AQD: Towards Accurate Quantized Object Detection

Jing Liu[1] Bohan Zhuang[3*] Peng Chen[2] Mingkui Tan[1*] Chunhua Shen[2]
[1]South China University of Technology [2]University of Adelaide [3]Monash University

Abstract

*Network quantization aims to lower the bitwidth of weights and activations and hence reduce the model size and accelerate the inference of deep networks. Even though existing quantization methods have achieved promising performance on image classification, applying aggressively low bitwidth quantization on object detection while preserving the performance is still a challenge. In this paper, we demonstrate that the poor performance of the quantized network on object detection comes from the inaccurate batch statistics of batch normalization. To solve this, we propose an accurate quantized object detection (AQD) method. Specifically, we propose to employ multi-level batch normalization (multi-level BN) to estimate the batch statistics of each detection head separately. We further propose a learned interval quantization method to improve how the quantizer itself is configured. To evaluate the performance of the proposed methods, we apply AQD to two one-stage detectors (*i.e.*, RetinaNet and FCOS). Experimental results on COCO show that our methods achieve near-lossless performance compared with the full-precision model by using extremely low bitwidth regimes such as 3-bit.* ***In particular, we even outperform the full-precision counterpart by a large margin with a 4-bit detector, which is of great practical value.*** *Code is available at* https://github.com/blueardour/model-quantization.

1. Introduction

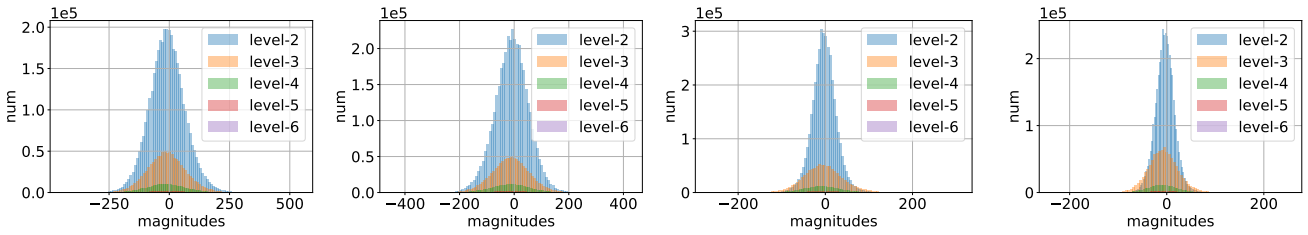
Since 2012, convolutional neural networks (CNNs) have achieved great success in image classification [9, 10, 17], object detection [20, 21, 29], face recognition [6, 23, 31], etc. However, existing networks always have a large number of parameters and high computational cost, which restricts their application on resources-limited devices such as smartphones, AR glasses, and drones. To solve this issue, network compression is an effective method, which aims to reduce the parameters and computational costs of the network without significant performance degradation.

*Corresponding authors.

Existing studies on network compression focus on channel pruning [11, 25, 42], efficient architecture design [12, 27, 30] and network quantization [36, 39, 40]. In particular, network quantization directly reduce the model size by converting the network weights into low-precision (e.g., 4bit or 2bit) ones. As a result, the low-precision model can achieve substantial memory saving (e.g., $8\times$ or $16\times$). More importantly, network quantization also converts the activations into low-precision ones. As a result, we can replace the compute-intensive floating-point operations with lightweighted fixed-point or bitwise operations, which greatly reduces the computational cost of the networks.

Although promising results on tasks such as image classification have been reported [7, 16, 39] using the aforementioned quantization techniques, using quantized networks for more complex tasks such as object detection still remains a challenge. Compared with image classification, object detection is more challenging. In object detection, the detector not only performs object classification, but also conducts bounding box regression, which makes it more difficult to quantize the detection model. Existing methods [15, 18, 37] quantize the detector to 4 or 8 bits and achieve promising performance. However, when it comes to lower bitwidth (*e.g.*, 2-bit) quantization, it incurs a significant performance drop. In this paper, we observe that the main challenge of quantization on object detection is the inaccurate batch statistics of batch normalization [14]. In a one-stage detector [21, 32], each detection head decodes scale-specific features. Different scales of features result in different means and variances. The discrepancy of means and variances across different levels of features leads to inaccurate statistics of shared batch normalization layers. This issue becomes even more severe when we perform low-precision quantization on the one-stage detector.

