Compact Powers-of-Two: An Efficient Non-Uniform Quantization for Deep Neural Networks

Xinkuang Geng*, Siting Liu†, Jianfei Jiang*, Kai Jiang‡, Honglan Jiang*
* Department of Micro-Nano Electronics, Shanghai Jiao Tong University, Shanghai, China
† School of Information Science and Technology, ShanghaiTech University, Shanghai, China
‡ Inspur Academy of Science and Technology, Jinan, Shandong, China
xinkuang@sjtu.edu.cn, liust@shanghaitech.edu.cn, jiangjianfei@sjtu.edu.cn, jiangkai@inspur.com, honglan@sjtu.edu.cn

***Abstract*—To reduce the demands for computation and memory of deep neural networks (DNNs), various quantization techniques have been extensively investigated. However, conventional methods cannot effectively capture the intrinsic data characteristics in DNNs, leading to a high accuracy degradation when employing low-bit-width quantization. In order to better align with the bell-shaped distribution, we propose an efficient non-uniform quantization scheme, denoted as compact powers-of-two (CPoT). Aiming to avoid the rigid resolution inherent in powers-of-two (PoT) without introducing new issues, we add a fractional part to its encoding, followed by a biasing operation to eliminate the unrepresentable region around 0. This approach effectively balances the grid resolution in both the vicinity of 0 and the edge region. To facilitate the hardware implementation, we optimize the dot product for CPoT based on the computational characteristics of the quantized DNNs, where the precomputable terms are extracted and incorporated into bias. Consequently, a multiply-accumulate (MAC) unit is designed for CPoT using shifters and look-up tables (LUTs). The experimental results show that, even with a certain level of approximation, our proposed CPoT outperforms state-of-the-art methods in data-free quantization (DFQ), a post-training quantization (PTQ) technique focusing on data privacy and computational efficiency. Furthermore, CPoT demonstrates superior efficiency in area and power compared to other methods in hardware implementation.**

I. Introduction

With the dramatic development of machine learning, deep neural networks (DNNs) have achieved unprecedented success in various intelligent applications [1]–[3]. However, the high computational complexity and memory access requirements of DNNs have long been limiting factors for their application in embedded devices and edge computing platforms. To address these issues, researchers have extensively explored quantization techniques [3] to compress neural network models and accelerate computation, including post-training quantization (PTQ) and quantization-aware training (QAT) [3].

Data-free quantization (DFQ) [4]–[8] is receiving increasing attention, due to concerns about data privacy and training costs. While maintaining an acceptable model accuracy under a low bit-width, DFQ enables PTQ without accessing any dataset. Among state-of-the-art DFQ methods, the Hessian-based weight-flipping algorithm has demonstrated superior performance [8].

Generally, data in DNNs follow a bell-shaped distribution [9]–[11]; thus, assigning more quantization levels to the region with higher data density can naturally reduce the quantization error. While powers-of-two (PoT) quantization satisfies this feature, it does not benefit from a higher bit-width due to the rigid resolution [11]. Hence, [11], [12] propose to sum the PoT terms from multiple groups as quantization levels. However, such a combination cannot guarantee a complete alignment with the bell-shaped distribution and introduces other problems in data representation.

Although existing non-uniform quantization methods have shown some superiorities to conventional uniform quantization, they exhibit different limitations. To address these issues, we propose compact powers-of-two (CPoT), resulting in a suitable representation that produces low quantization error for data in DNNs. Furthermore, as per the computational characteristics of the quantized DNNs, an efficient multiply-accumulate (MAC) unit for CPoT is devised based on shifters and look-up tables (LUTs). In this design, different approximation levels can be chosen to accommodate different resource budgets.

Moreover, we propose a mapping technique to integrate CPoT with the weight-flipping algorithm originally designed for uniform quantization, achieving state-of-the-art results. In contrast, prior works [11]–[14] on non-uniform quantization focus on QAT with the need for resource-intensive retraining. To assess the efficiency of the proposed CPoT and MAC unit, we develop a precise DNN simulation platform for the approximate multiplier to guarantee consistency between the simulation results and real hardware deployments. Compared with other quantization methods with comparable accuracy, our design demonstrates lower area and power.

The rest of the paper is organized as follows. In Section II, DNNs and quantization techniques are introduced. In Section III, we propose the CPoT quantization scheme, optimize the computational flow, and design the MAC unit. Section IV evaluates the accuracy and hardware overhead of the proposed CPoT, and compares it with state-of-the-art quantization methods. Finally, in section V, we conclude this paper.

II. Preliminaries

In general, the major computations in DNNs consist of convolution and matrix multiplication. Without considering the data layout, these operations can all be interpreted as a group of vector dot products with bias as

$$y = \sum_i w^{\langle i \rangle} x^{\langle i \rangle} + b, \tag{1}$$

where w and b are the weights and bias, respectively. x and y represent the input activations and dot product results. Additionally, an activation function is applied to the output, providing nonlinearity and generating the input for the next

This work was supported in part by the National Key Research and Development Science and Technology under grant 2022YFB4500200; and in part by the National Natural Science Foundation of China under grant numbers 62374108 and 62104127.

