NIPQ: Noise proxy-based Integrated Pseudo-Quantization Supplementary Materials

1. Overview

In this supplementary material, we present the details of our implementation and additional experimental results for various tasks and different datasets. We provide the following items:

- The detailed implementation of the cost loss function in Section 2.
- Detailed Configurations of our experiments in Section 3
- The results of quantization for super resolution task in Section 4.
- Additional experimental results of object detection on MS-COCO dataset in Section 5.
- An ablation study on the effect of stochastic rounding in Section 6.
- An ablation study on the effect of late training stage in Section 7.
- Experimental results on quantization parameter robustness in Section 8.
- The visualization of the quantization noise distribution in Section 9

2. Cost Loss Function

In order to restrict the utilization of memory and computation resources, we introduce an additional cost loss function in addition to the target loss, as explained in Equation (4) of the main paper. The cost functions for the memory consumption $\mathcal{L}_{cost-MP}$ and the computation cost $\mathcal{L}_{cost-BOP}$ are defined as follows:

$$\mathcal{L}_{cost-MP} = \lambda_w h(\frac{\Sigma_i \lfloor b_i^w \rceil \cdot e_i^w}{\Sigma_i e_i^w} - b_t) + \lambda_a h(\frac{\Sigma_i \lfloor b_i^a \rceil \cdot e_i^a}{\Sigma_i e_i^a} - b_t), \quad (1)$$

$$\mathcal{L}_{cost-BOP} = \lambda_b h(\Sigma_i \lfloor b_i^w \rceil \cdot \lfloor b_i^a \rceil \cdot OPS_i - b_t), \quad (2)$$

where $h(\cdot)$ denotes Huber loss, b_i^w/b_i^a denote the bit-width of i-th layer's weight/activation, e_i^w/e_i^a are the number of elements in the i-th layer's weight/activation, OPS_i is FLOPS of the i-th layer and b_t denotes the target bit-width. $\mathcal{L}_{cost-MP}$ regularizes the average bit-width of activation/weight to the target bit, and $\mathcal{L}_{cost-BOP}$ regularizes the sum of overall bit-operation (BOPs) to the target BOPs. Note that we utilize the bit-operations (BOPs) as a representative metric to measure the computation cost of a neural network, which is commonly used in many previous studies [5, 6, 23]. However, any arbitrary differentiable function can be used as a drop-in replacement for the cost function, and NIPQ automatically optimizes the layer-wise bit-width to the sweet spot.

On the other hand, while the per-layer (or per-tensor) bit-width also requires rounding operation during forward operation, NIPQ is not applicable for the bit-width because it relies on the statistics of quantization error, but it is improvable to achieve the statistics for the scalar value. To overcome this limitation, we propose to update the bit-width via stochastic rounding with STE (Section 6).

3. Experimental Configuration

In this paper, all experiments are conducted using GPU servers having 8 x NVIDIA GTX3090 with 24 GB VRAM with 2 x AMD 7313 (16 Core 32 T). The number of GPUs is selected to satisfy the minimum requirement of GPU memory for the target task. All of the experiments are implemented based on the PyTorch [16] framework (v1.12.1) [16]. Our source code is also provided. The additional details of training configuration, e.g., optimizer type, initial learning rate, decay policy, etc., are determined depending on the characteristics of applications and provided in the following paragraphs.

Table 1 shows the configurations of ImageNet training for NIPQ results. In this experiment, we apply quantization to every convolution and linear layer, including the first and last layers. One exception is that the input of the first convolution layer is fixed as 8-bit. We use SGD with momentum optimizer and cosine annealing with warmup scheduling for learning rate adjustment [15]. η_{min} is the final LR multiplier of cosine annealing, and λ_w, λ_a, and λ_b are the hyper-parameter of resource constraints for the bit-width of weight, bit-width of activation, and BOPs, re-

Table 1. Fine-tuning configurations of ImageNet classification task.

