OMPQ: Orthogonal Mixed Precision Quantization

Yuexiao Ma[1], Taisong Jin[2]*, Xiawu Zheng[3], Yan Wang[4], Huixia Li[2],
Yongjian Wu[5], Guannan Jiang[6], Wei Zhang[6], Rongrong Ji[1]

[1] MAC Lab, Department of Artificial Intelligence, School of Informatics, Xiamen University, China.
[2] MAC Lab, Department of Computer Science and Technology, School of Informatics, Xiamen University, China.
[3] Peng Cheng Laboratory, Shenzhen, China.
[4] Samsara, Seattle, WA, USA.
[5] Tencent Technology (Shanghai) Co., Ltd, China.
[6] CATL, China.
bobma@stu.xmu.edu.cn, jintaisong@xmu.edu.cn, zhengxw01@pcl.ac.cn, yan.wang@samsara.com,
hxlee@stu.xmu.edu.cn, littlekenwu@tencent.com, {jianggn, zhangwei}@catl.com, rrji@xmu.edu.cn

Abstract

To bridge the ever-increasing gap between deep neural networks' complexity and hardware capability, network quantization has attracted more and more research attention. The latest trend of mixed precision quantization takes advantage of hardware's multiple bit-width arithmetic operations to unleash the full potential of network quantization. However, existing approaches rely heavily on an extremely time-consuming search process and various relaxations when seeking the optimal bit configuration. To address this issue, we propose to optimize a proxy metric of network orthogonality that can be efficiently solved with linear programming, which proves to be highly correlated with quantized model accuracy and bit-width. Our approach significantly reduces the search time and the required data amount by orders of magnitude, but without a compromise on quantization accuracy. Specifically, we achieve 72.08% Top-1 accuracy on ResNet-18 with 6.7Mb parameters, which does not require any searching iterations. Given the high efficiency and low data dependency of our algorithm, we use it for the post-training quantization, which achieves 71.27% Top-1 accuracy on MobileNetV2 with only 1.5Mb parameters.

Introduction

Recently, we witness an obvious trend in deep learning that the models have rapidly increasing complexity (He et al. 2016; Simonyan and Zisserman 2014; Szegedy et al. 2015; Howard et al. 2017; Sandler et al. 2018; Zhang et al. 2018b). Due to practical limits such as latency, battery, and temperature, the host hardware where the models are deployed cannot keep up with this trend. It results in a large and ever-increasing gap between the computational demands and the resources. To address this issue, compression and acceleration methods such as neural architecture search (Zheng et al. 2019, 2020; Zhou et al. 2021; Zheng et al. 2022, 2021a,c; Zhang et al. 2021; Zheng et al. 2023; Zhang et al. 2023), quantization (Courbariaux et al. 2016; Rastegari et al. 2016; Kim et al. 2019; Banner, Nahshan, and Soudry 2019; Liu
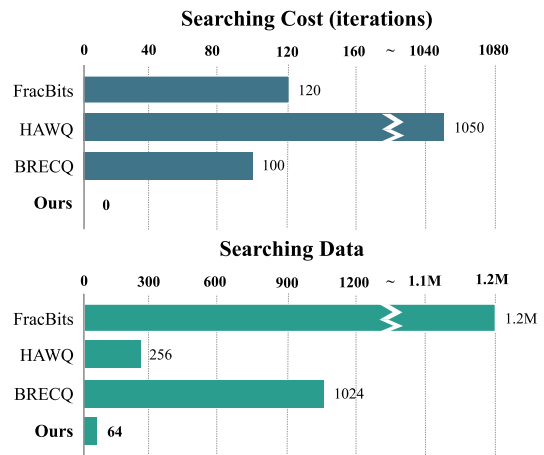

Figure 1: Comparison of the resources used to obtain the optimal bit configuration between our algorithm and other mixed precision algorithms (FracBits (Yang and Jin 2021), HAWQ (Dong et al. 2020), BRECQ (Li et al. 2021)) on ResNet-18. "Searching Data" is the number of input images.

et al. 2019; Li et al. 2020), and pruning (Zheng et al. 2021b; Han, Mao, and Dally 2015) have emerged. Among them, network quantization, which maps single-precision floating point weights or activations to lower bits integers for compression and acceleration, has attracted considerable research attention. Network quantization can be naturally formulated as an optimization problem and a straightforward approach is to relax the constraints to make it a tractable optimization problem, at a cost of an approximated solution, *e.g.* Straight Through Estimation (STE) (Bengio, Léonard, and Courville 2013).

With the recent development of inference hardware, arithmetic operations with variable bit-width become a possibility and bring further flexibility to the network quantization. To take full advantage of the hardware capability, mixed precision quantization (Dong et al. 2020; Wang et al. 2019; Li et al. 2021; Yang and Jin 2021) aims to quantize differ-

ent network layers to different bit-width, so as to achieve a better trade-off between compression ratio and accuracy. While benefiting from the extra flexibility, the mixed precision quantization also suffers from a more complicated and challenging optimization problem, with a non-differentiable and extremely non-convex objective function. Therefore, existing approaches (Dong et al. 2020; Yang and Jin 2021; Wang et al. 2019; Li et al. 2021) often require numerous data and computing resources to search for the optimal bit configuration. For instance, FracBits (Yang and Jin 2021) approximates the bit-width by performing a first-order Taylor expansion at the adjacent integer, making the bit variable differentiable. This allows it to integrate the search process into training to obtain the optimal bit configuration. However, to derive a decent solution, it still requires a large amount of computation resources in the searching and training process. To resolve the large demand on training data, Dong *et al.* (Dong et al. 2020) use the average eigenvalue of the hessian matrix of each layer as the metric for bit allocation. However, the matrix-free Hutchinson algorithm for implicitly calculating the average of the eigenvalues of the hessian matrix still needs 50 iterations for each network layer. Another direction is black box optimization. For instance, Wang *et al.* (Wang et al. 2019) use reinforcement learning for the bit allocation of each layer. Li *et al.* (Li et al. 2021) use evolutionary search algorithm (Guo et al. 2020) to derive the optimal bit configuration, together with a block reconstruction strategy to efficiently optimize the quantized model. But the population evolution process requires $1,024$ input data and 100 iterations, which is time-consuming.

