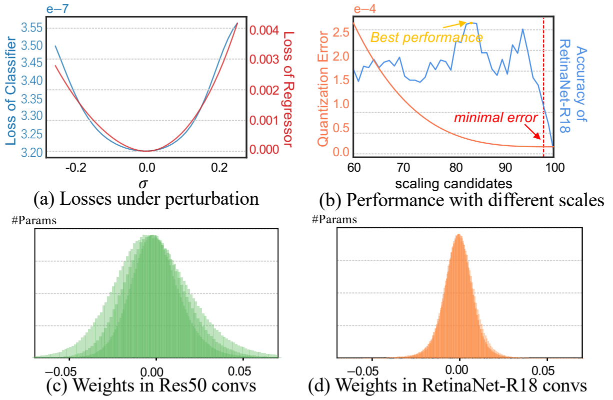
Reg-PTQ: Regression-specialized Post-training Quantization for Fully Quantized Object Detector

Supplementary Material

To further demonstrate the design and effectiveness of our Reg-PTQ framework, we first supplement the advantage of using toy experiments for observation. Then, we report more evaluation results in the supplementary material, including additional experimental results on more datasets and architectures, hardware implementation performance and more visualizations.



(a) Losses under perturbation (b) Performance with different scales

(c) Weights in Res50 convs (d) Weights in RetinaNet-R18 convs

Figure S1. *Observation#1-3* from real models.

G. Advantage of Toy Experiments

We choose toy models for observation mainly with three reasons.

First, toy models can reach the same conclusion with the real one. We had conducted experiments on real RetinaNet (with ResNet18) for regressor and ResNet50 for classifier to validate it, which have the similar parameter amounts. Specifically, for ***Observation#1***, we visualize the loss of RetinaNet and Res50 perturbed by the uniform noise in Fig. S1(a). classifier is more robust to quantization noise. This is because small perturbations on output probability may have little impact on the final results. But regressors' output is in the continuous space, and any perturbations will be reflected in the final loss. This difference exist both in toy and real models. For ***Observation#2***, it is well recognized that minimizing the local quantization error may not get the optimal scaling factors [22, 62]. We also verified it on RetinaNet-R18 in Fig. S1(b). For ***Observation#3*****,** due to the training techniques, such as weight decay, real classifiers are tend to have normal distributions as well []. But we have proven in Theoretical Analysis 3.2 that distance-based regression loss has stronger regularization to the weights, making the more non-uniform distributions of regression weights. Fig. S1(c-d) visualize the weights distributions of classifier and regressor head which showcase the impact of the loss regularization.

Second, toy models can better control the unrelated factors and helps to focus the analysis on the fundamental difference between classifier and regressor, *i.e.*, the objective of both tasks, because real models use sophisticated training techniques to ensure convergence, and introduces confounding variables.

Third, toy models enable precise calculation of intermediate results because of the significantly fewer parameters, such as precise Hessian matrix in Fig. 6(b). The 4-layer toy model, which has only <1k parameters in each layer, makes it practical to calculate the Hessian matrix precisely. Therefore, toy models can help probe and reveal the latent properties.

H. Derivation for Theoretical Analysis (Non-uniform Distribution Case)

In Theoretical Analysis 3.2, we take the uniform distribution to derive the posterior probability of W to simplify the derivation, but it also applicable if $P(W)$ is non-uniform, *e.g.*, normal distribution. We provide the derivation in the following to prove that if the weight obeys normal distribution as priori, the training procedure with L_p loss also results in a quasi normal distribution of weight.

Assume that the priori of weight distribution obeys the normal distribution $P(W) \sim N(0, \Sigma)$, which can be written as $P(W) = \frac{1}{\sqrt{2\pi\Sigma}}\exp\left(-\frac{1}{2}W^\top\Sigma^{-1}W\right)$. Then the posterior probability of W is

$$P(W|X,Y) = \frac{1}{\sqrt{2\pi\Sigma}\lambda_1}\exp\left(-\frac{||f(X)-Y||_p}{\lambda_2} - \frac{W^\top W}{2\Sigma}\right) \tag{S11}$$

$$\propto \exp\left(-\frac{(Y-(X^\top W+b))^\top\Sigma(Y-(X^\top W+b))+W^\top\lambda_2 W}{2\lambda_2\Sigma}\right) \tag{S12}$$

$$(\text{if } p=2) \tag{S13}$$

$$\propto \exp\left(-\frac{(W-\mu)^\top\left(\lambda_2^{-1}XX^\top+\Sigma^{-1}\right)(W-\mu)}{2}\right), \tag{S14}$$

where $\mu = \frac{1}{\lambda_2}\frac{1}{\lambda_2^{-1}XX^\top+\Sigma^{-1}}XY$. Simplify Eq. S14 then we have

$$P(W|X,Y) \propto N(\mu, \frac{1}{\lambda_2^{-1}XX^\top+\Sigma^{-1}}). \tag{S15}$$