Configuration		Epoch		SGD		Cosine annealing with warmup		λ		
		Stage-1	Stage-2	LR	Weight decay	Warmup len	η_{min}	λ_w	λ_a	λ_b
ResNet-18	ImageNet	40	3	0.04	1×10^{-5}	3	1×10^{-3}	1	1	1
MobileNet-v2	Cifar100	30	3	0.04	5×10^{-5}	5	1×10^{-3}	1	1	1
	ImageNet	40	3	0.04	1×10^{-5}	3	1×10^{-3}	1	1	3
MobileNet-v3	ImageNet	40	3	0.04	1×10^{-5}	3	1×10^{-3}	1	1	3

Table 2. Fine-tuning configurations of super-resolution task with EDSR.

Configuration		Epoch		Adam		Cosine annealing	λ		
		Stage-1	Stage-2	LR	Weight decay	η_{min}	λ_w	λ_a	λ_b
EDSR 4bit	DIV2K	30	10	1×10^{-4}	0	1×10^{-3}	15	15	-
EDSR 3bit	DIV2K	40	10	1×10^{-4}	0	1×10^{-3}	15	15	-

Table 3. Fine-tuning configurations of object detection task with YoloV5-S.

Configuration		Epoch		SGD		Cosine annealing with warmup		λ		
		Stage-1	Stage-2	LR	Weight decay	Warmup len	η_{min}	λ_w	λ_a	λ_b
YoloV5-S	Pascal VOC	30	5	0.0032	3.6×10^{-4}	5	1×10^{-1}	1	1	-
	COCO	35	5	0.0032	3.6×10^{-4}	5	1×10^{-1}	1	1	-

Table 4. Fine-tuning configurations of GLUE Dataset with BERT-base.

Configuration		Epoch		AdamW		Cosine annealing with warmup		λ		
		Stage-1	Stage-2	LR	Weight decay	Warmup len	η_{min}	λ_w	λ_a	λ_b
BERT-base	GLUE	25	5	1e-5	1×10^{-1}	5	?	1	1	1

spectively. When knowledge distillation is triggered, we use EfficientNet-B0 [18] as a teacher network. We use the conventional dark-knowledge-based distillation [8].

Tables 2 and 3 show the detailed configurations of super-resolution task and object detection task, respectively. In both experiments, we keep the precision of the first and last layers as full-precision and apply low-precision quantization to the rest of the layers. In the super-resolution task, we use Adam optimizer [11] and cosine annealing scheduling for learning rate adjustment. In the object detection task, we use SGD with momentum optimizer and cosine annealing with warmup scheduling for learning rate adjustment. Like the image classification task, η_{min} is the final LR multiplier of cosine annealing, and λ_w, λ_a, and λ_b represent the hyper-parameters of resource constraints for the bit-width of weight, bit-width of activation, and BOPs, respectively.

Table 4 shows the detailed configurations of BERT-base [4] on the GLUE Task dataset. In this experiment, we modified the code from the huggingface-transformer [22] library. We apply weight quantization to every linear layer except the last classification head. Note that we do not quantize activation or word embedding. We use the AdamW optimizer and CosineLR scheduler for fine-tuning BERT except for the bit parameters because we find that AdamW can induce instability during training when the magnitude of the cost loss is too large. We use SGD with momentum optimizer for the bit parameters. Besides, we also find that α and b parameters are not well trained when a single global learning rate is utilized ($1e-5$). For fast and reliable convergence, we use the learning rate of $1e-2$ for bit parameters. In addition, the gradient of α is multiplied $2^b - 1$ times over the global learning rate.

4. Super Resolution Experiments

Network	Method	Dataset							
		Set5		Set14		BSD100		Urban100	
		4bit	3bit	4bit	3bit	4bit	3bit	4bit	3bit
EDSRx2	DoReFa [25]	37.22	37.13	32.82	32.73	31.63	31.57	30.17	30
	TFLite [19]	37.64	37.33	33.24	32.98	31.94	31.76	31.11	30.48
	PACT [2]	37.57	37.36	33.2	32.99	31.93	31.77	31.09	30.57
	PAMS [12]	37.67	36.76	33.2	32.5	31.94	31.38	31.1	29.5
	DDTB [24]	37.72	37.51	**33.35**	33.17	**32.01**	31.89	**31.39**	31.01
	NIPQ	**37.74**	**37.66**	33.29	**33.20**	**32.01**	**31.95**	31.36	**31.13**
EDSRx4	DoReFa [25]	30.91	30.76	27.78	26.66	27.04	26.97	24.73	24.59
	TFLite [19]	31.54	31.05	28.2	27.92	27.31	27.12	25.28	24.85
	PACT [2]	31.32	30.98	28.07	27.87	27.21	27.09	25.05	24.82
	PAMS [12]	31.59	27.25	28.2	25.24	27.32	25.38	25.32	22.76
	DDTB [24]	**31.85**	31.52	**28.39**	28.18	**27.44**	27.3	**25.69**	25.33
	NIPQ	31.73	**31.62**	28.34	**28.25**	27.41	**27.36**	25.56	**25.39**