Different from the existing approaches of black box optimization or constraint relaxation, we propose to construct a proxy metric, which could have a substantially different form, but be highly correlated with the objective function of original linear programming. In general, we propose to obtain the optimal bit configuration by using the orthogonality of neural network. Specifically, we deconstruct the neural network into a set of functions, and define the orthogonality of the model by extending its definition from a function $f : \mathbb{R} \to \mathbb{R}$ to the entire network $f : \mathbb{R}^m \to \mathbb{R}^n$. The measurement of the orthogonality could be efficiently performed with Monte Carlo sampling and Cauchy-Schwarz inequality, based on which we propose an efficient metric named **OR**thogonality **M**etric (ORM) as the proxy metric. As illustrated in Fig. 1, we only need a single-pass search process on a small amount of data with ORM. In addition, we derive an equivalent form of ORM to accelerate the computation.

On the other hand, model orthogonality and quantization accuracy are positively correlated on different networks. Therefore, maximizing model orthogonality is taken as our objective function. Meanwhile, our experiments show that layer orthogonality and bit-width are also positively correlated. We assign a larger bit-width to the layer with larger orthogonality while combining specific constraints to construct a linear programming problem. The optimal bit configuration can be gained simply by solving the linear programming problem.

In summary, our contributions are listed as follows:

- We introduce a novel metric of layer orthogonality.

We introduce function orthogonality into neural networks and propose the **OR**thogonality **M**etric (ORM). We leverage it as a proxy metric to efficiently solve the mixed precision quantization problem, which is the first attempt in the community and can easily be integrated into any quantization scheme.

- We observe a positive correlation between ORM and quantization accuracy on different models. Therefore, we optimize the model orthogonality through the linear programming problem, which can derive the optimal bit-width configuration in a few seconds without iterations.

- We also provide extensive experimental results on ImageNet, which elaborate that the proposed orthogonality-based approach could gain the state-of-the-art quantization performance with orders of magnitude's speed up.

Related Work

Quantized Neural Networks: Existing neural network quantization algorithms can be divided into two categories based on their training strategy: post-training quantization (PTQ) and quantization-aware training (QAT). PTQ (Li et al. 2021; Cai et al. 2020; Nagel et al. 2019) is an offline quantization method, which only needs a small amount of data to complete the quantization process. Therefore, PTQ could obtain an optimal quantized model efficiently, at a cost of accuracy drop from quantization. In contrast, QAT (Gong et al. 2019; Zhou et al. 2016; Dong et al. 2020; Zhou et al. 2017; Chen, Wang, and Pan 2019; Cai et al. 2017; Choi et al. 2018) adopts an online quantization strategy, which utilizes the whole training dataset during quantization process. As a result, it has superior accuracy but limited efficiency.

If viewed from a perspective of bit-width allocation strategy, neural network quantization can also be divided into unified quantization and mixed precision quantization. Choi *et al.* (Choi et al. 2018) aim to optimize the parameterized clip boundary of activation value of each layer during training process. Recently, some works (Yang and Jin 2021; Dong et al. 2020; Li et al. 2021) that explore assigning different bit-widths to different layers begin to emerge. Yang *et al.* (Yang and Jin 2021) approximate the derivative of bit-width by first-order Taylor expansion at adjacent integer points, thereby fusing the optimal bit-width selection with the training process.

Network Similarity: Previous works (Bach and Jordan 2002; Gretton, Herbrich, and Smola 2003; Leurgans, Moyeed, and Silverman 1993; Fukumizu, Bach, and Jordan 2004; Gretton et al. 2005; Kornblith et al. 2019) define covariance and cross-covariance operators in the Reproducing Kernel Hilbert Spaces (RKHSs), and derive mutual information criteria based on these operators. Gretton *et al.* (Gretton et al. 2005) propose the Hilbert-Schmidt Independence Criterion (HSIC), and give a finite-dimensional approximation of it. Furthermore, Kornblith *et al.* (Kornblith et al. 2019) give the similarity criterion CKA based on HSIC, and study its relationship with the other similarity criteria. In the following, we propose a metric from the perspective of network orthogonality, and give a simple and clear derivation. Simultaneously, we use it to guide the network quantization.
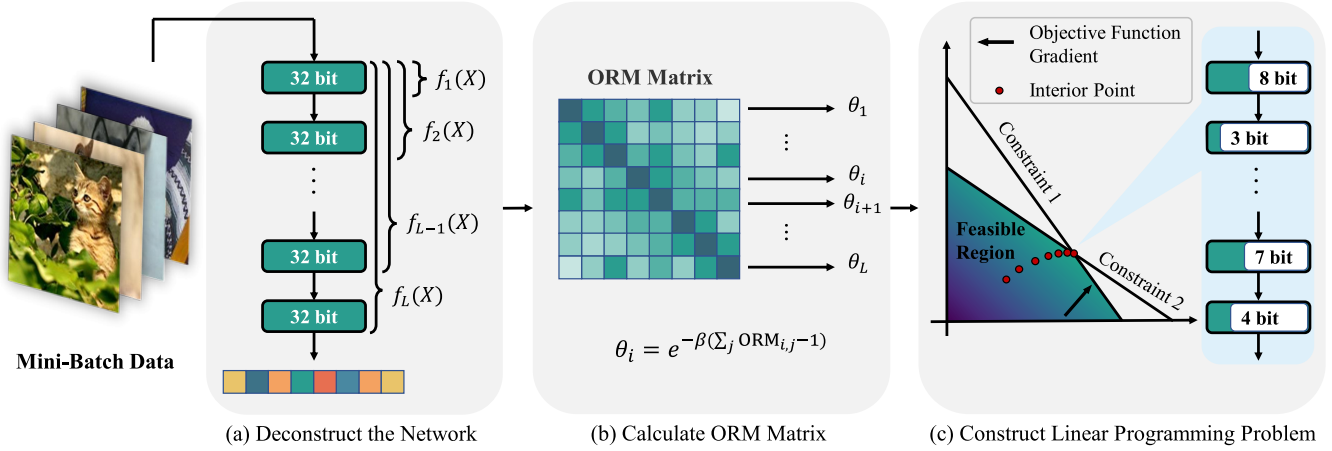
Figure 2: Overview. Left: Deconstruct the model into a set of functions $\mathcal{F}$. Middle: ORM symmetric matrix calculated from $\mathcal{F}$. Right: Linear programming problem constructed by the importance factor θ to derive optimal bit configuration.