Method	**#Bit$_{(W/A)}$**	**RetinaNet**		**YOLOF**	**Faster RCNN**	
		ResNet-50	**ResNet-101**	**ResNet-50**	**ResNet-50**	**ResNet-101**
Full-precision	32/32	37.4	38.9	37.5	38.5	39.8
baseline (FP Head)	2/4	27.0	29.0	25.7	28.1	30.0
Reg-PTQ (Ours)	**2/4**	**23.9**	**24.8**	**19.3**	**19.1**	**21.5**
baseline (FP Head)	3/3	30.0	30.8	28.2	31.7	31.1
Reg-PTQ (Ours)	**3/3**	**28.1**	**28.3**	**27.3**	**28.1**	**29.1**
baseline (FP Head)	4/4	35.2	36.5	34.4	37.7	38.0
Reg-PTQ (Ours)	**4/4**	**36.7**	**35.9**	**34.3**	**36.7**	**36.2**
baseline (FP Head)	4/8	37.4	38.9	37.1	38.3	39.5
Reg-PTQ (Ours)	**4/8**	**37.4**	**38.6**	**36.8**	**37.8**	**39.1**

Table S1. Comparison with other PTQ methods on various detectors with ResNet-50/101 as the backbone on COCO dataset.

It means that if the weight priori is the normal distribution symmetric to zero, which is a common conception of W, the prosterior probability is also likely to gather around the center. Therefore, it is straightforward to think that the L_p-like losses imposes a regularization on the weight to push them to the center.

I. More Experimental Results

To evaluate the performance of our Reg-PTQ framework when calibrating various object detection models, we perform extra experiments on more datasets and architectures, and compare with various existing PTQ methods.

I.1. Comparison to Only Quantizing Backbone

Accuracy of only quantizing the backbone and neck can be found in previous works [21, 38, 53] for corresponding methods. Therefore, due to the limited space, we do not put the results of only quantizing backbone in main text. We report the accuracy comparison of whether quantize head in Table S1. It shows that additionally quantizing head brings little accuracy drop.

I.2. Results on PASCAL VOC Dataset

Implementation detail. We further validate our method and compare it with other works on PASCAL VOC dataset [9], which is also a widely evaluated object detection dataset. We select RetinaNet [28] and Faster RCNN [44] with ResNet-50 backbone as representatives of one and two-stage detectors, respectively.

Results. As Table S2 shows, our Reg-PTQ has consistent performance improvement on PASCAL VOC dataset. It surpasses existing PTQ methods by wide margins, especially under lower bit-width. For example, when quantizing to W2A4 bit-width, Reg-PTQ outperforms existing SOTA methods by 5.4% with RetinaNet and 2% with Faster RCNN. The performance of the fully quantized detectors on W4A4 and W4A8 are comparable with the full-precision counterparts, which show the promising application potential of fully quantized detectors in the real-world.

Method	**#Bit$_{(W/A)}$**	**Faster RCNN**	**RetinaNet**
		ResNet-50	**ResNet-50**
Full-precision	32/32	80.4	77.3
AdaQuant [35]	2/4	0	0.6
BRECQ [35]	2/4	54.0	39.3
PD-Quant [30]	2/4	20.9	51.8
SubSetQ [40]	2/4	41.1	35.0
QDrop [56]	2/4	57.2	49.5
Reg-PTQ (Ours)	**2/4**	**59.2**	**57.2**
AdaQuant	3/3	29.4	57.8
BRECQ	3/3	59.5	66.8
PD-Quant	3/3	22.7	67.1
SubSetQ	3/3	66.1	63.2
QDrop	3/3	70.1	69.2
Reg-PTQ (Ours)	**3/3**	**72.3**	**70.9**
AdaRound	4/4	2.0	1.9
AdaQuant	4/4	50.5	74.8
BRECQ	4/4	73.5	75.2
PD-Quant	4/4	59.7	74.2
SubSetQ	4/4	76.8	74.6
QDrop	4/4	77.6	75.5
Reg-PTQ (Ours)	**4/4**	**78.3**	**76.0**
AdaQuant	4/8	54.2	76.5
BRECQ	4/8	57.5	77.1
PD-Quant	4/8	63.8	76.7
SubSetQ	4/8	79.2	77.1
QDrop	4/8	79.5	77.2
Reg-PTQ (Ours)	**4/8**	**79.6**	**77.2**

Table S2. Comparison with other PTQ methods on various detectors with ResNet-50 as the backbone on PASCAL VOC dataset.