Table 5. PSNR comparison of quantized EDSR [13] of scale 4 and scale 2

Table 5 shows the quantitative analysis of NIPQ on su-

(a) Ground truth (b) DDTB [24] (c) NIPQ (d) DDTB [24] (e) NIPQ

Figure 1. Qualitative results of super resolution on DIV2K dataset. EDSRx4 is quantized into 3-bit for both weights and activations.

per resolution task, and Figure 1 visualizes the quality of the generated figures. We report PSNR as a quantitative measure, one of the well-known metrics in the area of super resolution. NIPQ outmatches the specialized quantization algorithm for super resolution, DDTB [24], which applies dynamic quantization that adjusts the quantization step size depending on the input data. These experimental results indicate that NIPQ works well in the regression task as well.

5. Additional Experiments on Object Detection

Table 6. mAP comparison of Yolov5-S [10] on COCO dataset [14]

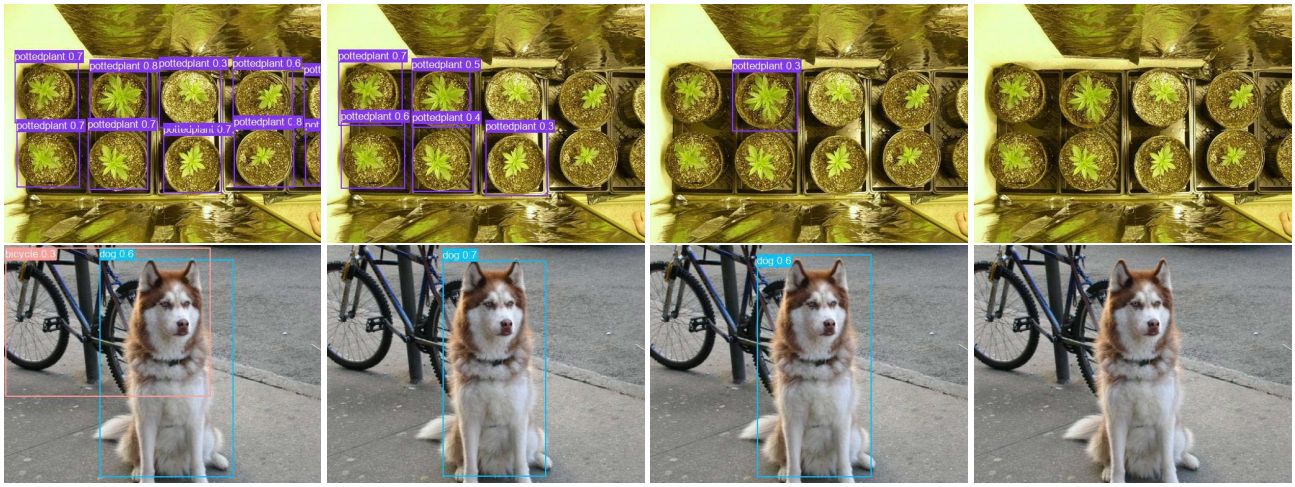
	Bit-width (Weight / Activation)			
	FP/FP	5/5	4/4	3/3
DoReFa [25]	0.354	0.266	0.24	0.191
PACT [2]	0.354	0.313	0.294	0.246
LSQ [7]	0.354	0.32	0.291	0.235
NIPQ	**0.354**	**0.33**	**0.317**	**0.284**

We conduct an additional experiment on object detection task with the COCO dataset and report mAP on Table 6. NIPQ obtains the best results compared to existing quantization studies in the same average bit-width.

