	ViT-B	ViT-L	DeiT-S	DeiT-B
Full Prec.		78.04	76.93	79.72	81.74
Data-free		55.70	57.82	13.99	37.72
Mr.BiQ	2/8	**75.46**$\pm$0.11	**75.86**$\pm$0.12	**73.15**$\pm$0.16	**78.97**$\pm$0.07
BRECQ[†] [22]		71.52$\pm$2.76	72.45$\pm$1.04	68.92$\pm$0.15	76.91$\pm$0.13
Data-free		74.73	74.57	67.01	75.43
Mr.BiQ	3/8	77.41$\pm$0.06	76.73$\pm$0.07	78.09$\pm$0.10	81.10$\pm$0.08
BRECQ[†] [22]		75.56$\pm$1.90	76.08$\pm$0.44	77.46$\pm$0.09	80.76$\pm$0.05
Data-free		76.98	76.15	75.72	79.49
Mr.BiQ	4/8	77.76$\pm$0.05	76.84$\pm$0.05	79.07$\pm$0.05	81.45$\pm$0.04
BRECQ[†] [22]		76.49$\pm$1.70	76.58$\pm$0.06	78.96$\pm$0.06	81.43$\pm$0.05
Data-free		74.28	74.69	71.58	78.70
Mr.BiQ		77.34$\pm$0.07	76.33$\pm$0.08	76.82$\pm$0.10	80.86$\pm$0.07
BRECQ[†] [22]		77.34$\pm$0.06	75.99$\pm$0.04	77.88$\pm$0.06	81.03$\pm$0.06
Percentile[a] [21]	6/6	71.58	71.48	70.49	73.99
PTQVT[a,b] [25]		75.26	75.46	75.1	77.47
Bit−Split[a] [44]		-	-	74.04	76.39
EasyQuant[a] [46]		-	-	73.26	75.86

[a] The figures are taken from [25]. The baselines are 77.91, 76.53, 79.8 and 81.8 for ViT-B, ViT-L, DeiT-S and DeiT-B, respectively.
[b] It represents the results for mixed-precision.
[†] We apply BRECQ to Transformer models for vision tasks based on the open-source code.

Although BRECQ did not evaluate Transformer models, we implemented block-wise adaptive rounding based on the open-source code, as an integer-based approach, for comparison as shown in Table 5 and 6. For quantization, we define a block as an encoder layer. In other words, we perform the optimization for each encoder consisting of several layers.

Similar to results in CNNs, Mr.BiQ is especially effective when the weights are quantized to 2-bit. To the best of our knowledge, this work is the first to present a reasonable accuracy on Transformer models with 2-bit weights after post-training quantization. Interestingly, when parameters of BERT and DistilBERT are optimized with small datasets such as MRPC, we observed the accuracy is improved over the full-precision baseline, which is also reported in other works (e.g., in [18]). This observation may indicated that quantization has a regularization effect. Compared to QAT approaches (such as Q8BERT [48], TerneryBERT [50], and KDLSQ-BERT [17]), Mr.BiQ shows prominent results particularly in low bit-width (i.e., 2-bit). Note that QAT performs end-to-end back-propagation with the entire dataset. In terms of the amount of dataset and time required for optimization, Mr.BiQ shows competitive accuracy (or score) even compared to QAT approaches.

5. Related Works

5.1. Quantization Strategies

Based on the types of constraints of dataset during the optimization, quantization can be categorized into two approaches: quantization-aware training (QAT) [7,47,49] and post-training quantization (PTQ) [22,29,30,40,44].

QAT demands whole training data set with associated labels and requires a thorough hyperparameter search procedure. Assume that the output loss L of a network f is measured by $L = f(\mathbf{x}, \mathbf{y}, \mathbf{w})$, where $(\mathbf{x}, \mathbf{y})$ indicates pairs of data and labels, and $\mathbf{w}$ denotes full-precision weights. To let the loss be aware of quantization error, QAT measures the loss by feeding quantized weights $\mathbf{w}^q$ instead of $\mathbf{w}$ (i.e., $L = f(\mathbf{x}, \mathbf{y}, \mathbf{w}^q \leftarrow Q(\mathbf{w}))$, where Q is a quantizer) and updates full-precision weights $\mathbf{w}$ using straight through estimator (STE) (i.e., $\frac{\partial L}{\partial \mathbf{w}} = \frac{\partial L}{\partial \mathbf{w}^q}$) [2] which means that $\mathbf{w}_{t+1} = \mathbf{w}_t - \gamma \frac{\partial \mathcal{L}}{\partial \mathbf{w}_t^q}$, where γ denotes the learning rate.