To address this issue, in this paper, we propose an accurate quantized object detection (AQD) method. Specifically, we construct multi-level batch normalization (multi-level BN) to obtain accurate batch statistics. We use independent batch normalization for each pyramid level of head, which can capture accurate batch statistics. Moreover, we further propose a learned interval quantization method (LIQ) to improve the performance of the quantized network.



(a) 1-th layer of classification head (b) 1-th layer of regression head (c) 2-th layer of classification head (d) 2-th layer of regression head

Figure 1. Histograms of batch statistics at the batch normalization layer of a 2-bit ResNet-18 FCOS detector. Level-x denotes that the prediction are made on the x-th pyramid level. Different pyramid levels of feature show different batch statistics.

Our main contributions are summarized as follows.

- We highlight that poor performance of the quantized detectors is due to the inaccurate statistics of batch normalization. We therefore propose multi-level batch normalization (multi-level BN) to capture accurate batch statistics of different scales of feature pyramid. We further propose a learned interval quantization method (LIQ) by improving how the quantizer itself is configured.
- We evaluate the proposed methods on COCO detection benchmark with multiple precisions. Experimental results show that our 3-bit AQD achieves comparable performance with its full-precision counterpart. To be emphasized, our 4-bit AQD even outperforms the 32-bit model by a large margin.

2. Related Work

Network quantization. Network quantization aims to represent the network weighs and/or activations with very low precision, which reduce the model size and computational cost. Existing methods can be divided into two categories, namely, binary quantization [3, 13, 28] and fixed-point quantization [35, 36, 39]. Binary quantization convert the full precision weights and activations to $\{+1, -1\}$. In this way, we can replace the matrix multiplication operations with the bitwise XNOR-popcount operations. As a result, the binary convolution layer can achieve up to $32\times$ memory saving and $58\times$ speedup on CPUs [28, 40]. Nevertheless, binary quantization incurs significant performance degradation compared with the full precision counterparts. To reduce the performance gap, fixed-point quantization [4, 35, 36, 39] have been proposed to represent weights and activations with higher bitwidth, which achieve impressive performance on image classification task.

Quantization on Object Detection. Many researchers have studied quantization on object detection to speed up on-device inference and save storage. Jacob *et al*. [15] propose a quantization scheme using integer-only arithmetic and perform object detection on COCO dataset with quantized 8-bit models. Wei *et al*. [33] utilize knowledge distillation and quantization to train very tiny CNNs for object detection. Observing the instability problem during the fine-tuning stage of the quantization process, Li *et al*. [18] propose to produce fully quantized 4-bit detectors based on RetinaNet and Faster R-CNN with three techniques, which include freezing batch normalization statistics, clamping activation based on percentile and quantizing with channel-wise scheme. Zhuang *et al*. [37] point out the difficulty of propagating gradient and propose to train low-precision network with a fully precision auxiliary module.

3. Proposed method

In this section, we describe the proposed accurate quantized object detection (AQD). We first introduce problem definition in Section 3.1. Then we introduce the inaccurate batch statistics issue and the multi-level batch normalization (multi-level BN) in Section 3.2. Then, we introduce the learned interval quantization (LIQ) in Section 3.3.