In addition, in Figure 2, we visualize the qualitative results of NIPQ on the VOC dataset. Bounding box regression and classification results of the quantized network are presented. As shown in the figure, NIPQ works surprisingly well in the 3-bit domain on the challenging object detection problem. YoloV5-S has a complicated structure, and the sensitivity of each layer is highly different. Because NIPQ has the ability to allocate the bit-width aware of the sensitivity automatically and enable stable convergence without STE instability, the quality of the quantized network outperforms all of the previous methods by a large margin.

6. Stochastic Rounding for Bit-width

While we propose an alternative training scheme for quantization instead of using STE, updating the bit-width is a remaining problem that is not addressed in the NIPQ pipeline. The proposed noise proxy is designed to update the learnable parameters by emulating the quantization operator based on PQN. However, the bit-width is assigned as a scalar value per the target tensor, and thereby it is impossible to aggregate the coarse-grained effect of the quantization operator. When we use rounding-based QAT with STE approximation, the bit-width also suffers from the instability of STE, resulting in highly unreliable result, as shown in Figure 3. Due to this limitation, many previous studies rely on the continuous approximation of bit-width during training [3, 20] to avoid the instability problem. However, the representation mismatches to the domain of bit-width, resulting in suboptimal convergence in practice, especially when the target bit-width is in a sub-4-bit domain. In this paper, we propose an alternative idea to utilize the stochastic rounding of bit-width during training. Stochastic rounding is an unbiased estimator, so the learnable bit-width converges to the optimal point as the learning progresses. In addition, the bit-width is evaluated in the discrete domain during training, which mitigates the domain gap between training and inference. As shown in Figure 3, the stochastic rounding consistently draws the pareto-front line with small variance, which enables us to search for the best quantization configurations within the given resource budget.



(a) NIPQ (b) LSQ [7] (c) PACT [2] (d) DoReFa [25]

Figure 2. Qualitative results of object detection on the VOC dataset. Yolov5-S is quantized into 3-bit weights and activations according to each quantization method.

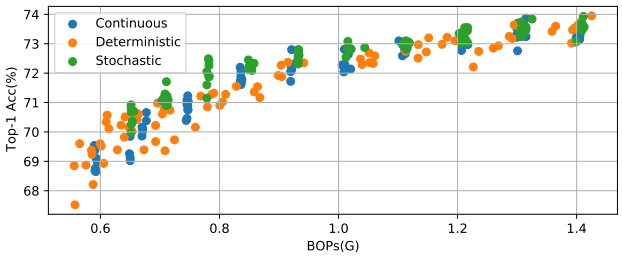


Figure 3. Accuracy comparison of MobileNet-v2 at CIFAR-100 dataset with different bit-width training strategies. We ran the experiments with 10 repetitions, increasing from 0.5 BOPs to 1.4 BOPs by 0.1 BOPs steps.

7. Comparison for the Late Training Stage

Table 7. Comparison of accuracy regarding the late training stage. MobileNet-v2 is trained in 30 epochs and finetuned in 3 epochs on the CIFAR-100 dataset. The target computation overhead is 1.0 GBlops