PTQ, which is our choice in this work, compresses networks using a pre-trained model with a small amount of unlabeled data set for calibration. Suppose that $\mathbf{y}_l$ are activations of intermediate layer f_l (i.e., $\mathbf{y}_l = f_l(\mathbf{x}, \mathbf{w})$). Then, intermediate activations of quantized networks can be expressed as $\mathbf{y}_l^q = f_l^q(\mathbf{x}, \mathbf{w}^q)$. PTQ utilizes $\mathbf{y}_l$ as soft labels and trains sub-networks to minimize $||\mathbf{y}_l - \mathbf{y}_l^q||^2$, which is called minimizing the reconstruction error (MRE). This process can be considered a sort of knowledge distillation in which a pre-trained model corresponds to a teacher while a model to be quantized becomes a student.

DFQ, one of the PTQ, refers to quantizing pre-trained models without any data. Due to the lack of prior information on input data, DFQ maps the weights to fixed points so as to minimize mean squared error (MMSE). By estimating the input data distribution from the learned parameters, one could improve the compression ratio or the accuracy of DFQ further [30].

5.2. Post-Training Quantization

Prior works demonstrated noticeable accuracy improvement when minimizing the reconstruction error (MRE) is performed for post-training quantization [22, 29]. One of the reasons for such outstanding quality is that MRE considers in-domain distribution from input data $\mathbf{x}$ while optimizing where to map each weight [40]. To illustrate specific steps of MRE, on the other hand, AdaRound [29] sets the objective function as $\arg\min_{\Delta\mathbf{w}} \mathbb{E}[\mathcal{L}(\mathbf{x}, \mathbf{y}, \mathbf{w} + \Delta\mathbf{w}) - \mathcal{L}(\mathbf{x}, \mathbf{y}, \mathbf{w})]$. Then, the objective can be approximated by the Taylor series:

$$\mathbb{E}[\mathcal{L}(\mathbf{x}, \mathbf{y}, \mathbf{w}^q) - \mathcal{L}(\mathbf{x}, \mathbf{y}, \mathbf{w})] \approx \mathbb{E}[\Delta\mathbf{w}^T \nabla_w \mathcal{L}(\mathbf{x}, \mathbf{y}, \mathbf{w}) + \frac{1}{2} \Delta\mathbf{w}^T \cdot \nabla^2_{\mathbf{w}} \mathcal{L}(\mathbf{x}, \mathbf{y}, \mathbf{w}) \cdot \Delta\mathbf{w}]. \quad (10)$$

Providing that a pre-trained model is successfully converged, the first term can be neglected [20]. Then, Eq. (10) is simplified as

$$\Delta\mathcal{L} \approx \frac{1}{2} \Delta\mathbf{w}^T \cdot \mathbf{H}^{(w)} \cdot \Delta\mathbf{w}, \quad (11)$$

where $\mathbf{H}^{(w)}$ is the Hessian matrix with respect to $\mathbf{w}$ (i.e., $\mathbb{E}[\nabla^2_{\mathbf{w}} \mathcal{L}(\mathbf{x}, \mathbf{y}, \mathbf{w})]$). Assuming that there is no inter-layer