Methodology

In this section, we will introduce our mixed precision quantization algorithm from three aspects: how to define the orthogonality, how to efficiently measure it, and how to construct a linear programming model to derive the optimal bit configuration.

Network Orthogonality

A neural network can be naturally decomposed into a set of layers or functions. Formally, for the given input $x \in \mathbb{R}^{1 \times (C \times H \times W)}$, we decompose a neural network into $\mathcal{F} = \{f_1, f_2, \cdots, f_L\}$, where f_i represents the transformation from input x to the result of the i-th layer. In other words, if g_i represents the function of the i-th layer, then $f_i(x) = g_i\big(f_{i-1}(x)\big) = g_i\Big(g_{i-1}\big(\cdots g_1(x)\big)\Big)$. Here we introduce the inner product (Arfken, Weber, and Spector 1999) between functions f_i and f_j, which is formally defined as,

$$\langle f_i, f_j \rangle_{P(x)} = \int_{\mathcal{D}} f_i(x) P(x) f_j(x)^T dx, \quad (1)$$

where $f_i(x) \in \mathbb{R}^{1 \times (C_i \times H_i \times W_i)}$, $f_j(x) \in \mathbb{R}^{1 \times (C_j \times H_j \times W_j)}$ are the known functions when the model is given, and $\mathcal{D}$ is the domain of x. If we set $f_i^{(m)}(x)$ to be the m-th element of $f_i(x)$, then $P(x) \in \mathbb{R}^{(C_i \times H_i \times W_i) \times (C_j \times H_j \times W_j)}$ is the probability density matrix between $f_i(x)$ and $f_j(x)$, where $P_{m,n}(x)$ is the probability density function of the random variable $f_i^{(m)}(x) \cdot f_j^{(n)}(x)$. According to the definition in (Arfken, Weber, and Spector 1999), $\langle f_i, f_j \rangle_{P(x)} = 0$ means that f_i and f_j are weighted orthogonal. In other words, $\langle f_i, f_j \rangle_{P(x)}$ is negatively correlated with the orthogonality between f_i and f_j. When we have a known set of functions to be quantized $\mathcal{F} = \{f_i\}_{i=1}^L$, with the goal to approximate an arbitrary function h^*, the quantization error can then be expressed by the mean square error: $\xi \int_{\mathcal{D}} |h^*(x) - \sum_i \psi_i f_i(x)|^2 dx$, where ξ and ψ_i are combination coefficient. According to Parseval equality (Tanton

2005), if $\mathcal{F}$ is an orthogonal basis functions set, then the mean square error could achieve 0. Furthermore, the orthogonality between the basis functions is stronger, the mean square error is smaller, i.e., the model corresponding to the linear combination of basis functions has a stronger representation capability. Here we further introduce this insight to network quantization. In general, the larger the bit, the more representational capability of the corresponding model (Liu et al. 2018). Specifically, we propose to assign a larger bit-width to the layer with stronger orthogonality against all other layers to maximize the representation capability of the model. However, Eq. 1 has the integral of a continuous function which is untractable in practice. Therefore, we derive a novel metric to efficiently approximate the orthogonality of each layer in the next section.

Efficient Orthogonality Metric

To avoid the intractable integral, we propose to leverage the Monte Carlo sampling to approximate the orthogonality of the layers. Specifically, from the Monte Carlo integration perspective in (Caflisch 1998), Eq. 1 can be rewritten as

$$\langle f_i, f_j \rangle_{P(x)} = \int_{\mathcal{D}} f_i(x) P(x) f_j(x)^T dx$$
$$= \left\| E_{P(x)}[f_j(x)^T f_i(x)] \right\|_F. \quad (2)$$

We randomly get N samples $x_1, x_2, \ldots, x_N$ from a training dataset with the probability density matrix $P(x)$, which allows the expectation $E_{P(x)}[f_j(x)^T f_i(x)]$ to be further approximated as,

$$\left\| E_{P(x)}[f_j(x)^T f_i(x)] \right\|_F \approx \frac{1}{N} \left\| \sum_{n=1}^{N} f_j(x_n)^T f_i(x_n) \right\|_F$$
$$= \frac{1}{N} \left\| f_j(X)^T f_i(X) \right\|_F, \quad (3)$$

where $f_i(X) \in \mathbb{R}^{N \times (C_i \times H_i \times W_i)}$ represents the output of the i-th layer, $f_j(X) \in \mathbb{R}^{N \times (C_j \times H_j \times W_j)}$ represents the out-

put of the j-th layer, and $||\cdot||_F$ is the Frobenius norm. From Eqs. 2-3, we have

$$N \int_{\mathcal{D}} f_i(x) P(x) f_j(x)^T dx \approx \left\| f_j(X)^T f_i(X) \right\|_F. \quad (4)$$

However, the comparison of orthogonality between different layers is difficult due to the differences in dimensionality. To this end, we use the Cauchy-Schwarz inequality to normalize it in $[0, 1]$ for the different layers. Applying Cauchy-Schwarz inequality to the left side of Eq. 4, we have

$$0 \leq \left(N \int_{\mathcal{D}} f_i(x) P(x) f_j(x)^T dx \right)^2$$
$$\leq \int_{\mathcal{D}} N f_i(x) P_i(x) f_i(x)^T dx \int_{\mathcal{D}} N f_j(x) P_j(x) f_j(x)^T dx. \quad (5)$$