Method	**#Bit$_{(W/A)}$**	**DETR**
		ResNet-50
Full-precision	32/32	39.9
QDrop [56]	2/4	12.2
Reg-PTQ (Ours)	**2/4**	**18.3**
QDrop [56]	3/3	19.1
Reg-PTQ (Ours)	**3/3**	**29.4**
QDrop	4/4	29.5
Reg-PTQ (Ours)	**4/4**	**37.4**

Table S3. Evaluation on transformer-based architectures on COCO dataset.

I.3. Results on Transformer-based Architectures

Implementation detail. Besides CNN-based detectors, we also conduct full quantization on transformer-based archi-

Method	#Bit$_{\text{(W/A)}}$	RetinaNet		YOLOF	Faster RCNN		Mask RCNN	
		ResNet-50	ResNet-101	ResNet-50	ResNet-50	ResNet-101	ResNet-50	ResNet-101
Full-precision	32/32	37.4	38.9	37.5	38.5	39.8	39.2	40.8
AdaQuant	2/4	0	0	0	0	0	0	0
AdaRound	2/4	2.6	6.8	1.4	14	15	14.3	16.3
SubSetQ	2/4	6.9	6.1	7.1	6.9	7.9	8.4	8.7
Reg-PTQ (Ours)	**2/4**	**23.9**	**24.8**	**19.3**	**19.1**	**21.5**	**19.1**	**20.7**
SubSetQ	3/3	23.6	24.7	20.7	23.2	18.2	24.1	25.3
Reg-PTQ (Ours)	**3/3**	**28.1**	**28.3**	**27.3**	**28.1**	**29.1**	**28.4**	**28.8**
AdaRound	4/8	20.1	21.2	14.1	22.1	23.8	22.9	24.3
Reg-PTQ (Ours)	**4/8**	**37.4**	**38.6**	**36.8**	**37.8**	**39.1**	**38.3**	**40.0**

Table S4. More results compared with existing PTQ methods on COCO dataset.

DataType	Latency$_{\text{(ms)}}$	Storage$_{\text{(MB)}}$
Float32	796.4	129.7
INT16	438.4	68.6
INT4*	132.8	38.6
INT4	84.5	22.8

Table S5. Efficiency and storage reduction on single NVIDIA Tesla T4 implemented with TVM. **DataType** denotes the weights and activation datatype. INT4* means we only quantize backbone and FPN neck to 4-bit but leave the heads full-precision. Other results without * means full quantization with uniform bitwidth.

tecture on COCO object detection dataset [25]. Representatively, we select DETR [5] with transformer-based encoder and decoder as heads and ResNet-50 as the backbone.

Results. Table S3 shows the results of our Reg-PTQ approach, which demonstrates that full quantization is also feasible on transformer-based architectures. The W4A4 quantized DETR using our Reg-PTQ achieves comparable performance to its full-precision counterpart, which has a slight 2.5% performance drop. It has a noticeable 7.9% advancement compared to the state-of-the-art baseline approach QDrop [56]. Meanwhile, the improvement under lower bit-width is more impressive, which is 10.3% under W3A3 and 6.1% under W2A4 compared with the baseline method QDrop. It shows that our method is capable of transformer-based detection architectures and accomplishes promising performance.

I.4. More Results on COCO Dataset

Due to the limited length, we leave out some comparison results on COCO detection dataset [25] in our main text, including AdaRound [35], AdaQuant [19] and SubSetQ [40], which crashes under ultra-low bit-width. We report the rest of the results in Table S4, and the results of our Reg-PTQ are bolded. As shown in the table, our Reg-PTQ outperforms other baseline approaches, which further demonstrates the effectiveness of our method.