3.1. Problem definition

We consider build a one-stage quantized detector for object detection. One-stage object detector consists of a backbone, a feature pyramid and prediction heads. For a convolutional layer in a detector, we define the input $\mathbf{X}$ and weight parameter $\mathbf{W}$. Let x_m and w_m be the m-th element of $\mathbf{X}$ and $\mathbf{W}$, respectively. Here, we omit the subscript m for convenience. Quantization seeks to reduce the bitwidth of w and x via quantizers:

$$\begin{aligned} Q_{\mathbf{W}}: \ w \xrightarrow{T_{\mathbf{W}}} \hat{w} \xrightarrow{D_{\mathbf{W}}} \bar{w} \\ Q_{\mathbf{X}}: \ x \xrightarrow{T_{\mathbf{X}}} \hat{x} \xrightarrow{D_{\mathbf{X}}} \bar{x}, \end{aligned} \tag{1}$$

where a quantizer $Q_\Delta(\Delta \in \{\mathbf{W}, \mathbf{X}\})$ contains a transformer $T_\Delta(\Delta \in \{\mathbf{W}, \mathbf{X}\})$ and a discretizer $D_\Delta(\Delta \in \{\mathbf{W}, \mathbf{X}\})$. Following [7], the transformer $T_\Delta(\Delta \in \{\mathbf{W}, \mathbf{X}\})$ transforms the value v to $[-Q_N, Q_P]$ with learnable step size s. Let k be the bitwidth of the weights and activations of the quantized networks. For weights, Q_N and Q_P are 2^{k-1} and $2^{k-1}-1$. For activations, Q_N and Q_P

are 0 and $2^k - 1$. The discretizer $D_\Delta(\Delta \in \{\mathbf{W}, \mathbf{X}\})$ maps the continuous value $\hat{v}$ in the range $[-Q_N, Q_P]$ to some discrete value $\bar{v}$.

3.2. Multi-level batch normalization

During the training of the quantized detector, batch normalization [14] normalize the input features and update exponential moving average (EMA) statistics μ_{EMA} and σ_{EMA} with current batch statistics μ and σ. In one-stage object detection frameworks, each prediction head encodes a corresponding feature level. However, due to the quantization process, there may be large divergence of batch statistics between different feature levels, as shown in Figure 1. Therefore, using a shared batch normalization across prediction heads may lead to inaccurate batch statistics, which will cause a significant performance drop.

To solve this issue, we propose a simple yet effective method, called multi-level batch normalization (multi-level BN), that uses independent batch normalization for different feature levels. The multi-level BN can capture individual batch statistics of the corresponding feature level. There are two advantages of our method comparing to the standard shared BN strategy. First, multi-level BN only introduces negligible parameters. In fact, multi-level BN only has less than 1.1% of the model size. Second, multi-level BN does not change the architecture of the network. Therefore, the proposed multi-level BN does not increase any additional computational cost. An empirical study on the effect of multi-level BN can be found in Section 4.5.

3.3. Learned interval quantization

During the training of the quantized network, the gradient of step size $\partial\mathcal{L}/\partial s$ is defined as follow:

$$\frac{\partial \mathcal{L}}{\partial s} = \sum_i \frac{\partial \mathcal{L}}{\partial \bar{v}_i} \frac{\partial \bar{v}_i}{\partial s}, \tag{2}$$

where i is over all elements in the corresponding layer. The summation over all elements makes the gradient of step size $\partial\mathcal{L}/\partial s$ much larger than the gradient of the input value $\partial\mathcal{L}/\partial v$. Hence, directly training the quantized network with learnable step size is unstable. One possible solution is to rescale the gradient [7]. However, it is hard to determine the value of scaling factor. The improper gradient scaling factor may hamper the performance of the quantized network.

To solve this issue, we propose a learned interval quantization (LIQ) method to quantize the network with trainable interval instead of step size, which can avoid rescaling the gradient. The transformer and discretizer for the proposed LIQ can be defined as follow:

Transformer: The transformer $T_\Delta(\Delta \in \{\mathbf{W}, \mathbf{X}\})$ transforms the weights and activations to $[0, 1]$, which are defined as follows:

$$\hat{w} = T_{\mathbf{W}}(w) = \frac{1}{2} \cdot (\max(-1, \min(\frac{w}{\alpha_{\mathbf{W}}}, 1)) + 1) \tag{3}$$

$$\hat{x} = T_{\mathbf{X}}(x) = \max(0, \min(\frac{x}{\alpha_{\mathbf{X}}}, 1)), \tag{4}$$

where $\alpha_{\mathbf{W}}$ and $\alpha_{\mathbf{X}}$ are trainable interval parameters that limits the range of weights and activations.