We substitute Eq. 4 into Eq. 5 and perform some simplifications to derive our **OR**thogonality **M**etric (ORM) [1], refer to **supplementary material** for details:

$$\mathrm{ORM}(X, f_i, f_j) = \frac{||f_j(X)^T f_i(X)||_F^2}{||f_i(X)^T f_i(X)||_F ||f_j(X)^T f_j(X)||_F}, \quad (6)$$

where $\mathrm{ORM} \in [0, 1]$. f_i and f_j is orthogonal when $\mathrm{ORM} = 0$. On the contrary, f_i and f_j is dependent when $\mathrm{ORM} = 1$. Therefore, ORM is negatively correlated to orthogonality.

Calculation Acceleration. Given a specific model, calculating Eq. 6 involves the huge matrices. Suppose that $f_i(X) \in \mathbb{R}^{N \times (C_i \times H_i \times W_i)}$, $f_j(X) \in \mathbb{R}^{N \times (C_j \times H_j \times W_j)}$, and the dimension of features in the j-th layer is larger than that of the i-th layer. Furthermore, the time complexity of computing $\mathrm{ORM}(X, f_i, f_j)$ is $O(NC_j^2 H_j^2 W_j^2)$. The huge matrix occupies a lot of memory resources, and also increases the time complexity of the entire algorithm by several orders of magnitude. Therefore, we derive an equivalent form to accelerate calculation. If we take $Y = f_i(X)$, $Z = f_j(X)$ as an example, then $YY^T, ZZ^T \in \mathbb{R}^{N \times N}$. We have:

$$||Z^T Y||_F^2 = \left\langle \mathbf{vec}(YY^T), \mathbf{vec}(ZZ^T) \right\rangle, \quad (7)$$

where $\mathbf{vec}(\cdot)$ represents the operation of flattening matrix into vector. From Eq. 7, the time complexity of calculating $\mathrm{ORM}(X, f_i, f_j)$ becomes $O(N^2 C_j H_j W_j)$ through the inner product of vectors. When the number of samples N is larger than the dimension of features $C \times H \times W$, the norm form is faster to calculate thanking to lower time complexity, vice versa. Specific acceleration ratio and the proof of Eq. 7 are demonstrated in **supplementary material**.

---

[1]ORM is formally consistent with CKA. However, we pioneer to discover its relationship with quantized model accuracy and confirm its validity in mixed precision quantization from the perspective of function orthogonality, and CKA explores the relationship between hidden layers from the perspective of similarity. In other words, CKA implicitly verifies the validity of ORM further.
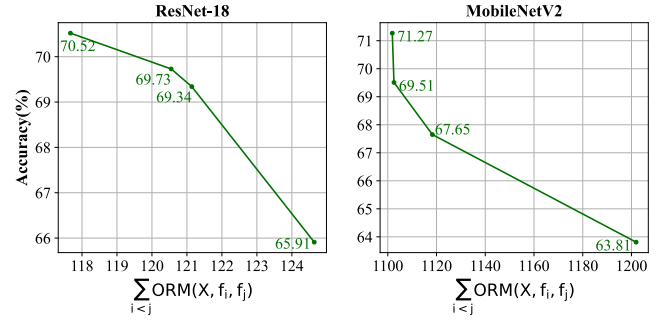

Figure 3: Relationship between orthogonality and accuracy for different quantization configurations on ResNet-18 and MobileNetV2.

Mixed Precision Quantization

Effectiveness of ORM on Mixed Precision Quantization. ORM directly indicates the importance of the layer in the network, which can be used to decide the configuration of the bit-width eventually. We conduct extensive experiments to provide sufficient and reliable evidence for such claim. Specifically, we first sample different quantization configurations for ResNet-18 and MobileNetV2. Then finetuning to obtain the performance. Meanwhile, the overall orthogonality of the sampled models is calculated separately. Interestingly, we find that model orthogonality and performance are positively correlated to the sum of ORM in Fig. 3. Naturally, inspired by this finding, maximizing orthogonality is taken as our objective function, which is employed to integrate the model size constraints and construct a linear programming problem to obtain the final bit configuration. The detailed experiments are provided in the **supplementary material**.

For a specific neural network, we can calculate an orthogonality matrix K, where $k_{ij} = \mathrm{ORM}(X, f_i, f_j)$. Obviously, K is a symmetric matrix and the diagonal elements are 1. Furthermore, we show some ORM matrices on widely used models with the different number of samples N in the **supplementary material**. We add up the non-diagonal elements of each row of the matrix,

$$\gamma_i = \sum_{j=1}^{L} k_{ij} - 1. \quad (8)$$

Smaller γ_i means stronger orthogonality between f_i and other functions in the function set $\mathcal{F}$, and it also means that former i layers of the neural network are more independent. Thus, we leverage the monotonically decreasing function e^{-x} to model this relationship:

$$\theta_i = e^{-\beta \gamma_i}, \quad (9)$$

where β is a hyper-parameter to control the bit-width difference between different layers. We also investigate the other monotonically decreasing functions (For the details, please refer to the ablation study). θ_i is used as the importance factor for the former i layers of the network, then we define a linear programming problem as follows:

Decreasing Function	ResNet-18 (%)	MobileNetV2 (%)	Changing Rate
e^{-x}	72.30	**63.51**	e^{-x}
$-logx$	72.26	63.20	x^{-2}
$-x$	**72.36**	63.0	0
$-x^3$	71.71	-	$6x$
$-e^x$	-	-	e^x

Table 1: The Top-1 accuracy (%) with different monotonically decreasing functions on ResNet-18 and MobileNetV2.