979-8-3503-4859-0/DATE24/© 2024 EDAA

layer. The most commonly used activation function is the rectified linear unit (ReLU), which forces negative values to 0. In this case, the input of a neuron can be considered non-negative, saving a sign bit in practical encoding.

The quantization of weights or activations in DNNs can be regarded as a process of projecting their floating-point values onto some discretized grids as

$$z_g = \Pi_G \operatorname{clip}(z, -\lambda, \lambda), \tag{2}$$

where z is the data to be quantized, λ represents the statistical range of z, $\operatorname{clip}(\cdot)$ is used to limit z within $[-\lambda, \lambda]$, and G denotes the set of quantization grids that is the target for the projection. Without loss of generality, we assume that the clipping range is symmetric, otherwise, an additional zero point [3] is necessary to correct the data. G can be derived through a normalization step from the set of quantization points P as

$$G = \{ \frac{\lambda}{\kappa} z_p \mid z_p \in P, \kappa = \max(P) \}. \tag{3}$$

For uniform quantization, P_u is defined as

$$P_u = \{ 0, 1, 2, \cdots, 2^B - 1 \}, \tag{4}$$

where B represents the quantization bit-width. For simplicity, we assume that the data are unsigned when discussing different sets of quantization points. For signed data, the available bit-width will be reduced by 1, because an extra bit is required to identify the sign.

Since λ and κ are the same for all input data, floating-point operations can be avoided via quantization. Convolution and matrix multiplication are performed exclusively within the set of quantization points P as

$$y \approx \tilde{y} = \alpha \sum_i w_p^{\langle i \rangle} x_p^{\langle i \rangle} + b, \text{ where } \alpha = \frac{\lambda_w \lambda_x}{\kappa_w \kappa_x}. \tag{5}$$

It is noteworthy that the weight quantization is usually done offline, whereas the quantized activations are generated in real-time, making the re-quantization step necessary in DNN accelerators. However, compared with the numerous MAC operations, the re-quantize operation only needs to be done per dot product. Thus, the re-quantization accounts for a small portion of all computations in the quantized DNN inference, and can be efficiently implemented using the technique based on LUTs [12], especially for low-bit-width quantization. In other words, re-quantization incurs a minor hardware overhead. Therefore, in this paper, we focus on optimizing the dot product for quantization points.

III. Compact Powers-of-Two Quantization

A. Representation Method

As data in DNNs follow a bell-shaped distribution [9]–[11], to enhance the quantization accuracy, it is natural to assign more quantization levels around 0. Logarithmic quantization [9], also known as powers-of-two (PoT), is a common choice. Also, PoT is hardware-friendly since the multiplication can be replaced with addition and shift operations. The set of quantization points for PoT is defined as

$$P_p = \{ 0, 2^1, 2^2, \cdots, 2^{2^B-2}, 2^{2^B-1} \}. \tag{6}$$

Each element in P_p except for 0 can be expressed as 2^i, where $i \in \mathbb{N}^+$. Different from uniform quantization, PoT does not always benefit from increasing the quantization bit-width. This phenomenon is referred to as the rigid resolution [11]. According to (3), the sets of quantization grids for PoT with different bit-widths ($B_2 > B_1$) are

$$\begin{aligned} G_p^{B_1} &= \lambda \{ 0, \quad 2^{2-2^{B_1}}, 2^{3-2^{B_1}}, \cdots, 2^{-1}, 1 \} \text{ and} \\ G_p^{B_2} &= \lambda \{ 0, 2^{2-2^{B_2}}, \cdots, 2^{2-2^{B_1}}, 2^{3-2^{B_1}}, \cdots, 2^{-1}, 1 \}. \end{aligned} \tag{7}$$

It can be seen that, increasing B from B_1 to B_2 only results in the additional grids smaller than $\lambda 2^{2-2^{B_1}}$. As depicted in Fig. 1(a), when increasing B from 3 to 4, only the grids in the region below 2^{-6} become more compact, while the resolution of others remains constant. Thus, only a small portion of data very close to 0 can benefit from higher quantization bit-width.

To tackle this issue, [11] proposes additive powers-of-two (APoT). This approach involves grouping alternating PoT terms and obtaining quantization points by summing the terms from different groups. For instance, the sets of quantization points for APoT with B of 3 and 4 are defined as

$$\begin{aligned} P_a^3 &= \{ i + j \mid i \in \{ 0, 2^0, 2^2, 2^3 \}, j \in \{ 0, 2^1 \} \} \text{ and} \\ P_a^4 &= \{ i + j \mid i \in \{ 0, 2^0, 2^2, 2^4 \}, j \in \{ 0, 2^1, 2^3, 2^5 \} \}, \end{aligned} \tag{8}$$

where the PoT terms are divided into two groups, each occupying a part of the encoding space. For 5-bit and 6-bit quantization, the terms are divided into three groups. It is noteworthy that there is no clear inclusion relationship between PoT groups for different bit-widths. This means that the multiplier designed for a specific bit-width is not compatible with lower ones, unlike the integer multiplier.