	FP	Without Tuning	BN update	QAT finetune
Top-1	75.04	70.45	72.99	73.29

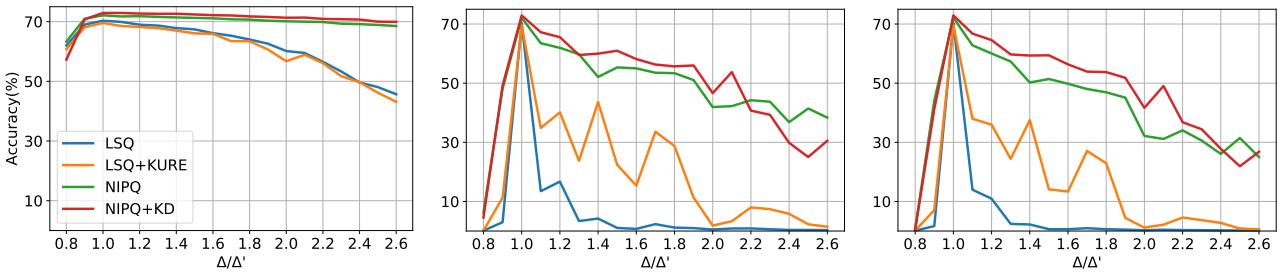
In Table 7, we show the results of NIPQ with different late training stage policies. As shown in the table, BN update offers a large performance benefit compared to the accuracy of the NIPQ training without the late stage tuning. Because PQN of NIPQ disturbs the statistics of normalization layers, the correction of the statistics is essential to maximize the accuracy in the inference phase. In addition, QAT finetune offers an additional performance improvement by giving an additional chance to adjust the learnable parameters of the entire network without the effect of PQN with a small learning rate, which enables the stabilization of network parameters near the optimal point with the tiniest effect of STE instability.

8. Robustness of the quantization parameters

NIPQ also enhances the robustness of the quantization parameters as well as the network parameter. Figure 4 visualizes the results of measuring the accuracy while changing the quantization step size or the truncation interval. The more robust the network, the more it can endure the change of the quantization configuration. As shown in the figure, NIPQ shows comparable or superior results to the previous best algorithm for robustness, KURE [17]. It is especially worthy that existing studies have focused on improving the robustness of weight only [1, 17], but NIPQ also improves the robustness of activation as well by a large margin. To the best of our knowledge, this is for the first time that activation robustness can be improved, which is a crucial benefit of deploying networks in a noisy environment.

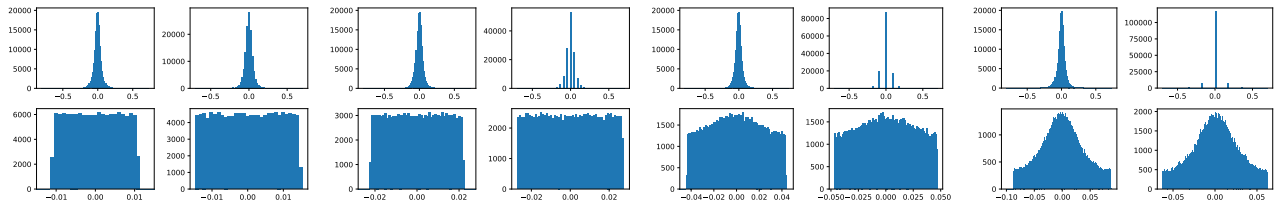
9. Quantization Noise Distribution

In Figure 5, we visualize the quantization noise distribution of ResNet-18's 12-th convolution weight in different bit-widths. When applying quantization, not only rounding but also truncation is applied. The previous study argues that the quantization noise distribution follows a uniform distribution regardless of input distribution when the number of bits is sufficiently large [21]. According to our observation, the statement is held in practice when the bit-width is larger than 4-bit. However, in sub-4-bit precision, the



(a) Activation (b) Weight (c) Activation & Weight

Figure 4. Robustness of quantized MobileNet-V2 on ImageNet against the change of α of the quantization operator for weight and activation. Δ' is the trained α and Δ is the scaled one. NIPQ+KD represents the quantized network with knowledge distillation [9] using EfficientNet-B0 [18] as a teacher.



(a) 5-bit (b) 4-bit (c) 3-bit (d) 2-bit

Figure 5. Quantization noise distribution of ResNet-18's 12-th convolution weight. (left top) 32 bit (FP) distribution. (right top) N-bit uniform quantized distribution. (left bottom) Real quantization noise distribution. (right bottom) Sampled quantization noise distribution.

distribution of noise seems to follow a bell-shaped curve instead of a uniform distribution. Due to these characteristics, conventionally uniform or gaussian distributions are often used to approximate PQN [3, 20]. However, as presented in this paper, the precise sampling of PQN following the quantization error distribution is essential to guarantee the convergence on the optimal point, while the uniform distribution shows comparable results in practice empirically. In this work, we realize the sampling process of quantization error distribution on GPU with practical performance as follows: first, the probability density function (PDF) of the quantization error distribution is estimated based on the histogram with 256 bins. Then, the distribution is sampled from the estimated PDF of the histogram. As shown in Figure 5, the sampled distribution precisely follows the quantization error distribution.

