SPQAT: A SPARSE QUANTIZATION-AWARE TRAINING METHOD

Anonymous authors
Paper under double-blind review

ABSTRACT

Quantization-aware training (QAT) has been demonstrated to not only reduce computational cost and storage footprint, but well retain the performance of full-precision neural networks. However, the tedious retraining requirement greatly weakens the practical value of QAT methods. In this paper, we attempt to reduce the training costs of QAT methods, which to our best knowledge are barely investigated in the literature. Our motive stands upon a straightforward-yet-valuable observation: A large portion of quantized weights, referred to as the partly scratch-off lottery ticket, reach the optimal quantization level after a few training epochs. This naturally inspires us to reduce calculations by freezing these weights in the remaining training period. Accordingly, we develop an efficient sparse QAT method, dubbed SpQAT. The method freezes a weight once the distance between the full-precision one and its quantization level is smaller than a controllable threshold. Along these lines, we show that the proposed SpQAT accurately identifies the partly scratch-off lottery ticket and results in a sparse weight gradient where many weights are pulled out of the training and their related computations are avoided. Extensive experiments demonstrate the efficacy of our SpQAT with 20%-60% weight gradient sparsity. With the elimination of related gradient calculation in the backward propagation, the performance of our SpQAT is still on par with or even better than the compared baseline.

1 INTRODUCTION

Deep neural networks (DNNs) have been the foundation of many computer vision tasks for their superiority in performance. However, the success usually relies on increasing model size and computational costs, which barricades deployments of DNNs on popular resource-limited devices, such as mobile phones. In recent years, a variety of model compression techniques have been excavated to overcome this issue (Han et al., 2015; Krishnamoorthi, 2018; Hinton et al., 2015; Lin et al., 2020a). Among these techniques, network quantization represents the full-precision weights and activations within DNNs in a low-bit format, making it one of the most promising techniques for simultaneously reducing both the model size and computational cost (Banner et al., 2019; Lin et al., 2020b; Zhong et al., 2022; Bulat et al., 2020; Gong et al., 2019).

In spite of the merits, quantized networks easily suffer from severe performance drops if simply performing the low-bit quantization, since the low-bit data format possesses a very limited representation capacity compared with the full-precision counterpart. Thus, the quantized network usually needs to be trained for dozens of epochs to compensate for the performance drop, which is known as quantization-aware training (QAT) (Jacob et al., 2018). Specifically, in the forward pass, weights and activations are quantized to simulate quantized inference. In the back propagation, the weights are updated as usual. All weights of the network are still represented in the full-precision structure so that they can be updated by small gradients. In QAT, the network weights are adjusted to accommodate the quantization effects, which therefore alleviates the performance drop significantly.

Nonetheless, narrowing the performance in a training-aware fashion entails a heavier training overhead. Generally, QAT trains quantized networks at the cost of similar training epochs to training the full-precision network (Esser et al., 2020; Lee et al., 2021b). However, the extra quantization operation gives rise to more computational costs in each training epoch. For example, it takes 2.5 hours to train the quantized ResNet-50 on a single NVIDIA 3090 for each epoch with a batch size

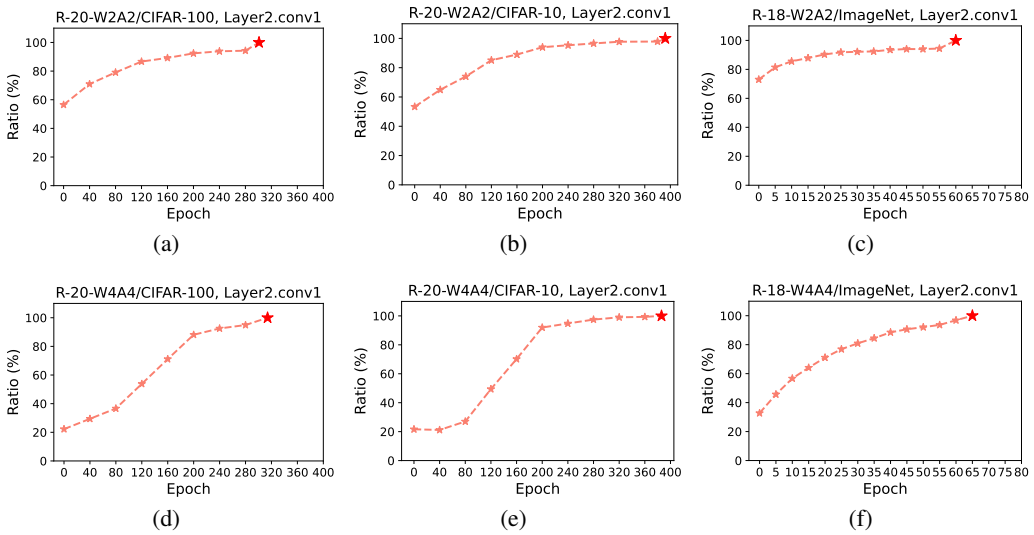
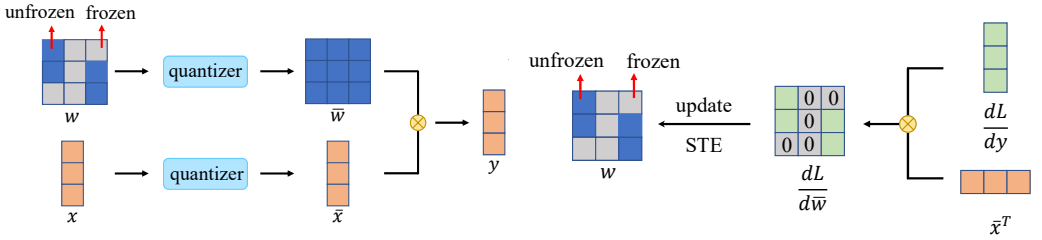