As shown in Fig. 1(b), APoT exhibits another issue. Since the quantization points are generated by combination, the grid resolution does not align well with the bell-shaped distribution. APoT allocates additional closely spaced grids in the edge region with little benefit in reducing the relative quantization error. We refer to these grids as inefficient grids, which are formed by summing terms from different groups with significant disparities. Similar issues have also arisen in other work [12] that employs additions of grouped PoT terms to define quantization points.

The presence of the rigid resolution is attributed to the fact that PoT for different bit-widths maintains the same logarithm base two. Consequently, the numerical values between adjacent grids are doubled, resulting in a low resolution in the edge region. Some other works [9], [10], [15] address this issue by adjusting the logarithm base, i.e., adding a fractional part to the PoT encoding. This encoding scheme is referred to as fractional powers-of-two (FPoT). The set of quantization points for FPoT is defined as

$$P_f = \{ 0, 2^{\frac{1}{\gamma}}, 2^{\frac{2}{\gamma}}, \cdots, 2^{\frac{2^B-2}{\gamma}}, 2^{\frac{2^B-1}{\gamma}} \}, \tag{9}$$

where γ represents the base factor and $\log_2 \gamma$ determines the fractional bit-width of the exponent. Likewise, each element in P_f except for 0 can be expressed as $2^{\frac{i}{\gamma}}$, where $i \in \mathbb{N}^+$. According to (3), the set of quantization grids for FPoT is

$$G_f = \lambda \{ 0, 2^{\frac{2-2^B}{\gamma}}, 2^{\frac{3-2^B}{\gamma}}, \cdots, 2^{-\frac{1}{\gamma}}, 1 \}. \tag{10}$$

When simultaneously increasing the quantization bit-width and γ, the newly generated grids will not be limited to around 0. Instead, new grids will be inserted between the original

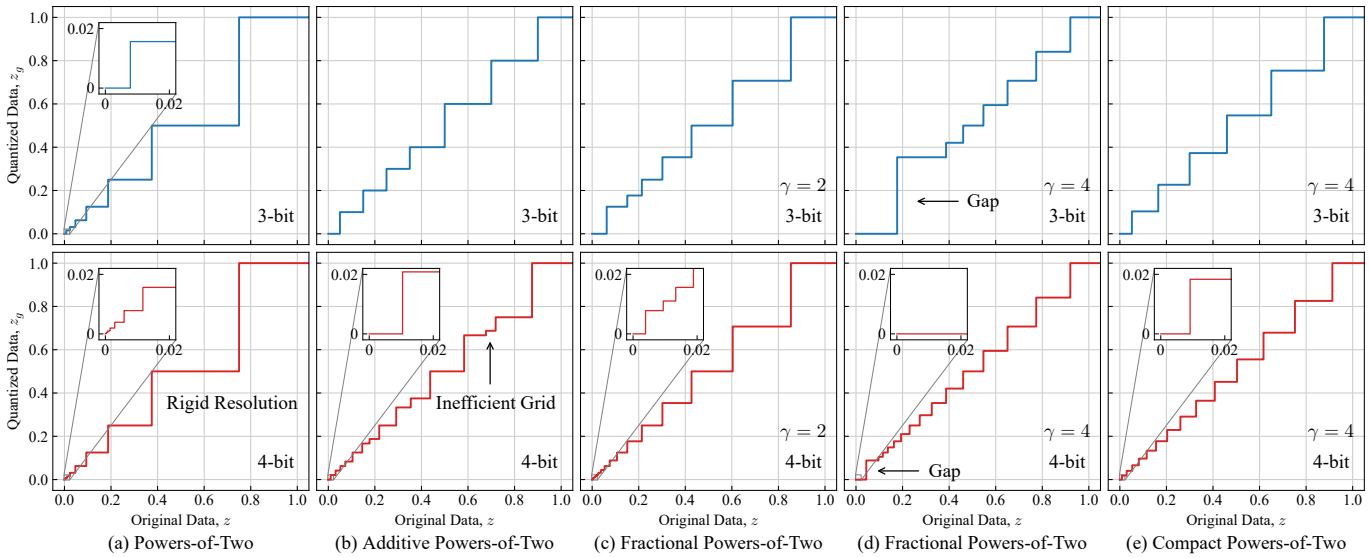


Fig. 1: Quantization grids with B of 3 and 4 for unsigned data ($\lambda = 1$) using different methods. CPoT quantization grids do not exhibit inefficient grids or a gap around 0 while eliminating the rigid resolution.