Discretizer: The discretizer $D_\Delta(\Delta \in \{\mathbf{W}, \mathbf{X}\})$ maps the continuous value $\hat{v}$ in the range $[0, 1]$ to some discrete value $\bar{v}$, which is defined as follow:

$$\bar{v} = D_\Delta(\hat{v}) = \text{round}(\hat{v} \cdot q) \cdot \frac{1}{q}, \tag{5}$$

where q is the number of discrete values (except 0). Let k be the bitwidth of the weights and activations of the quantized networks. Then, q can be computed as $q = 2^k - 1$. After weights and activations quantization, we use affine transformation to bring the range of weights to $[-\alpha_{\mathbf{W}}, \alpha_{\mathbf{W}}]$ and range of activations to $[0, \alpha_{\mathbf{X}}]$.

Back-propagation with gradient approximation: In general, the discretizer function is non-differentiable. Therefore, it is impossible to train the quantized network through back-propagation. To solve this issue, we use the straight-through estimator [1, 5, 36] (STE) to approximate the gradients. Then, we can use following equations to compute the gradient of $\partial\bar{w}/\partial\alpha_{\mathbf{W}}$ and $\partial\bar{x}/\partial\alpha_{\mathbf{X}}$ respectively:

$$\frac{\partial \bar{w}}{\partial \alpha_{\mathbf{W}}} = \begin{cases} \text{round}(\frac{w}{\alpha_{\mathbf{W}}}) - \frac{w}{\alpha_{\mathbf{W}}} & |w| < \alpha_\Delta \\ 1 & |w| \geq \alpha_\Delta \end{cases}, \tag{6}$$

$$\frac{\partial \bar{x}}{\partial \alpha_{\mathbf{X}}} = \begin{cases} \text{round}(\frac{x}{\alpha_{\mathbf{X}}}) - \frac{x}{\alpha_{\mathbf{X}}} & x < \alpha_\Delta \\ 1 & x \geq \alpha_\Delta \end{cases}. \tag{7}$$

4. Experiments

4.1. Compared methods

To investigate the effectiveness of the proposed method, we consider several state-of-the-art quantization methods for comparison. On FCOS, we compare the proposed methods with Group-Net [40]. On RetinaNet, we compare the proposed methods with FQN [18] and Auxi [38].

4.2. Data sets

We evaluate our proposed methods on the COCO detection benchmark [22]. COCO detection benchmark is a large-scale benchmark data set for object detection, which is widely used to evaluate the performance of the detector. Following [20, 41], we use the COCO *trainval35k* split (115K images) for training and *minival* split (5K images) for validation.

Table 1. Results of FCOS on the COCO validation set.

Backbone	Model	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
ResNet-18	Full precision [32]	33.9	51.2	36.4	19.3	36.2	44.0
	Group-Net [41] (4 bases)	28.9	45.3	31.2	15.4	30.5	38.1
	AQD (4-bit)	35.2	52.7	37.8	20.3	37.2	46.1
	AQD (3-bit)	34.1	51.4	36.7	19.1	35.8	45.2
	AQD (2-bit)	30.6	47.3	32.4	16.6	31.9	41.3
ResNet-34	Full precision [32]	38.0	55.9	41.0	23.0	40.3	49.4
	Group-Net [41] (4 bases)	31.5	47.6	33.8	16.9	32.3	40.1
	AQD (4-bit)	38.6	56.9	41.5	22.5	41.2	51.0
	AQD (3-bit)	37.4	55.5	40.3	21.2	39.7	48.8
	AQD (2-bit)	34.5	52.4	37.0	19.0	36.6	46.0

Table 2. Results of RetinaNet on the COCO validation set.