Model	W bit	Layer	Block	Stage	Net
ResNet-18	5*	72.51	**72.52**	72.47	72.31
MobileNetV2	3*	**69.37**	69.10	68.86	63.99

Table 2: Top-1 accuracy (%) of different deconstruction granularity. The activations bit-width of MobileNetV2 and ResNet-18 are both 8. * means mixed bit.

$$\text{Objective: } \max_{\mathbf{b}} \sum_{i=1}^{L} \left(\frac{b_i}{L-i+1} \sum_{j=i}^{L} \theta_j \right),$$

$$\text{Constraints: } \sum_{i}^{L} M^{(b_i)} \leq \mathcal{T}. \tag{10}$$

$M^{(b_i)}$ is the model size of the i-th layer under b_i bit quantization and $\mathcal{T}$ represents the target model size. $\mathbf{b}$ is the optimal bit configuration. Maximizing the objective function means assigning the larger bit-width to more independent layer, which implicitly maximizes the model's representation capability. More details of network deconstruction, linear programming construction and the impact of β are provided in the **supplementary material**.

Note that it is extremely efficient to solve the linear programming problem in Eq. 10, which only takes a few seconds on a single CPU. In other words, our method is extremely efficient (9s on MobileNetV2) when comparing to the previous methods (Yang and Jin 2021; Dong et al. 2020; Li et al. 2021) that require lots of data or iterations for searching. In addition, our algorithm can be combined as a plug-and-play module with quantization-aware training or post-training quantization schemes thanking to the high efficiency and low data requirements. In other words, our approach is capable of improving the accuracy of SOTA methods, where detail results are reported in the next section.

Experiments

In this section, we conduct a series of experiments to validate the effectiveness of OMPQ on ImageNet. We first introduce the implementation details of our experiments. Ablation experiments about the monotonically decreasing function and deconstruction granularity are then conducted to demonstrate the importance of each component. Finally, we

combine OMPQ with widely-used QAT and PTQ schemes, which shows a better compression and the accuracy trade-off comparing to the SOTA methods.

Implementation Details

The ImageNet dataset includes 1.2M training data and 50,000 validation data. We randomly obtain 64 training data samples for ResNet-18/50 and 32 training data samples for MobileNetV2 following similar data pre-processing (He et al. 2016) to derive the set of functions $\mathcal{F}$. OMPQ is extremely efficient which only needs a piece of Nvidia Geforce GTX 1080Ti and a single Intel(R) Xeon(R) CPU E5-2620 v4. For the models that have a large amount of parameters, we directly adopt the round function to convert the bit-width into an integer after linear programming. Meanwhile, we adopt depth-first search (DFS) to find the bit configuration that strictly meets the different constraints for a small model, *e.g.* ResNet-18. The aforementioned processes are extremely efficient, which only take a few seconds on these devices. Besides, OMPQ is flexible, which is capable of leveraging different search spaces with QAT and PTQ under different requirements. Finetuning implementation details are listed as follows.

For the experiments on QAT quantization scheme, we use two NVIDIA Tesla V100 GPUs. Our quantization framework does not contain integer division or floating point numbers in the network. In the training process, the initial learning rate is set to 1e-4, and the batch size is set to 128. We use cosine learning rate scheduler and SGD optimizer with 1e-4 weight decay during 90 epochs without distillation. We fix the weight and activation of first and last layer at 8 bit following previous works, where the search space is 4-8 bit.

For the experiments on PTQ quantization scheme, we perform OMPQ on an NVIDIA Geforce GTX 1080Ti and combine it with the finetuning block reconstruction algorithm BRECQ. In particular, the activation precision of all layers are fixed to 8 bit. In other words, only the weight bit is searched, which is allocated in the 2-4 bit search space.

Ablation Study

Monotonically Decreasing Function. We then investigate the monotonically decreasing function in Eq. 9. Obviously, the second-order derivatives of monotonically decreasing functions in Eq. 9 influence the changing rate of orthogonality differences. In other words, the variance of the orthogonality between different layers becomes larger as the rate becomes faster. We test the accuracy of five different monotonically decreasing functions on quantization-aware training of ResNet-18 (6.7Mb) and post-training quantization of MobileNetV2 (0.9Mb). We fix the activation to 8 bit.

It can be observed from Table 1 that the accuracy gradually decreases with the increasing of changing rate. For the corresponding bit configuration, we also observe that a larger changing rate also means a more aggressive bit allocation strategy. In other words, OMPQ tends to assign more different bits between layers under a large changing rate, which leads to worse performance in network quantization. Another interesting observation is the accuracy on ResNet-18 and MobileNetV2. Specifically, quantization-aware train-

(a) ResNet-18

Method	W/A	Int-Only	Uniform	Model Size (Mb)	BOPs (G)	Top-1 (%)
Baseline	32/32	✗	-	44.6	1,858	73.09
RVQuant	8/8	✗	✗	11.1	116	70.01
HAWQ-V3	8/8	✔	✔	11.1	116	71.56
OMPQ	*8	✔	✔	6.7	97	**72.30**
PACT$^\nabla$	5/5	✗	✔	7.2	74	69.80
LQ-Nets$^\nabla$	4/32	✗	✗	5.8	225	70.00
HAWQ-V3	*/*	✔	✔	6.7	72	70.22
OMPQ	*/6	✔	✔	6.7	75	**72.08**

(b) ResNet-50

Method	W/A	Int-Only	Uniform	Model Size (Mb)	BOPs (G)	Top-1 (%)
Baseline	32/32	✗	-	97.8	3,951	77.72
PACT$^\nabla$	5/5	✗	✔	16.0	133	**76.70**
LQ-Nets$^\nabla$	4/32	✗	✗	13.1	486	76.40
RVQuant	5/5	✗	✗	16.0	101	75.60
HAQ	*/32	✗	✗	9.62	520	75.48
OneBitwidth	*/8	✗	✔	12.3	494	**76.70**
HAWQ-V3	*/*	✔	✔	18.7	154	75.39
OMPQ	*/5	✔	✔	16.0	141	76.20
OMPQ	*/5	✔	✔	18.7	156	76.28

Table 3: Mixed precision quantization results of ResNet-18 and ResNet-50. "Int-Only" means only including integer during quantization process. "Uniform" represents uniform quantization. W/A is the bit-width of weight and activation. * indicates mixed precision. ∇ represents not quantizing the first and last layers.

ing on ResNet-18 requires numerous data, which makes the change of accuracy insignificant. On the contrary, post-training quantization on MobileNetV2 is incapable of assigning bit configuration that meets the model constraints when the functions are set to $-x^3$ or $-e^x$. To this end, we select e^{-x} as our monotonically decreasing function in the following experiments.