References

[1] Milad Alizadeh, Arash Behboodi, Mart van Baalen, Christos Louizos, Tijmen Blankevoort, and Max Welling. Gradient $\ell 1$ regularization for quantization robustness. *International Conference on Learning Representations (ICLR)*, 2020. 4

[2] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. PACT: parameterized clipping activation for quantized neural networks. *CoRR*, abs/1805.06085, 2018. 2, 3, 4

[3] Alexandre Défossez, Yossi Adi, and Gabriel Synnaeve. Differentiable model compression via pseudo quantization noise. *CoRR*, abs/2104.09987, 2021. 3, 5

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 2

[5] Zhen Dong, Zhewei Yao, Daiyaan Arfeen, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. HAWQ-V2:hessian aware trace-weighted quantization of neural networks. *Neural Information Processing Systems (NeurIPS)*, 2020. 1

[6] Zhen Dong, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. HAWQ: hessian aware quantization of neural networks with mixed-precision. *International Conference on Computer Vision, (ICCV)*, 2019. 1

[7] Steven K. Esser, Jeffrey L. McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S. Modha. Learned step size quantization. *International Conference on Learning Representations (ICLR)*, 2020. 3, 4

[8] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Dark knowledge. *Presented as the keynote in BayLearn*, 2(2), 2014. 2

[9] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015. 5

[10] Glenn Jocher, Alex Stoken, Jirka Borovec, Liu Changyu, Adam Hogan, L Diaconu, F Ingham, J Poznanski, J Fang, L Yu, et al. ultralytics/yolov5: v3. 1-bug fixes and performance improvements. *Zenodo*, 2020. 3

[11] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2

[12] Huixia Li, Chenqian Yan, Shaohui Lin, Xiawu Zheng, Baochang Zhang, Fan Yang, and Rongrong Ji. PAMS: quantized super-resolution via parameterized max scale. *European Conference on Computer Vision (ECCV)*, 2020. 2

[13] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 1132–1140. IEEE Computer Society, 2017. 2

[14] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Doll'a r, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 3

[15] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 1

[16] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 1

[17] Moran Shkolnik, Brian Chmiel, Ron Banner, Gil Shomron, Yury Nahshan, Alex M. Bronstein, and Uri C. Weiser. Robust quantization: One model to rule them all. *Neural Information Processing Systems (NeurIPS)*, 2020. 4

[18] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 2019. 2, 5

[19] Andrew Tulloch and Yangqing Jia. Quantization and training of neural networks for efficient integer-arithmetic-only inference. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[20] Ying Wang, Yadong Lu, and Tijmen Blankevoort. Differentiable joint pruning and quantization for hardware efficiency. *European Conference on Computer Vision (ECCV)*, 2020. 3, 5

[21] Bernard Widrow, Istvan Kollar, and Ming-Chang Liu. Statistical theory of quantization. *IEEE Transactions on instrumentation and measurement*, 45(2):353–361, 1996. 4

[22] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, Oct. 2020. Association for Computational Linguistics. 2

[23] Zhewei Yao, Zhen Dong, Zhangcheng Zheng, Amir Gholami, Jiali Yu, Eric Tan, Leyuan Wang, Qijing Huang, Yida Wang, Michael W. Mahoney, and Kurt Keutzer. HAWQ-V3: dyadic neural network quantization. *Proceedings of the 38th International Conference on Machine Learning, ICML*, 2021. 1

[24] Yunshan Zhong, Mingbao Lin, Xunchao Li, Ke Li, Yunhang Shen, Fei Chao, Yongjian Wu, and Rongrong Ji. Dynamic dual trainable bounds for ultra-low precision super-resolution networks. *arXiv preprint arXiv:2203.03844*, 2022. 2, 3

[25] Shuchang Zhou, Zekun Ni, Xinyu Zhou, He Wen, Yuxin Wu, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *CoRR*, abs/1606.06160, 2016. 2, 3, 4