Backbone	Model	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
ResNet-18	Full precision [32]	32.3	50.9	34.2	18.9	35.6	42.5
	FQN [18] (4-bit)	28.6	46.9	29.9	14.9	31.2	38.7
	Auxi [38] (4-bit)	31.9	50.4	33.7	16.5	34.6	42.3
	AQD (4-bit)	34.0	53.1	36.3	18.8	37.2	45.3
	AQD (3-bit)	32.8	51.7	34.9	18.1	35.1	44.6
	AQD (2-bit)	29.6	48.1	15.9	31.7	15.9	41.1
ResNet-34	Full precision [32]	36.3	56.2	39.1	22.4	39.8	46.9
	FQN [18] (4-bit)	31.3	50.4	33.3	16.1	34.4	41.6
	Auxi [38] (4-bit)	34.7	53.7	36.9	19.3	38.0	45.9
	AQD (4-bit)	37.0	57.0	39.8	21.6	40.1	49.1
	AQD (3-bit)	35.9	56.0	38.5	20.9	39.0	47.9
	AQD (2-bit)	33.1	52.5	35.4	18.6	36.1	45.2

4.3. Implementation details

We implement the proposed method based on detectron2 [34]. We apply our AQD on two one-stage detectors, namely, FCOS [32] and RetinaNet [21]. FCOS and RetinaNet contain three parts, including a backbone, a feature pyramid, and detection heads. We use ResNet-18 and ResNet-34 [9] as backbones. Following [37, 41], we quantize all the layers in the network except the first layer in the backbone and the last layer in the detection heads. To stabilize the optimization, each convolution layer is followed by BN and ReLU. We do not fix the BN during training. We replace the BN with synchronized batch normalization (Sync BN) to fully exploit data across all devices. Following APOT [19], we use weight normalization before weight quantization to provide relatively consistent and stable input distribution.

Following [18, 37], all images in the training and validation set are resized so that their shorter edges are 800 pixels. During training, images are augmented by random horizontal flipping. During evaluation, we do not perform any augmentations. We train the network for 90K iterations with a mini-batch size of 16. We use SGD with momentum for optimization. The learning is started at 0.01, and divided by 10 at iterations 60K and 80K. We set the weight decay to 0.0001. More details on the other hyper-parameters settings can be found at [21, 32].

4.4. Comparisons on COCO

We compare the proposed methods with several state-of-the-art quantized models and report the results in Table 1 and Table 2. From the results, we have the following observations. First, our AQD outperforms the considered baselines on different detection frameworks and backbones. For example, our 4-bit RetinaNet detector with ResNet-18 backbone outperforms FQN [18] and Auxi [38] by a large margin. Second, our 4-bit quantized detectors even outperform the corresponding full-precision models. Specifically, on a 4-bit RetinaNet detector, our AQD surpasses the full-precision model by 1.7% on the ResNet-18 backbone. Third, when performing 3-bit quantization, our AQD achieves near lossless performance compared with the full-precision counterpart. To be specific, on a 3-bit RetinaNet detector with ResNet-34 backbone, our AQD only leads to 0.4% performance degradation on the AP. Forth, when conducting aggressive 2-bit quantization, our AQD still achieves comparable performance. For example, our

Table 3. Effect of the multi-level batch normalization. We quantize the FCOS detector to 2-bit and evaluate the performance on COCO.

Backbone	Model	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
ResNet-18	AQD w/ Shared Sync BN	29.5	46.6	31.7	19.0	32.8	35.8
	AQD w/ Multi-level Sync BN	30.6	47.3	32.4	16.6	31.9	41.3

Table 4. Effect of quantization on different components. We quantize the FCOS detector to 2-bit and evaluate the performance on COCO.

Backbone	Model	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
ResNet-18	Full precision [32]	33.9	51.2	36.4	19.3	36.2	44.0
	Backbone	33.2	50.1	35.5	19.1	34.7	44.5
	Backbone + Feature Pyramid	32.3	49.0	34.9	17.4	33.7	43.7
	Backbone + Feature Pyramid + Heads	30.6	47.3	32.4	16.6	31.9	41.3

Table 5. Comparisons of different methods on ImageNet. "-" denotes that the results are not reported.