adjacent grids. For instance, the corresponding parts of the sets of quantization grids for PoT and FPoT are

$$\begin{aligned} G_p &= \lambda\{\cdots, 2^{-i-1}, \qquad\qquad 2^{-i}, \cdots\} \text{ and} \\ G_f &= \lambda\{\cdots, 2^{-i-1}, 2^{-i-\frac{\gamma-1}{\gamma}}, \cdots, 2^{-i-\frac{1}{\gamma}}, 2^{-i}, \cdots\}. \end{aligned} \tag{11}$$

It can be seen that with the same bit-width of the integer part encoding, FPoT inserts $\gamma - 1$ new grids between most adjacent grids of PoT, thus avoiding the rigid resolution in the edge region. Prior works [9], [10], [15] have demonstrated that FPoT with $\gamma = 2$ outperforms PoT in DNN quantization. However, further increasing γ tends to decrease the accuracy.

According to (10), as γ increases, the grid $g_1 = 2^{\frac{2-2^B}{\gamma}}$, which is the closest to 0, moves further away from 0. Consequently, the region between 0 and g_1 occupies a larger proportion, where the data cannot be efficiently represented, as shown in Fig. 1(c) and 1(d). It is noteworthy that, with the same quantization bit-width, as γ increases, the impact of the gap between 0 and g_1 becomes more significant, particularly for the data of bell-shaped distribution. This phenomenon is more pronounced in low-bit-width quantization, especially when more bits are assigned to the fractional part encoding to enhance the resolution in the edge region.

From another perspective, this gap around 0 of FPoT arises because a larger γ results in a much smaller range of P_f, while the quantization point closest to 0, $p_1 = 2^{\frac{1}{\gamma}}$, keeps greater than 1. Notably, as γ increases, the gap between p_1 and 1 does decrease. This is consistent with our objective of optimizing the grid resolution by increasing γ. Therefore, it is natural to replace 0 with 1 as the starting point for quantization. Based on this idea, we propose compact powers-of-two (CPoT). The set of quantization points for CPoT is defined as

$$P_c = \{2^{\frac{0}{\gamma}} - 1, 2^{\frac{1}{\gamma}} - 1, 2^{\frac{2}{\gamma}} - 1, \cdots, 2^{\frac{2^B-1}{\gamma}} - 1\}. \tag{12}$$

Each element in P_c can be expressed as $2^{\frac{i}{\gamma}} - 1$, where $i \in \mathbb{N}$. CPoT addresses the issue that FPoT grids cannot efficiently represent the data around 0 for a large γ, by biasing all quantization points in FPoT towards 0 by 1. As depicted in Fig. 1(e), the CPoT grids are generated by scaling the cropped FPoT grids back to the entire range. In other words, CPoT redistributes the gap between 0 and g_1 of FPoT across all grids, and the major reduction in grid resolution is absorbed by the sparse grids in the edge region, which minimizes the impact on the relative quantization error.

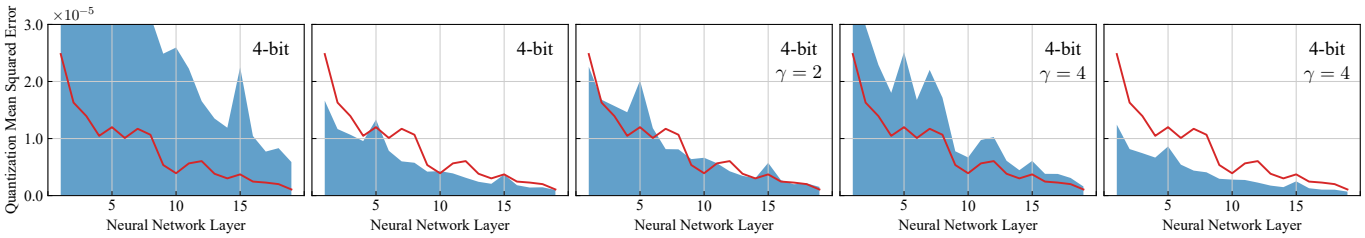
Such non-uniform grids for CPoT offer two additional advantages. Firstly, the rigid resolution and inefficient grids are eliminated. Secondly, within the set of quantization points for CPoT, 0 does not need to be specially encoded and processed; instead, it shares the same expression with other points as shown in (12), becoming a regular element in the number system. This feature enhances the consistency and simplifies the design of arithmetic units.

To compare the performance of CPoT with other quantization methods, we evaluate the quantization error of the weights in each layer of ResNet-18 [1], based on the mean squared error (MSE). The quantization bit-width is set to 5, corresponding to the 4-bit grids shown in Fig. 1, after excluding the sign bit. The evaluation results of different quantization methods are depicted in Fig. 2. Notably, PoT introduces a larger MSE than the uniform quantization. As γ increases, compared with PoT, the MSE of FPoT first decreases ($\gamma = 2$), and then increases ($\gamma = 4$). While APoT performs a lower MSE than uniform quantization in most layers, CPoT achieves the lowest quantization error.