Deconstruction Granularity. We study the impact of different deconstruction granularity on model accuracy. Specifically, we test four different granularity including layer-wise, block-wise, stage-wise and net-wise on the quantized-aware training of ResNet-18 and the post-training quantization of MobileNetV2. As reported in Table 2, the accuracy of the two models is increasing with finer granularities. Such difference is more significant on MobileNetV2 due to the different sensitiveness between the point-wise and depth-wise convolution. We thus employ layer-wise granularity in the following experiments.

Quantization-Aware Training

We perform quantization-aware training on ResNet-18/50, where the results and compress ratio are compared with the previous unified quantization methods (Park, Yoo, and Vajda 2018; Choi et al. 2018; Zhang et al. 2018a) and

mixed precision quantization (Wang et al. 2019; Chin et al. 2020; Yao et al. 2021). As shown in Table 3, OMPQ shows the best trade-off between accuracy and compress ratio on ResNet-18/50. For example, we achieve 72.08% on ResNet-18 with only 6.7Mb and 75BOPs. Comparing with HAWQ-V3(Yao et al. 2021), the difference of the model size is negligible (6.7Mb, 75BOPs vs 6.7Mb, 72BOPs). Meanwhile, the model compressed by OMPQ is 1.86% higher than HAWQ-V3(Yao et al. 2021). Similarly, we achieve 76.28% on ResNet-50 with 18.7Mb and 156BOPs. OMPQ is 0.89% higher than HAWQ-V3 with similar model size (18.7Mb, 156BOPs vs 18.7Mb, 154BOPs).

Post-Training Quantization

As we mentioned before, OMPQ can also be combined with PTQ scheme to further improve the quantization efficiency thanking to its low data dependence and search efficiency. Previous PTQ method BRECQ (Li et al. 2021) proposes block reconstruction quantization strategy to reduce quantization errors. We replace the evolutionary search algorithm with OMPQ and combine it with the finetuning process of BRECQ, which rapidly reduces the search cost and achieves better performance. Experiment results are demonstrated in Table 4, we can observe that OMPQ clearly shows the su-

(a) ResNet-18

Method	W/A	Model Size (Mb)	Top-1 (%)	Searching Data	Searching Iterations
Baseline	32/32	44.6	71.08	-	-
FracBits-PACT	*/*	4.5	69.10	1.2M	120
OMPQ	*/4	4.5	68.69	**64**	**0**
OMPQ	*/8	4.5	**69.94**	**64**	**0**
ZeroQ	4/4	5.81	21.20	-	-
BRECQ[†]	4/4	5.81	69.32	-	-
PACT	4/4	5.81	69.20	-	-
HAWQ-V3	4/4	5.81	68.45	-	-
FracBits-PACT	*/*	5.81	**69.70**	1.2M	120
OMPQ	*/4	5.5	69.38	**64**	**0**
BRECQ	*/8	4.0	68.82	1,024	100
OMPQ	*/8	4.0	**69.41**	**64**	**0**

(b) MobileNetV2

Method	W/A	Model Size (Mb)	Top-1 (%)	Searching Data	Searching Iterations
Baseline	32/32	13.4	72.49	-	-
BRECQ	*/8	1.3	68.99	1,024	100
OMPQ	*/8	1.3	**69.62**	**32**	**0**
FracBits	*/*	1.84	69.90	1.2M	120
BRECQ	*/8	1.5	70.28	1,024	100
OMPQ	*/8	1.5	**71.39**	**32**	**0**

Table 4: Mixed precision post-training quantization experiments on ResNet-18 and MobileNetV2. † means using distilled data in the finetuning process.
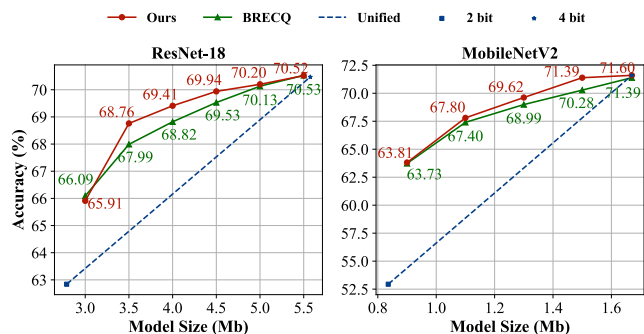

Figure 4: Mixed precision quantization comparison of OMPQ and BRECQ on ResNet-18 and MobileNetV2.

perior performance to unified quantization and mixed precision quantization methods under different model constraints. In particular, OMPQ outperforms BRECQ by 0.52% on ResNet-18 under the same model size (4.0Mb). OMPQ also outperforms FracBits by 1.37% on MobileNetV2 with a smaller model size (1.5Mb vs 1.8Mb).

We also compare OMPQ with BRECQ and unified quantization, where the results are reported in Fig. 4. Obviously,

the accuracy of OMPQ is generally higher than BRECQ on ResNet-18 and MobileNetV2 with different model constraints. Furthermore, OMPQ and BRECQ are both better than unified quantization, which shows that mixed precision quantization is superior.