Network	Model	Top-1 Acc. (%)	Top-5 Acc. (%)
ResNet-18	DoReFa [36]	62.6	84.4
	PACT [4]	64.4	85.6
	DSQ [8]	65.2	-
	LQ-Net [35]	64.9	85.9
	QIL [16]	65.7	-
	LSQ+ [2]	66.8	-
	LIQ	**67.4**	**87.5**

2-bit FCOS detector with ResNet-18 backbone only suffer 3.3% AP loss compared with its full-precision baseline. These results justify the superior performance of our proposed AQD.

4.5. Effect of multi-level batch normalization

To study the effect of multi-level BN, we quantize the FCOS detector with multi-level BN and shared BN. Here, the detector with shared BN indicates that the batch normalization in the detection heads are shared across different pyramid levels. The results are shown in Table 3. From the results, the detector with multi-level sync BN outperforms the one with shared sync BN by 1.1% on AP, which demonstrates the effectiveness of the proposed multi-level BN.

4.6. Effect of learned interval quantization

To investigate the effect of LIQ, we quantize ResNet-18 to 2-bit with different quantization methods and evaluate the model on ImageNet [17]. Following the settings in LSQ [7], we train the quantized network for 90 epochs using a mini-batch size of 256. We use SGD with nesterov [26] for optimization. The momentum is set to 0.9. The learning rate is initialized to 0.01, and decrease to 0 following the cosine function [24]. We report the results in Table 5. From the results, we have following observations. First, LIQ outperforms LSQ+ by 0.6% in the Top-1 accuracy, which demonstrates the effectiveness of the learned interval quantization. Second, LIQ outperforms all the consider baselines, which shows the superior performance of our proposed LIQ.

4.7. Effect of quantization on different components

We study the effect of quantizing different components in object detection models. The results are shown in Table 4. From the results, we have the following observations. Quantizing the backbone only leads to a small performance drop (0.7% in AP). Nevertheless, quantizing the feature pyramid and detection heads will cause significant performance degradation (3.3% in AP). These results show that the detector is sensitive to the quantization of feature pyramid and detection heads, which provides a direction to improve the performance of the quantized network.

5. Conclusions

In this paper, we have proposed an accurate quantized object detection (AQD) framework. We have first proposed multi-level batch normalization to capture batch statistics for different detection heads. Then, we have proposed a learned interval quantization (LIQ) strategy to further improve the performance of the quantized network. To evaluate the performance of the proposed methods, we have applied our AQD to two classical one-stage detectors. Experimental results have justified that our quantized 3-bit detector achieves near-lossless performance compared with the full-precision counterpart. More importantly, our 4-bit detector even outperforms the full-precision model by a large margin.

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. 3

[2] Yash Bhalgat, Jinwon Lee, Markus Nagel, Tijmen Blankevoort, and Nojun Kwak. Lsq+: Improving low-bit quantization through learnable offsets and better initialization. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2020. 5

[3] Adrian Bulat and Yorgos Tzimiropoulos. Hierarchical binary cnns for landmark localization with limited resources. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2018. 2

[4] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018. 2, 5

[5] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830*, 2016. 3

[6] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 4690–4699, 2019. 1

[7] Steven K. Esser, Jeffrey L. McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S. Modha. Learned step size quantization. In *Proc. Int. Conf. Learn. Repren.*, 2020. 1, 2, 3, 5

[8] Ruihao Gong, Xianglong Liu, Shenghu Jiang, Tianxiang Li, Peng Hu, Jiazhen Lin, Fengwei Yu, and Junjie Yan. Differentiable soft quantization: Bridging full-precision and low-bit neural networks. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 4852–4861, 2019. 5

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 770–778, 2016. 1, 4

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Proc. Eur. Conf. Comp. Vis.*, pages 630–645, 2016. 1

[11] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 1389–1397, 2017. 1

[12] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 1

[13] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In *Proc. Adv. Neural Inf. Process. Syst.*, pages 4107–4115, 2016. 2

[14] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. Int. Conf. Mach. Learn.*, pages 448–456, 2015. 1, 3