B. Computation Scheme and Hardware Design

According to (5), in the DNN quantized by CPoT, the major computations during inference occur within the set of quantization points P_c. We encode the element $2^{\frac{i}{\gamma}} - 1$ as a fixed-point number $\frac{i}{\gamma}$. If the data are signed, an additional sign bit will occupy the most significant bit. Assuming that W and X represent the encodings of w_p and x_p with a $\log_2\gamma$-bit fractional part. Subsequently, the multiplication in (5) can be expressed as

$$\begin{aligned} w_p x_p &= \operatorname{sgn}(w_p)\left(2^W - 1\right)\left(2^X - 1\right) \\ &= \operatorname{sgn}(w_p)\left(2^{W+X} - 2^X\right) - w_p, \end{aligned} \tag{13}$$



(a) Powers-of-Two (b) Additive Powers-of-Two (c) Fractional Powers-of-Two (d) Fractional Powers-of-Two (e) Compact Powers-of-Two

Fig. 2: Quantization MSEs across layers in ResNet-18 for various methods. The blue area represents the error specific to each method, while the red line serves as a reference for uniform quantization.

where w_p is separated from the dot product and to be accumulated offline, reducing the complexity of the MAC unit. Furthermore, to minimize the additional logic for dealing with negative values during accumulation, we convert two's complement to one's complement by rewriting (13) as

$$w_p x_p = c\left(w_p, 2^{W+X} - 2^X\right) + s\left(w_p\right) - w_p, \tag{14}$$

where

$$c\left(u, v\right) = \begin{cases} v, & u \geq 0 \\ \backsim v, & \text{else} \end{cases} \quad \text{and} \quad s\left(u\right) = \begin{cases} 0, & u \geq 0 \\ '1, & \text{else} \end{cases}, \tag{15}$$

where $'1$ indicates 1 at the least significant bit of the addition.

After separating these precomputable terms from the core computation flow, we incorporate all these terms from the dot product into bias. Consequently, (5) can be rewritten as

$$\tilde{y} = \alpha \sum_i c\left(w_p^{\langle i \rangle}, 2^{W^{\langle i \rangle} + X^{\langle i \rangle}} - 2^{X^{\langle i \rangle}}\right) + \tilde{b}, \tag{16}$$

where

$$\tilde{b} = \alpha \sum_i s\left(w_p^{\langle i \rangle}\right) - \alpha \sum_i w_p^{\langle i \rangle} + b. \tag{17}$$

Finally, we approximate each FPoT term in (16) by using an LUT as

$$2^X = 2^{X_i + \frac{X_f}{\gamma}} = 2^{\frac{X_f}{\gamma}} \ll X_i \approx \mathrm{LUT}\left(X_f\right) \ll X_i, \tag{18}$$

where X_i and $\frac{X_f}{\gamma}$ represent the integer and fractional parts of the fixed-point number X, respectively. Each entry in the LUT stores the value of $2^{\frac{X_f}{\gamma}}$ with an implicit leading one and a B_{lut}-bit fractional part; it can be precomputed as

$$\mathrm{LUT}\left(i\right) = 2^{-B_{\mathrm{lut}}} \lfloor 2^{\frac{i}{\gamma}} 2^{B_{\mathrm{lut}}} \rceil, \ i = 0, 1, \cdots, \gamma - 1, \tag{19}$$

where $\lfloor \cdot \rceil$ denotes the round-to-nearest operation. The value of B_{lut} determines the approximation level. Notably, the degree of the approximation is relatively low. For instance, setting B_{lut} to 0 offers an approximation equivalent to discarding the fractional part from the encoding, and it still yields performance similar to PoT quantization with a lower bit-width.

Based on the optimized computation flow, we design a MAC unit for CPoT, as depicted in Fig. 3. The LUT is implemented by a combinational logic block with decoding functionality, which takes the index i as input and generates the corresponding value $\mathrm{LUT}\left(i\right)$ as output. Furthermore, our proposed MAC unit exhibits compatibility with operations employing lower bit-widths by padding zeros before and after the encoding, while maintaining a fixed position for the decimal point.

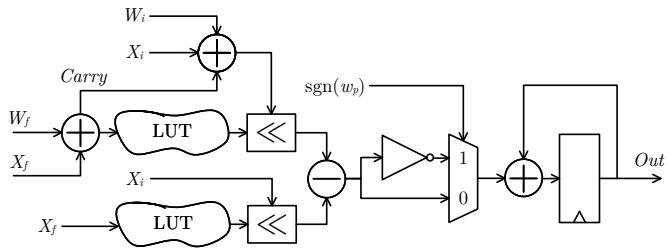


Fig. 3: Design of the MAC unit for CPoT.

IV. Experiments

A. Accuracy Evaluation

To assess the efficiency of the proposed CPoT, it is evaluated in the experiments of DFQ with various quantization bit-widths on the ImageNet dataset [16]. Three convolutional neural network (CNN) architectures, ResNet-18 [1], ResNet-50 [1], and Inception-V3 [2] are considered. Furthermore, four existing DFQ methods, ZeroQ [5], DSG [6], GDFQ [7], and SQuant [8], are compared.