Table 6. Evaluation on Natural Language Processing Tasks

Model		BERT			DistilBERT		
Task (Dataset)		SQuAD v1.1	MRPC	MNLI	SQuAD v1.1	MRPC	MNLI
Metric		F1	F1	Accuracy	F1	F1	Accuracy
Full Prec. (Baseline)		88.22	88.93	84.01	85.64	85.91	81.03
Data-free	2/32	22.22	0.04	60.67	16.14	81.22	44.51
Mr.BiQ	2/32	86.87$\pm$0.12	89.21$\pm$0.28	83.29$\pm$0.11	83.86$\pm$0.14	86.62$\pm$0.36	79.97$\pm$0.14
BRECQ† [22]	2/32	84.24$\pm$0.14	88.85$\pm$0.36	82.88$\pm$0.18	80.49$\pm$0.17	86.32$\pm$0.39	79.96$\pm$0.16
Data-free	3/32	79.96	87.22	80.30	75.80	81.22	72.49
Mr.BiQ	3/32	87.80$\pm$0.08	89.63$\pm$0.42	83.88$\pm$0.12	85.30$\pm$0.09	86.71$\pm$0.35	80.87$\pm$0.08
BRECQ† [22]	3/32	87.65$\pm$0.07	89.24$\pm$0.34	83.79$\pm$0.11	85.22$\pm$0.07	86.45$\pm$0.52	80.76$\pm$0.10
Data-free	2/8	22.14	0.01	59.70	17.04	81.22	41.83
Mr.BiQ	2/8	**86.69**$\pm$0.17	**89.13**$\pm$0.36	**83.15**$\pm$0.13	**83.78**$\pm$0.16	**86.17**$\pm$0.63	**79.88**$\pm$0.15
BRECQ† [22]	2/8	84.23$\pm$0.10	88.79$\pm$0.54	82.81$\pm$0.18	80.39$\pm$0.12	86.02$\pm$0.60	79.81$\pm$0.16
Q8BERT‡ [48]	2/8	3.411	81.224	38.27	-	-	-
TernaryBERT‡ [50]	2/8	87.67	90.44	83.78	-	-	-
KDLSQ-BERT‡ [17]	2/8	88.45	90.29	83.51	-	-	-
Data-free	3/8	78.98	86.90	79.50	75.19	81.22	69.45
Mr.BiQ	3/8	87.69$\pm$0.09	89.43$\pm$0.31	83.80$\pm$0.10	85.15$\pm$0.12	86.42$\pm$0.38	80.76$\pm$0.08
BRECQ† [22]	3/8	87.41$\pm$0.10	89.05$\pm$0.34	83.75$\pm$0.10	85.00$\pm$0.09	86.40$\pm$0.36	80.68$\pm$0.12

† We apply BRECQ to Transformer models for NLP based on the open-source code.
‡ They are quantization-aware training (QAT) methods, whose figures are taken from [17]. The baselines of QAT methods are 88.696 (F1), 90.625 (F1), and 84.463 (Acc.) for SQuAD v1.1, MRPC, and MNLI, respectively.

dependency, we can relax Eq. (11) to per-layer optimization problem, namely,

$$\arg\min_{\Delta \mathbf{w}^{(l)}} \Delta \mathbf{w}^{(l)T} \cdot \mathbf{H}^{(w^{(l)})} \cdot \Delta \mathbf{w}^{(l)}, \tag{12}$$

where $\mathbf{w}^{(l)}$ and $\mathbf{H}^{(w^{(l)})}$ are weights and the Hessian matrix of layer l, respectively. Then, when $\mathbf{z}^{(l)} = \mathbf{W}^{(l)} \cdot \mathbf{x}^{(l-1)}$, the Hessian matrix of layer l can be expressed as follows:

$$\mathbf{H}^{(w^{(l)})} = \mathbb{E}[\mathbf{x}^{(l-1)}\mathbf{x}^{(l-1)T} \otimes \nabla^2_{z^l}\mathcal{L}] \tag{13}$$

$$\overset{(a_1)}{\approx} \mathbb{E}[\mathbf{x}^{(l-1)}\mathbf{x}^{(l-1)T} \otimes diag(\nabla^2_{z^l}\mathcal{L}_{i,i})] \tag{14}$$

$$\overset{(a_2)}{\approx} \mathbb{E}[\mathbf{x}^{(l-1)}\mathbf{x}^{(l-1)T}], \tag{15}$$

where $\otimes$ denotes the Kronecker product. By further assuming that (for (a_1) in Eq. (14)) every off-diagonal element of the Hessian is 0 and (for (a_2) in Eq. (15)) $\nabla^2_{z^l}$ is a constant independent of the input data samples (i.e., $\nabla^2_{z^l}\mathcal{L}_{i,i} = c$), the objective function can be approximated as follows:

$$\arg\min_{\Delta \mathbf{w}^{(l)}} \mathbb{E}[(\Delta \mathbf{W}^{(l)}\mathbf{x}^{(l-1)})^2]. \tag{16}$$

In other words, through a few approximation steps, minimizing reconstruction error becomes equivalent to minimizing $\Delta\mathcal{L}$ induced by quantization. As a key operation to perform Eq. (16) in a layer, AdaRound decides either rounding-up or -down each parameter by gradient-based optimization while maintaining the initial step size [29]. Meanwhile, BRECQ [22] includes additional considerations as an extensive study of AdaRound. Specifically, given that $\mathbf{z}$ is the pre-activations of the last layer, BRECQ describes the following approximation step:

$$\arg\min_{\Delta \mathbf{w}} \Delta \mathbf{w}^T \cdot \mathbf{H}^{(w)} \cdot \Delta \mathbf{w} \approx \arg\min_{\Delta \mathbf{w}} \Delta \mathbf{z}^T \cdot \mathbf{H}^{(z)} \cdot \Delta \mathbf{z}. \tag{17}$$

Then, BRECQ replaces the Hessian matrix with the diagonal empirical Fisher [27] ($diag((\frac{\partial L}{\partial \mathbf{z}_1})^2, ..., (\frac{\partial L}{\partial \mathbf{z}_n})^2)$). BRECQ also defines a set of reconstruction units (e.g., layer-, block-, and network-wise), among which block-wise optimization empirically shows the best performance. Thus, we adopt a block-wise reconstruction scheme to quantize models in a non-uniform manner.