Conclusion

In this paper, we have proposed a novel mixed precision algorithm, termed OMPQ, to effectively search the optimal bit configuration on the different constraints. Firstly, we derive the orthogonality metric of neural network by generalizing the orthogonality of the function to the neural network. Secondly, we leverage the proposed orthogonality metric to design a linear programming problem, which is capable of finding the optimal bit configuration. Both orthogonality generation and linear programming solving are extremely efficient, which are finished within a few seconds on a single CPU and GPU. Meanwhile, OMPQ also outperforms the previous mixed precision quantization and unified quantization methods. Furthermore, we will explore the mixed precision quantization method combining multiple knapsack problem with the network orthogonality metric.

Acknowledgments

References

Arfken, G. B.; Weber, H. J.; and Spector, D. 1999. Mathematical Methods for Physicists. *American Journal of Physics*, 67(2): 165–169.

Bach, F. R.; and Jordan, M. I. 2002. Kernel Independent Component Analysis. *Journal of Machine Learning Research (JMLR)*, 3: 1–48.

Banner, R.; Nahshan, Y.; and Soudry, D. 2019. Post training 4-bit quantization of convolutional networks for rapid-deployment. *Neural Information Processing Systems(NeurIPS)*, 32.

Bengio, Y.; Léonard, N.; and Courville, A. 2013. Estimating or Propagating Gradients through Stochastic Neurons for Conditional Computation. *arXiv preprint arXiv:1308.3432*.

Caflisch, R. E. 1998. Monte carlo and quasi-monte carlo methods. *Acta numerica*, 7: 1–49.

Cai, Y.; Yao, Z.; Dong, Z.; Gholami, A.; Mahoney, M. W.; and Keutzer, K. 2020. Zeroq: A Novel Zero Shot Quantization Framework. In *Computer Vision and Pattern Recognition (CVPR)*, 13169–13178.

Cai, Z.; He, X.; Sun, J.; and Vasconcelos, N. 2017. Deep Learning with Low Precision by Half-Wave Gaussian Quantization. In *Computer Vision and Pattern Recognition (CVPR)*, 5918–5926.

Chen, S.; Wang, W.; and Pan, S. J. 2019. Metaquant: Learning to Quantize by Learning to Penetrate Non-Differentiable Quantization. *Neural Information Processing Systems(NeurIPS)*, 32: 3916–3926.

Chin, T.-W.; Pierce, I.; Chuang, J.; Chandra, V.; and Marculescu, D. 2020. One Weight Bitwidth to Rule Them All. In *European Conference on Computer Vision (ECCV)*, 85–103.

Choi, J.; Wang, Z.; Venkataramani, S.; Chuang, P. I.-J.; Srinivasan, V.; and Gopalakrishnan, K. 2018. Pact: Parameterized Clipping Activation for Quantized Neural Networks. *arXiv preprint arXiv:1805.06085*.

Courbariaux, M.; Hubara, I.; Soudry, D.; El-Yaniv, R.; and Bengio, Y. 2016. Binarized Neural Networks: Training Deep Neural Networks with Weights and Activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830*.

Dong, Z.; Yao, Z.; Arfeen, D.; Gholami, A.; Mahoney, M. W.; and Keutzer, K. 2020. HAWQ-V2: Hessian Aware Trace-Weighted Quantization of Neural Networks. In

Neural Information Processing Systems(NeurIPS), 18518–18529.

Fukumizu, K.; Bach, F. R.; and Jordan, M. I. 2004. Dimensionality Reduction for Supervised Learning with Reproducing Kernel Hilbert Spaces. *Journal of Machine Learning Research (JMLR)*, 5: 73–99.

Gong, R.; Liu, X.; Jiang, S.; Li, T.; Hu, P.; Lin, J.; Yu, F.; and Yan, J. 2019. Differentiable soft quantization: Bridging full-precision and low-bit neural networks. In *International Conference on Computer Vision (ICCV)*, 4852–4861.

Gretton, A.; Bousquet, O.; Smola, A.; and Schölkopf, B. 2005. Measuring Statistical Dependence with Hilbert-Schmidt Norms. In *Algorithmic Learning Theory (ALT)*, 63–77.

Gretton, A.; Herbrich, R.; and Smola, A. J. 2003. The Kernel Mutual Information. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IV–880.

Guo, Z.; Zhang, X.; Mu, H.; Heng, W.; Liu, Z.; Wei, Y.; and Sun, J. 2020. Single Path One-Shot Neural Architecture Search with Uniform Sampling. In *European Conference on Computer Vision (ECCV)*, 544–560.

Han, S.; Mao, H.; and Dally, W. J. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *Computer Vision and Pattern Recognition (CVPR)*, 770–778.

Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. Mobilenets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv preprint arXiv:1704.04861*.

Kim, H.; Kim, K.; Kim, J.; and Kim, J.-J. 2019. BinaryDuo: Reducing Gradient Mismatch in Binary Activation Network by Coupling Binary Activations. In *International Conference on Learning Representations (ICLR)*.

Kornblith, S.; Norouzi, M.; Lee, H.; and Hinton, G. 2019. Similarity of Neural Network Representations Revisited. In *International Conference on Machine Learning (ICML)*, 3519–3529.

Leurgans, S. E.; Moyeed, R. A.; and Silverman, B. W. 1993. Canonical Correlation Analysis when The Data are Curves. *Journal of the Royal Statistical Society: Series B (Methodological)*, 55: 725–740.

Li, H.; Yan, C.; Lin, S.; Zheng, X.; Zhang, B.; Yang, F.; and Ji, R. 2020. Pams: Quantized super-resolution via parameterized max scale. In *European Conference on Computer Vision*, 564–580. Springer.

Li, Y.; Gong, R.; Tan, X.; Yang, Y.; Hu, P.; Zhang, Q.; Yu, F.; Wang, W.; and Gu, S. 2021. {BRECQ}: Pushing the Limit of Post-Training Quantization by Block Reconstruction. In *International Conference on Learning Representations (ICLR)*.