In order to compare with the optimal DFQ methods, we integrate CPoT with the Hessian-based weight-flipping algorithm proposed in [8], which reconstructs the quantized weights as

$$\tilde{w}_g = \mathrm{flip}\left(\Gamma, w_g, w\right), \tag{20}$$

where $w_g, \tilde{w}_g \in G_u$, Γ represents the DNN architecture, and $\mathrm{flip}\left(\cdot\right)$ flips certain quantized weights w_g to the adjacent grid to minimize the absolute sum of error [8]. Therefore, the reconstructed weights no longer follow a simple round-to-nearest rule concerning the original weights. However, the weight-flipping algorithm is designed for uniform quantization, assuming equal spacing between adjacent grids. To adapt the algorithm for CPoT, we define a mapping as

$$\mu\left(z\right) = \mathrm{sgn}\left(z\right) \log_2\left(\frac{\kappa}{\lambda}\left|z\right| + 1\right), \tag{21}$$

to map w_g to a new set of discrete grids with equal intervals, while w is also mapped within the range of the set. It is noteworthy that the relative order of all data before and after mapping remains unchanged. This implies that the mapped data still preserves some of the properties required by the weight-flipping algorithm. As the mapping is invertible, the modified weight-flipping algorithm can be expressed as

$$\tilde{w}_g = \mu^{-1} \circ \mathrm{flip}\left(\Gamma, \mu\left(w_g\right), \mu\left(w\right)\right), \tag{22}$$

where $w_g, \tilde{w}_g \in G_c$. While it may not be the optimal weight reconstruction method for CPoT, it does produce satisfactory

results and offers a simple way to integrate CPoT with existing quantization techniques.

In the experiments, we test the cases involving 4-bit and 6-bit quantization. For CPoT encoding, the fractional bit-width is set to half of the quantization bit-width, i.e., $\log_2 \gamma = 2$ for 4-bit quantization and $\log_2 \gamma = 3$ for 6-bit quantization. This configuration is applied to both weights and activations. Since weights are not non-negative, we apply symmetric quantization to weights by simply designating the most significant bit of the encoding as the sign bit. The calibration data are generated randomly, and the entire quantization process is conducted independently of any data from the training or validation datasets. Following [8], we determine the quantization range λ in (2) based on the standard deviation of the data. Additionally, 8-bit uniform quantization is applied to the input of the last layer to ensure fairness when compared with other DFQ implementations.

TABLE I: TOP-1 ACCURACY OF THREE CNNS ON IMAGENET USING DIFFERENT DFQ METHODS

MODEL	METHOD	W	A	TOP-1	W	A	TOP-1
RESNET -18 PARAMS. 11.7M	FP	32	32	71.47			
	ZEROQ	4	4	19.09	6	6	69.84
	DSG	4	4	34.53	6	6	70.46
	GDFQ	4	4	60.60	6	6	70.13
	SQUANT	4	4	66.14	6	6	70.74
	CPOT	4	4	**68.22**	6	6	**71.35**
RESNET -50 PARAMS. 25.6M	FP	32	32	77.74			
	ZEROQ	4	4	7.75	6	6	72.93
	DSG	4	4	23.10	6	6	76.07
	GDFQ	4	4	55.65	6	6	76.59
	SQUANT	4	4	70.80	6	6	77.05
	CPOT	4	4	**71.69**	6	6	**77.37**
INCEPTION -V3 PARAMS. 23.8M	FP	32	32	78.81			
	ZEROQ	4	4	18.20	6	6	74.94
	DSG	4	4	34.89	6	6	76.52
	GDFQ	4	4	70.39	6	6	77.20
	SQUANT	4	4	73.26	6	6	78.30
	CPOT	4	4	**76.13**	6	6	**78.63**

Table I shows the accuracy results for 4-bit and 6-bit quantization based on different quantization methods. The experimental results demonstrate that CPoT achieves the highest accuracy, outperforming state-of-the-art DFQ methods. In the cases of 6-bit quantization, CPoT results in an accuracy loss of less than 0.2% in ResNet-18 and Inception-V3, and less than 0.4% in ResNet-50, compared with the 32-bit floating-point implementations. In the cases of 4-bit quantization, CPoT exhibits significant advantages over state-of-the-art methods, achieving accuracy improvements ranging from approximately 1% to 3% compared with the best results. Furthermore, the existing DFQ implementations employ asymmetric quantization for weights, which requires additional storage and computation for zero points. In contrast, our proposed CPoT quantization scheme eliminates the need for these demands.

B. Ablation Study

Our proposed DFQ scheme comprises two techniques, CPoT defines the quantization points, and the modified weight-flipping algorithm reconstructs the quantized weights using second-order information. The ablation experiments are conducted separately on these two techniques.