6. Conclusion

We have proposed a post-training non-uniform quantization method, aiming at minimizing reconstruction error. The method exploits multi-level binary as the quantization data format, which has the benefits of a high compression ratio and multiple choices for activation quantization. Unlike prior works, this work has proposed a novel bottom-up approach for optimizing the multi-level step size and bit-code jointly. All in all, we have achieved a new state-of-the-art accuracy of post-training quantization on various models including CNNs and Transformer models.

Acknowledgements

The authors would like to thank Dr. Seijoon Kim for proofreading the manuscript and Dr. Daehyun Kim for helpful discussion.

References

[1] Renzo Andri, Lukas Cavigelli, Davide Rossi, and Luca Benini. YodaNN: An architecture for ultra-low power binary-weight CNN acceleration. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 37(1):48–60, 2017. 2

[2] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. 2, 7

[3] Yaohui Cai, Zhewei Yao, Zhen Dong, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. ZeroQ: A novel zero shot quantization framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13169–13178, 2020. 6

[4] 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. 1, 2

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, volume 1, pages 4171–4186, 2019. 4, 5

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2020. 4, 5

[7] Steven K Esser, Jeffrey L McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S Modha. Learned step size quantization. In *International Conference on Learning Representations (ICLR)*, 2019. 1, 4, 5, 7

[8] Mario Fischer and Juergen Wassner. BinArray: A scalable hardware accelerator for binary approximated CNNs. In *2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0197–0205, 2021. 2

[9] Jianping Gou, Baosheng Yu, Stephen John Maybank, and Dacheng Tao. Knowledge distillation: A survey. *arXiv preprint arXiv:2006.05525*, 2020. 1, 3

[10] Yiwen Guo, Anbang Yao, Hao Zhao, and Yurong Chen. Network sketching: Exploiting binary structure in deep CNNs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5955–5963, 2017. 2

[11] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *International Conference on Learning Representations (ICLR)*, 2016. 1

[12] Song Han, Jeff Pool, John Tran, and William J Dally. Learning both weights and connections for efficient neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1135–1143, 2015. 1

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 4

[14] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. 1, 3

[15] Itay Hubara, Yury Nahshan, Yair Hanani, Ron Banner, and Daniel Soudry. Improving post training neural quantization: Layer-wise calibration and integer programming. *arXiv preprint arXiv:2006.10518*, 2020. 6

[16] Yongkweon Jeon, Baeseong Park, Se Kwon, Byeongwook Kim, Jeongin Yun, and Dongsoo Lee. BiQGEMM: Matrix multiplication with lookup table for binary-coding-based quantized DNNs. In *The International Conference for High Performance Computing, Networking, Storage, and Analysis (SC)*, pages 1343–1356. IEEE Computer Society, 2020. 2

[17] Jing Jin, Cai Liang, Tiancheng Wu, Liqin Zou, and Zhiliang Gan. KDLSQ-BERT: A quantized BERT combining knowledge distillation with learned step size quantization. *arXiv preprint arXiv:2101.05938*, 2021. 7, 8

[18] Sehoon Kim, Amir Gholami, Zhewei Yao, Michael W Mahoney, and Kurt Keutzer. I-BERT: Integer-only BERT quantization. *International Conference on Machine Learning (ICML)*, 2021. 7

[19] Frederik Kunstner, Lukas Balles, and Philipp Hennig. Limitations of the empirical fisher approximation for natural gradient descent. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4156–4167, 2019. 3

[20] Yann LeCun, John S Denker, Sara A Solla, Richard E Howard, and Lawrence D Jackel. Optimal brain damage. In *Advances in Neural Information Processing Systems (NIPS)*, volume 2, pages 598–605, 1989. 7

[21] Rundong Li, Yan Wang, Feng Liang, Hongwei Qin, Junjie Yan, and Rui Fan. Fully quantized network for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2810–2819, 2019. 7

[22] Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. BRECQ: Pushing the limit of post-training quantization by block reconstruction. In *International Conference on Learning Representations (ICLR)*, 2021. 1, 3, 5, 6, 7, 8

