PD-Quant: Post-Training Quantization Based on Prediction Difference Metric Supplementary Materials

A. More Details of CNN models Implementations

This section will add more experimental details for CNN models. We apply different hyper-parameters λ_r and λ_c for different types of networks. The regularization parameter λ_r is set to 0.2 for ResNet-18 and ResNet-50 [4] and 0.1 for other CNN architectures. Moreover, we set the hyper-parameter λ_c for DC to 0.005 for MobileNetV2 [7], 0.001 for MNasNet [8], and 0.02 for other CNN architectures.

B. Effects on different calibration data sizes

We conduct experiments on 256, 1024, and 4096 calibration data sizes. Tab. 1 shows that PD is effective on calibration data of different sizes. The effect of DC decreases as the size of calibration sets increases because the calibration set's distribution is getting closer to the training set.

Model	**ResNet-18**			**MobileNetV2**		
Size	256	1024	4096	256	1024	4096
QDrop	46.22	51.42	54.48	7.53	10.28	10.88
PD	46.76	52.74	55.30	9.29	13.49	16.47
PD+DC	**47.28**	**53.08**	**55.33**	**9.48**	**14.17**	**16.55**

Table 1. Effects on different calibration dataset sizes for PD-Quant. All the results in the table are quantized to W2A2.

C. PD Loss on Transformer Models

Besides CNN, we further extend the proposed method to Transformer models. We evaluate our PD-Quant on both ViT [2] and DeiT [9] at different bit settings.

C.1. Implementation Details

We keep most parameter settings the same as in CNN, including the learning rate, iterations, and calibration data numbers. However, we set the batch size to 16 and regularization parameters λ_r to 0.1 for Transformer models. We did not apply DC to the quantization of Transformer models because there are no batch normalization layers.

We quantize all the weights and inputs for the fully-connect layers, including the first projection layer and the last head layer. The two input matrices for the matrix multiplications in the self-attention modules are also quantized. Moreover, the inputs of the softmax layers and the normalization layers are not quantized, the same as in previous work [5, 12].

We still take QDrop as the baseline method and define the encoder in Transformer models as the block. Our implementation for Transformer models is based on open-source code, and the pre-trained FP models are all from [11].

C.2. Performance Comparison

We compare our proposed PD-Quant with QDrop [10] and PTQ4ViT [12] for both ViT and DeiT. PQT4ViT is a post-training quantization framework designed for Transformer model quantization. Moreover, it shows the state-of-the-art results among all transformer quantization algorithms in W6A6. We keep the same quantization environment and use the same pre-trained model for comparison.

As seen in Appendix C.1, PD-Quant can improve the results of QDrop, similar to the effects in CNN models. We implemented PTQ4ViT based on open-source code.

D. Optimization of Activation Scaling Factors and Rounding values

QAT method LSQ [3] first optimizes activation scaling factors (S_a) by final objective. Since only limited unlabeled data is available in PTQ, we propose PD loss to optimize S_a. When optimizing only S_a, the gradients are given by

$$\frac{\partial \mathcal{L}_{PD}}{\partial S_a} = \begin{cases} \frac{\partial \mathcal{L}_{PD}}{\partial \tilde{x}} q_{max} & \frac{x}{S_a} \geq q_{max} \\ \frac{\partial \mathcal{L}_{PD}}{\partial \tilde{x}} \left(\lfloor \frac{x}{S_a} \rceil - \frac{x}{S_a} \right) & q_{min} < \frac{x}{S_a} < q_{max} \\ \frac{\partial \mathcal{L}_{PD}}{\partial \tilde{x}} q_{min} & \frac{x}{S_a} \leq q_{min} \end{cases}, \tag{1}$$

where STE [1] calculates the gradients of the round function.

When optimizing rounding values (θ), we follow [6] to adopt a sigmoid-like function $\sigma(\theta)$ deciding weight round-

Model	Method	Bits (W/A)	Acc (%)
ViT-S/16/224 74.65	PTQ4ViT* [12]	W6A6	70.72
	QDrop* [10]		70.25
	PD-Quant		**70.84**
	PTQ4ViT* [12]	W4A6	53.55
	QDrop* [10]		67.57
	PD-Quant		**68.64**
	PTQ4ViT* [12]	W2A6	0.31
	QDrop* [10]		45.16
	PD-Quant		**48.13**
ViT-B/16/224 78.01	PTQ4ViT* [12]	W6A6	74.24
	QDrop* [10]		75.76
	PD-Quant		**75.82**
	PTQ4ViT* [12]	W4A6	52.97
	QDrop* [10]		75.51
	PD-Quant		**75.52**
	PTQ4ViT* [12]	W2A6	0.24
	QDrop* [10]		63.74
	PD-Quant		**64.51**
DeiT-S/16/224 79.71	PTQ4ViT* [12]	W6A6	76.83
	QDrop* [10]		77.95
	PD-Quant		**78.33**
	PTQ4ViT* [12]	W4A6	74.17
	QDrop* [10]		77.66
	PD-Quant		**77.88**
	PTQ4ViT* [12]	W2A6	3.79
	QDrop* [10]		65.76
	PD-Quant		**67.53**

Table 2. Comparison on PD-Quant for Transformer models. * represents our implementation with open-source code. ViT-S/16/224 denotes patch size is 16×16 the input resolution is 224×224. All the results listed are the top-1 accuracy (%).

ing up or down. The minimization problem for θ convergence is given by

$$\arg\min_{\theta} \sum (1 - |2\sigma(\theta) - 1|^{\beta}), \tag{2}$$

where $\sigma(\theta) = 0$ means weight rounds down and $\sigma(\theta) = 1$ means weight rounds up.

References

[1] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. 1

[2] 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. *arXiv preprint arXiv:2010.11929*, 2020. 1

[3] Steven K Esser, Jeffrey L McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S Modha. Learned step size quantization. *arXiv preprint arXiv:1902.08153*, 2019. 1

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[5] Zhenhua Liu, Yunhe Wang, Kai Han, Wei Zhang, Siwei Ma, and Wen Gao. Post-training quantization for vision transformer. *Advances in Neural Information Processing Systems*, 34:28092–28103, 2021. 1

[6] 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*, pages 7197–7206. PMLR, 2020. 1

[7] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 1

[8] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2820–2828, 2019. 1

[9] 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*, pages 10347–10357. PMLR, 2021. 1

[10] Xiuying Wei, Ruihao Gong, Yuhang Li, Xianglong Liu, and Fengwei Yu. Qdrop: Randomly dropping quantization for extremely low-bit post-training quantization. *arXiv preprint arXiv:2203.05740*, 2022. 1, 2

[11] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019. 1

[12] Zhihang Yuan, Chenhao Xue, Yiqi Chen, Qiang Wu, and Guangyu Sun. Ptq4vit: Post-training quantization for vision transformers with twin uniform quantization. In *European Conference on Computer Vision*, pages 191–207. Springer, 2022. 1, 2