In the experiments, we compare methods involving different quantization points, uniform quantization [3], JLQ [14], PoT [9], APoT [11], FPoT [10], and CPoT quantization. These methods are evaluated both with and without the (modified) weight-flipping algorithm. Since JLQ is specifically designed for weights, we keep the activations as floating-point numbers. In the cases of FPoT quantization, the optimal fractional bit-width $\log_2 \gamma$ is determined through an accuracy-oriented exploration. As a result, we set $\log_2 \gamma = 1$ for 4-bit quantization and $\log_2 \gamma = 2$ for 6-bit quantization.

It is noteworthy that if it is not feasible to find an invertible mapping that can map the data within the range of the non-uniform grids to a uniform domain, the method cannot be integrated with the existing weight-flipping algorithm tailored for uniform quantization. Consequently, we restrict the application of the modified weight-flipping algorithm to PoT and FPoT, with a mapping similar to (21), while disregarding the discontinuity around 0.

TABLE II: TOP-1 ACCURACY OF RESNET-18 ON IMAGENET USING DIFFERENT METHODS

METHOD	W	A	FLIP†	TOP-1	FLIP†	TOP-1
UNIFORM	4	4	✗	44.13	✓	66.14
JLQ	4	32	✗	8.37	✓	-
POT	4	4	✗	22.09	✓	41.55
APOT	4	4	✗	42.07	✓	-
FPOT	4	4	✗	40.35	✓	21.29
CPOT	4	4	✗	**49.50**	✓	**68.22**
UNIFORM	6	6	✗	69.49	✓	70.74
JLQ	6	32	✗	12.81	✓	-
POT	6	6	✗	31.31	✓	41.47
APOT	6	6	✗	70.63	✓	-
FPOT	6	6	✗	70.51	✓	70.33
CPOT	6	6	✗	**71.04**	✓	**71.35**

† Whether to use the (modified) weight-flipping algorithm to refine the quantized weights.

The accuracy results of ResNet-18 using different techniques are presented in Table II. It can be seen that CPoT consistently exhibits superior performance across different scenarios. In contrast, JLQ, designed for QAT on small datasets with low quantization bit-widths (2-bit or 3-bit), demonstrates suboptimal performance in DFQ on the ImageNet dataset. We also observe that increasing the bit-width from 4 to 6 in PoT yields a limited improvement in accuracy, due to the rigid resolution. While APoT outperforms uniform quantization without using the weight-flipping algorithm for 6-bit quantization, it results in a decreased accuracy for the 4-bit case. This may be due to the impact of the inefficient grids that are more pronounced at lower quantization bit-widths. When integrated with the modified weight-flipping algorithm, an accuracy loss is introduced in FPoT. The phenomenon is likely attributed to the discontinuous mapping (the gap around 0), which might significantly affect the efficiency of the weight-flipping algorithm.

C. Approximation Evaluation

To evaluate the performance of the proposed approximate multiplier for CPoT, we implement it using CUDA C++ based on the optimized computation flow. Subsequently, we encapsulate this multiplication to construct operators within PyTorch, including convolution and matrix multiplication, and

then replace the original floating-point operators during the inference phase.

TABLE III: TOP-1 ACCURACY OF THREE CNNS ON IMAGENET USING APPROXIMATE MULTIPLIERS WITH DIFFERENT BIT-WIDTHS FOR THE LUT ENTRY

MODEL	W	A	FULL†	8-BIT	6-BIT	4-BIT	2-BIT
RESNET -18	4	4	68.22	68.04	68.00	**67.68**	57.87
	6	6	71.35	71.42	71.35	71.44	**71.01**
RESNET -50	4	4	71.69	71.86	71.95	**71.58**	56.58
	6	6	77.37	77.41	77.41	77.36	**76.81**
INCEPTION -V3	4	4	76.13	76.04	76.06	**75.72**	64.26
	6	6	78.63	78.64	78.63	78.68	**78.55**

† Floating-point operations are used to simulate the multiplication between quantization points without approximation.

Table III shows the accuracy of the three CNNs using multipliers with different values of B_{lut} that represent the approximation level. For 6-bit quantization, by setting B_{lut} to 2, the obtained accuracy is close to those of floating-point implementations, outperforming most of the quantization methods presented in Table I. When B_{lut} is 4, a slight improvement occurs in the accuracy. We speculate that at this approximation level, the approximate multiplier exhibits a regularization effect, which might help in mitigating overfitting. For 4-bit quantization, by setting B_{lut} to 4, the accuracy drop of all three DNNs can be kept within 0.6%, still outperforming all the quantization methods presented in Table I.

D. Hardware Evaluation

To evaluate the hardware efficiency of the proposed MAC unit for CPoT, it is synthesized by using Synopsys Design Compiler based on HLMC 28 nm technology. In addition, the MAC units designed for other methods, including uniform and APoT quantization, are considered for comparison. For the MAC unit designed for uniform quantization, we directly use the integer multiplier IP in DesignWare. For the MAC unit designed for APoT quantization, we also perform careful optimization, such as reducing the additional logic for dealing with the negative values before accumulation, as shown in (14). Consistent constraints are applied to all designs, and the clock frequency is set as 1 GHz. Table IV presents the area and power for the MAC units.