[15] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 2704–2713, 2018. 1, 2

[16] Sangil Jung, Changyong Son, Seohyung Lee, Jinwoo Son, Jae-Joon Han, Youngjun Kwak, Sung Ju Hwang, and Changkyu Choi. Learning to quantize deep networks by optimizing quantization intervals with task loss. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2019. 1, 5

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. Adv. Neural Inf. Process. Syst.*, pages 1097–1105, 2012. 1, 5

[18] Rundong Li, Yan Wang, Feng Liang, Hongwei Qin, Junjie Yan, and Rui Fan. Fully quantized network for object detection. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2019. 1, 2, 3, 4

[19] Yuhang Li, Xin Dong, and Wei Wang. Additive powers-of-two quantization: An efficient non-uniform discretization for neural networks. In *Proc. Int. Conf. Learn. Repren.*, 2020. 4

[20] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 2117–2125, 2017. 1, 3

[21] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 2980–2988, 2017. 1, 4

[22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proc. Eur. Conf. Comp. Vis.*, pages 740–755. Springer, 2014. 3

[23] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 212–220, 2017. 1

[24] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with warm restarts. In *Proc. Int. Conf. Learn. Repren.*, 2017. 5

[25] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proc. IEEE Int. Conf. Comp. Vis.*, pages 5058–5066, 2017. 1

[26] Yurii E Nesterov. A method for solving the convex programming problem with convergence rate o (1/kˆ 2). In *Proceedings of the USSR Academy of Sciences*, volume 269, pages 543–547, 1983. 5

[27] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameter sharing. In *Proc. Int. Conf. Mach. Learn.*, pages 4092–4101, 2018. 1

[28] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *Proc. Eur. Conf. Comp. Vis.*, pages 525–542, 2016. 2

[29] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Proc. Adv. Neural Inf. Process. Syst.*, pages 91–99, 2015. 1

[30] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 4510–4520, 2018. 1

[31] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 815–823, 2015. 1

[32] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: Fully convolutional one-stage object detection. In *Proc. IEEE Int. Conf. Comp. Vis.*, 2019. 1, 4, 5

[33] Yi Wei, Xinyu Pan, Hongwei Qin, Wanli Ouyang, and Junjie Yan. Quantization mimic: Towards very tiny cnn for object detection. In *Proc. Eur. Conf. Comp. Vis.*, September 2018. 2

[34] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 4

[35] Dongqing Zhang, Jiaolong Yang, Dongqiangzi Ye, and Gang Hua. Lq-nets: Learned quantization for highly accurate and compact deep neural networks. In *Proc. Eur. Conf. Comp. Vis.*, 2018. 2, 5

[36] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016. 1, 2, 3, 5

[37] Bohan Zhuang, Lingqiao Liu, Mingkui Tan, Chunhua Shen, and Ian Reid. Training quantized network with auxiliary gradient module. *arXiv preprint arXiv:1903.11236*, 2019. 1, 2, 4

[38] Bohan Zhuang, Lingqiao Liu, Mingkui Tan, Chunhua Shen, and Ian Reid. Training quantized neural networks with a full-precision auxiliary module. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2020. 3, 4

[39] Bohan Zhuang, Chunhua Shen, Mingkui Tan, Lingqiao Liu, and Ian Reid. Towards effective low-bitwidth convolutional neural networks. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 7920–7928, 2018. 1, 2

[40] Bohan Zhuang, Chunhua Shen, Mingkui Tan, Lingqiao Liu, and Ian Reid. Structured binary neural networks for accurate image classification and semantic segmentation. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, June 2019. 1, 2, 3

[41] Bohan Zhuang, Chunhua Shen, Mingkui Tan, Lingqiao Liu, and Ian Reid. Structured binary neural networks for image recognition. *arXiv preprint arXiv:1909.09934*, 2019. 3, 4

[42] Zhuangwei Zhuang, Mingkui Tan, Bohan Zhuang, Jing Liu, Yong Guo, Qingyao Wu, Junzhou Huang, and Jinhui Zhu. Discrimination-aware channel pruning for deep neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Proc. Adv. Neural Inf. Process. Syst.*, pages 881–892. Curran Associates, Inc., 2018. 1