[23] Yuanzhi Li, Yingyu Liang, and Andrej Risteski. Recovery guarantee of weighted low-rank approximation via alternating minimization. In *International Conference on Machine Learning (ICML)*, pages 2358–2367, 2016. 1

[24] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. In *International Conference on Learning Representations (ICLR)*, 2018. 1

[25] Zhenhua Liu, Yunhe Wang, Kai Han, Siwei Ma, and Wen Gao. Post-training quantization for vision transformer. *arXiv preprint arXiv:2106.14156*, 2021. 5, 7

[26] Christos Louizos, Max Welling, and Diederik P Kingma. Learning sparse neural networks through l_0 regularization.

In *International Conference on Learning Representations (ICLR)*, 2018. 3

[27] Alexander Ly, Maarten Marsman, Josine Verhagen, Raoul PPP Grasman, and Eric-Jan Wagenmakers. A tutorial on fisher information. *Journal of Mathematical Psychology*, 80:40–55, 2017. 8

[28] Huizi Mao, Song Han, Jeff Pool, Wenshuo Li, Xingyu Liu, Yu Wang, and William J Dally. Exploring the regularity of sparse structure in convolutional neural networks. *arXiv preprint arXiv:1705.08922*, 2017. 1

[29] Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. In *International Conference on Machine Learning (ICML)*, pages 7197–7206, 2020. 1, 3, 4, 5, 7, 8

[30] Markus Nagel, Mart van Baalen, Tijmen Blankevoort, and Max Welling. Data-free quantization through weight equalization and bias correction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1325–1334, 2019. 7

[31] Yury Nahshan, Brian Chmiel, Chaim Baskin, Evgenii Zheltonozhskii, Ron Banner, Alex M Bronstein, and Avi Mendelson. Loss aware post-training quantization. *Machine Learning*, pages 1–18, 2021. 6

[32] Gabriele Prato, Ella Charlaix, and Mehdi Rezagholizadeh. Fully quantized transformer for machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1–14, 2020. 5

[33] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10428–10436, 2020. 2, 4

[34] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2383–2392, 2016. 4

[35] 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 (ECCV)*, pages 525–542, 2016. 1

[36] 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. 4

[37] Tara N Sainath, Brian Kingsbury, Vikas Sindhwani, Ebru Arisoy, and Bhuvana Ramabhadran. Low-rank matrix factorization for deep neural network training with high-dimensional output targets. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6655–6659, 2013. 1

[38] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobileNetV2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4510–4520, 2018. 4

[39] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019. 2, 5

[40] Pierre Stock, Armand Joulin, Rémi Gribonval, Benjamin Graham, and Hervé Jégou. And the bit goes down: Revisiting the quantization of neural networks. In *International Conference on Learning Representations (ICLR)*, 2019. 1, 7

[41] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnas-Net: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2820–2828, 2019. 4

[42] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning (ICML)*, pages 10347–10357, 2021. 2, 4, 5

[43] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations (ICLR)*, 2018. 4

[44] Peisong Wang, Qiang Chen, Xiangyu He, and Jian Cheng. Towards accurate post-training network quantization via bit-split and stitching. In *International Conference on Machine Learning (ICML)*, pages 9847–9856, 2020. 1, 6, 7

[45] Bernard L Welch. The generalization of 'student's'problem when several different population varlances are involved. *Biometrika*, 34(1-2):28–35, 1947. 6

[46] Di Wu, Qi Tang, Yongle Zhao, Ming Zhang, Ying Fu, and Debing Zhang. EasyQuant: Post-training quantization via scale optimization. *arXiv preprint arXiv:2006.16669*, 2020. 7

[47] Chen Xu, Jianqiang Yao, Zhouchen Lin, Wenwu Ou, Yuanbin Cao, Zhirong Wang, and Hongbin Zha. Alternating multi-bit quantization for recurrent neural networks. In *International Conference on Learning Representations (ICLR)*, 2018. 1, 2, 3, 5, 7

[48] Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. Q8BERT: Quantized 8bit BERT. *arXiv preprint arXiv:1910.06188*, 2019. 7, 8

[49] 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 (ECCV)*, pages 365–382, 2018. 1, 2, 3, 5, 7

[50] Wei Zhang, Lu Hou, Yichun Yin, Lifeng Shang, Xiao Chen, Xin Jiang, and Qun Liu. TernaryBERT: Distillation-aware ultra-low bit BERT. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 509–521, 2020. 5, 7, 8