TABLE IV: AREA AND POWER OF VARIOUS MAC UNITS

METHOD	W	A	B†	C‡	A(μm^2)	P(μW)
UNIFORM	6	6	-	✓	306.7	172.3
APOT	6	6	-	✗	410.1 +33.7%	204.2 +18.5%
CPOT	6	6	4	✓	**290.3** −5.3%	**167.3** −2.9%
CPOT	6	6	2	✓	**259.0** −15.6%	**125.1** −27.4%

† Bit-width of the LUT entry.
‡ Compatibility with operations of any lower bit-width.

It shows that the proposed MAC unit demonstrates higher resource efficiency than others, for both 4-bit and 2-bit LUT entries. Compared with the MAC unit for APoT, our design reduces the area and power by up to 36.8% and 38.7%, respectively; it also offers compatibility with operations of any lower bit-width. Notably, while the design for APoT is multiplication-free, the combination of the PoT terms from different groups results in an addition of numerous partial sums with high bit-widths, thereby incurring an additional resource overhead.

V. CONCLUSION

In this paper, we propose compact powers-of-two (CPoT), an efficient non-uniform quantization technique for data in DNNs. By aligning the grid resolution to match the bell-shaped distribution, we address the prevalent issues in existing non-uniform quantization methods. After being integrated into the DFQ framework, CPoT achieves state-of-the-art accuracy results. Additionally, leveraging the optimized computation flow, a MAC unit has been designed for CPoT with adjustable approximation levels. With a higher accuracy, our MAC unit reduces the area and power by up to 15.5% and 27.4% compared with the integer counterpart.

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[2] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[3] M. Nagel, M. Fournarakis, R. A. Amjad, Y. Bondarenko, M. Van Baalen, and T. Blankevoort, "A white paper on neural network quantization," *arXiv preprint arXiv:2106.08295*, 2021.

[4] M. Nagel, M. v. Baalen, T. Blankevoort, and M. Welling, "Data-free quantization through weight equalization and bias correction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1325–1334.

[5] Y. Cai, Z. Yao, Z. Dong, A. Gholami, M. W. Mahoney, and K. Keutzer, "ZeroQ: A novel zero shot quantization framework," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 13 169–13 178.

[6] X. Zhang, H. Qin, Y. Ding, R. Gong, Q. Yan, R. Tao, Y. Li, F. Yu, and X. Liu, "Diversifying sample generation for accurate data-free quantization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 658–15 667.

[7] S. Xu, H. Li, B. Zhuang, J. Liu, J. Cao, C. Liang, and M. Tan, "Generative low-bitwidth data free quantization," in *European Conference on Computer Vision*, 2020, pp. 1–17.

[8] C. Guo, Y. Qiu, J. Leng, X. Gao, C. Zhang, Y. Liu, F. Yang, Y. Zhu, and M. Guo, "SQuant: On-the-fly data-free quantization via diagonal hessian approximation," in *International Conference on Learning Representations*, 2022, pp. 1–18.

[9] D. Miyashita, E. H. Lee, and B. Murmann, "Convolutional neural networks using logarithmic data representation," *arXiv preprint arXiv:1603.01025*, 2016.

[10] S. Vogel, M. Liang, A. Guntoro, W. Stechele, and G. Ascheid, "Efficient hardware acceleration of cnns using logarithmic data representation with arbitrary log-base," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2018, pp. 1–8.

[11] Y. Li, X. Dong, and W. Wang, "Additive powers-of-two quantization: An efficient non-uniform discretization for neural networks," in *International Conference on Learning Representations*, 2019, pp. 1–15.

[12] T. Xia, B. Zhao, J. Ma, G. Fu, W. Zhao, N. Zheng, and P. Ren, "An energy-and-area-efficient cnn accelerator for universal powers-of-two quantization," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 70, no. 3, pp. 1242–1255, 2022.

[13] M. Elhoushi, Z. Chen, F. Shafiq, Y. H. Tian, and J. Y. Li, "DeepShift: Towards multiplication-less neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 2359–2368.

[14] L. Jiang, D. Aledo, and R. van Leuken, "Jumping Shift: A logarithmic quantization method for low-power cnn acceleration," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2023, pp. 1–6.

[15] J. Zhao, S. Dai, R. Venkatesan, B. Zimmer, M. Ali, M.-Y. Liu, B. Khailany, W. J. Dally, and A. Anandkumar, "LNS-Madam: Low-precision training in logarithmic number system using multiplicative weight update," *IEEE Transactions on Computers*, vol. 71, no. 12, pp. 3179–3190, 2022.

[16] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, pp. 211–252, 2015.