Liu, C.; Ding, W.; Xia, X.; Zhang, B.; Gu, J.; Liu, J.; Ji, R.; and Doermann, D. 2019. Circulant binary convolutional networks: Enhancing the performance of 1-bit dcnns with

circulant back propagation. In *Computer Vision and Pattern Recognition (CVPR)*, 2691–2699.

Liu, Z.; Wu, B.; Luo, W.; Yang, X.; Liu, W.; and Cheng, K.-T. 2018. Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In *European Conference on Computer Vision (ECCV)*, 722–737.

Nagel, M.; Baalen, M. v.; Blankevoort, T.; and Welling, M. 2019. Data-Free Quantization through Weight Equalization and Bias Correction. In *International Conference on Computer Vision (ICCV)*, 1325–1334.

Park, E.; Yoo, S.; and Vajda, P. 2018. Value-Aware Quantization for Training and Inference of Neural Networks. In *European Conference on Computer Vision (ECCV)*, 580–595.

Rastegari, M.; Ordonez, V.; Redmon, J.; and Farhadi, A. 2016. Xnor-net: Imagenet Classification Using Binary Convolutional Neural Networks. In *European Conference on Computer Vision (ECCV)*, 525–542.

Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen, L.-C. 2018. Mobilenetv2: Inverted Residuals and Linear Bottlenecks. In *Computer Vision and Pattern Recognition (CVPR)*, 4510–4520.

Simonyan, K.; and Zisserman, A. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv preprint arXiv:1409.1556*.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going Deeper with Convolutions. In *Computer Vision and Pattern Recognition (CVPR)*, 1–9.

Tanton, J. 2005. *Encyclopedia of Mathematics*. Facts on file.

Wang, K.; Liu, Z.; Lin, Y.; Lin, J.; and Han, S. 2019. Haq: Hardware-Aware Automated Quantization with Mixed Precision. In *Computer Vision and Pattern Recognition (CVPR)*, 8612–8620.

Yang, L.; and Jin, Q. 2021. FracBits: Mixed Precision Quantization via Fractional Bit-Widths. *AAAI Conference on Artificial Intelligence (AAAI)*, 35: 10612–10620.

Yao, Z.; Dong, Z.; Zheng, Z.; Gholami, A.; Yu, J.; Tan, E.; Wang, L.; Huang, Q.; Wang, Y.; Mahoney, M.; et al. 2021. HAWQ-V3: Dyadic Neural Network Quantization. In *International Conference on Machine Learning (ICML)*, 11875–11886.

Zhang, D.; Yang, J.; Ye, D.; and Hua, G. 2018a. Lq-nets: Learned Quantization for Highly Accurate and Compact Deep Neural Networks. In *European Conference on Computer Vision (ECCV)*, 365–382.

Zhang, S.; Jia, F.; Wang, C.; and Wu, Q. 2023. Targeted Hyperparameter Optimization with Lexicographic Preferences Over Multiple Objectives. In *The Eleventh International Conference on Learning Representations*.

Zhang, S.; Zheng, X.; Yang, C.; Li, Y.; Wang, Y.; Chao, F.; Wang, M.; Li, S.; Yang, J.; and Ji, R. 2021. You Only Compress Once: Towards effective and elastic BERT compression via exploit-explore stochastic nature gradient. *arXiv preprint arXiv:2106.02435*.

Zhang, X.; Zhou, X.; Lin, M.; and Sun, J. 2018b. Shufflenet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. In *Computer Vision and Pattern Recognition (CVPR)*, 6848–6856.

Zheng, X.; Fei, X.; Zhang, L.; Wu, C.; Chao, F.; Liu, J.; Zeng, W.; Tian, Y.; and Ji, R. 2022. Neural Architecture Search With Representation Mutual Information. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 11912–11921.

Zheng, X.; Ji, R.; Chen, Y.; Wang, Q.; Zhang, B.; Chen, J.; Ye, Q.; Huang, F.; and Tian, Y. 2021a. MIGO-NAS: Towards fast and generalizable neural architecture search. *IEEE TPAMI*, 43(9): 2936–2952.

Zheng, X.; Ji, R.; Tang, L.; Zhang, B.; Liu, J.; and Tian, Q. 2019. Multinomial distribution learning for effective neural architecture search. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 1304–1313.

Zheng, X.; Ji, R.; Wang, Q.; Ye, Q.; Li, Z.; Tian, Y.; and Tian, Q. 2020. Rethinking Performance Estimation in Neural Architecture Search. In *CVPR*.

Zheng, X.; Ma, Y.; Xi, T.; Zhang, G.; Ding, E.; Li, Y.; Chen, J.; Tian, Y.; and Ji, R. 2021b. An Information Theory-inspired Strategy for Automatic Network Pruning. *arXiv preprint arXiv:2108.08532*.

Zheng, X.; Yang, C.; Zhang, S.; Wang, Y.; Zhang, B.; Wu, Y.; Wu, Y.; Shao, L.; and Ji, R. 2023. DDPNAS: Efficient Neural Architecture Search via Dynamic Distribution Pruning. *International Journal of Computer Vision*, 1–16.

Zheng, X.; Zhang, Y.; Hong, S.; Li, H.; Tang, L.; Xiong, Y.; Zhou, J.; Wang, Y.; Sun, X.; Zhu, P.; et al. 2021c. Evolving fully automated machine learning via life-long knowledge anchors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(9): 3091–3107.

Zhou, A.; Yao, A.; Guo, Y.; Xu, L.; and Chen, Y. 2017. Incremental Network Quantization: Towards Lossless CNNs with Low-Precision Weights. In *International Conference on Learning Representations (ICLR)*.

Zhou, Q.; Zheng, X.; Cao, L.; Zhong, B.; Xi, T.; Zhang, G.; Ding, E.; Xu, M.; and Ji, R. 2021. EC-DARTS: Inducing Equalized and Consistent Optimization Into DARTS. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 11986–11995.

Zhou, S.; Wu, Y.; Ni, Z.; Zhou, X.; Wen, H.; and Zou, Y. 2016. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*.