J. Practical Speedup and Storage Saving

We also conduct hardware deployment to demonstrate the practical value of full quantization. We give a comparison of the acceleration and storage saving between fully quantized model, model with backbone and neck quantized only, and full-precision model.

Implementation detail. We implement detection models on 1 NVIDIA Tesla T4 GPU by TVM deployment framework [3]. We follow HAWQ [61] to implement INT4 operators, which realizes the bit-packing and data allocation layouts. We choose RetinaNet as an example for demonstration. The backbone is a standard ResNet-50 with four stages, and the neck is FPN [26]. The classification and regression heads have five layers each, and we leave the last layer unquantized. For speedup testing, we use one image with 512×512 pixels as the input and calculate the averaged inference time of 10 running. For storage compression testing, we pack eight INT4 tensors to INT32 according to the data allocation layouts of HAWQ. It should be noted that we implement uniform quantization to all layers to simplify the project. Previous works [15, 42, 49] have implemented the logarithmic-like quantizers on FPGAs. Fortunately, compared with them, there is no additional operations in Reg-PTQ. We will investigate the deployment of Reg-PTQ in future work.

Results. Table S5 shows the inference time and storage of RetinaNet with ResNet-50 under different bit-width settings. The **Latency** of processing a 512×512 image is 796.4ms using the full-precision model on the Tesla T4. And that for the fully quantized INT16 model is 438.4ms, which achieves nearly 2× acceleration. Moreover, the inference time of the INT4 fully quantized detector is 84.5ms, which achieves an impressive 9.4× speedup compared with its full-precision counterpart.

As for the **Storage**, the fully quantized INT4 model is 22.8MB, while the one only quantizing backbone and FPN neck is 38.6MB. In this case, we additionally achieve 15.8MB storage saving brought by quantizing detection heads, which can further compress the INT4* model by

[3] https://github.com/apache/tvm

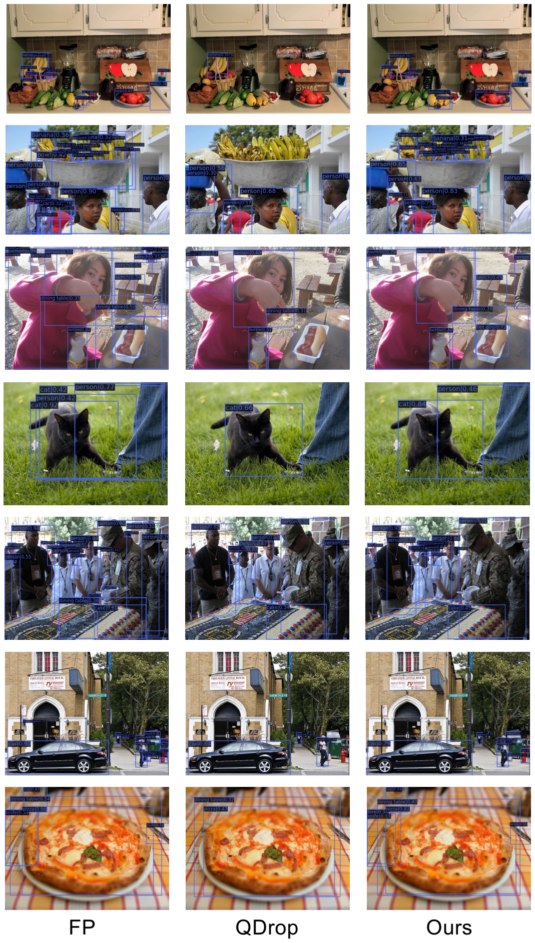


Figure S2. Visualization of detection results by full-precision (FP) detectors and 3-bit quantized models.

40.9%. Compared with the full precision model, we achieve 5.7× compression for INT4 fully quantized one. This acceleration and compression are roughly consistent with the theoretical ratios calculated in our main text. The small difference between the theoretical ratios and practical ones is caused by many factors, such as the hardware condition or operator optimization, *etc*.

In a word, the remarkable acceleration and storage reduction on hardware show the potential of fully quantized detectors on real-world edge devices. Compared with the model only quantizing backbone and head, fully quantized one can achieve more speedup and storage saving.

J.1. Visualization

We visualize the detection results of our Reg-PTQ on W3A3 in Figure S2 compared with other PTQ methods. Our Reg-PTQ predicts the bounding boxes more accurately, with fewer objects missed and higher classification confidence compared to QDrop, which indicates the great potential of applying fully quantized detectors in real-world scenarios.