Figure 1: The ratio of the weights reaching the optimal quantization level *w.r.t.* training epochs. The red pentagon denotes the best model. "WBAB" indicates the weights and activations are quantized to B-bit. "R-20" and "R-18" indicate ResNet-20 and ResNet-18, respectively. Similar observations can be found in other networks on other bit-widths as well (See Appendix A.1).

of 128, while only 1.5 hours are required to complete training the full-precision ResNet-50. The intensive overhead calls for more computational costs and energy consumption, which undoubtedly barricades the application of quantized networks. Thus, an efficient training method for QAT becomes an imperative task in the research community.

In this paper, we investigate the problem of efficient QAT. At first, we identify an interesting observation during training a quantized network that a large portion of network weights quickly reach their optimal quantization levels after a few training epochs. As shown in Fig. 1(a), for "layer2.conv1" of the 2-bit ResNet-20 (He et al., 2016) trained on CIFAR-100 (Krizhevsky, 2009), near 60% weights have already reached their optimal quantization levels even without any training. At epoch 80, the portion rises up to around 80%. Similar observations can be found from Fig. 1(b)-Fig. 1(f) across different bit widths and datasets. We attribute this phenomenon to two characteristics in network quantization. First, a network is often quantized from a well-trained full-precision model. Second, a discrete quantization level often covers a range of continuous numerical values. As a result, most weights rapidly converge to their best quantization levels although their full-precision version remains far away from the optimal states. Such observations indicate that a portion of weights can be safely frozen and their gradients do not require calculating throughout the whole training process. As analyzed by (Zhu et al., 2020), backward propagation take up to two-thirds of the total training time. Therefore, the training costs can be well reduced in a weight gradient sparsity manner where the optimized weights are pulled out of the training process by zeroing out their gradients. Similar to the lottery ticket hypothesis in network pruning (Frankle & Carbin, 2019), We term these weights that quickly reach the optimal quantization levels as the partly scratch-off lottery ticket in this paper.

We utilize a simple-yet-effective heuristic rule to find the partly scratch-off lottery ticket. Simply speaking, we first measure the distance between a normalized weight and its corresponding quantization level. Then, the partly scratch-off lottery ticket comprises these weights whose exponential moving average (EMA) distance is lower than a pre-defined threshold. Accordingly, an efficient sparse QAT method, dubbed SpQAT, is introduced to reduce the training costs of quantized networks. An illustration of our SpQAT is presented in Fig. 2. In the forward pass, the full-precision weights and activations are first quantized with the quantizer, the results of which are multiplied to derive the convolutional outputs. In the backward pass, we freeze these weights within the partly scratch-off lottery ticket by setting their gradients to zero, leading to a sparse weight gradient. As a result, the related calculation of these gradients can be eliminated. By precisely identifying the partly scratch-off lottery ticket, the proposed SpQAT not only significantly reduces the training



(a) Freezing weights in the forward pass (b) Sparsifying weight gradients in the backward pass

Figure 2: An illustration of SpQAT.

costs, but also achieves comparable or even better performance. Typically, SpQAT obtains an average of 20%-60% gradient sparsity, which eliminates a total of 10%-30% overall gradient calculation. Moreover, SpQAT achieves comparable performance in the high-bit quantization (*e.g.*, 6, 8-bit), and even greatly improves the performance when quantizing the network into low-bit (*e.g.*, 2, 3, 4-bit).

2 Related Work

2.1 Quantization-Aware Training

Network quantization has long been the research focus of model compression community for its superiority in simultaneously reducing the model size and computational costs. However, the quantized network often suffers from performance degradation. A variety of techniques are explored to overcome this issue. Most existing QAT methods focus on designing quantizers that are more suitable for the network weights or activations (Esser et al., 2020; Li et al., 2020; Jung et al., 2019; Liu et al., 2022; Zhang et al., 2021). Other methods instead pay attention to special training strategies for quantized networks (Zhou et al., 2017; Zhuang et al., 2020; Lee et al., 2021b), approximating gradients of the quantization functions (Gong et al., 2019; Kim et al., 2021), training regularization (Lee et al., 2021a; Han et al., 2021) or mix-precision quantization (Dong et al., 2019; Yang & Jin, 2021; Chen et al., 2021b). Although these techniques alleviate the performance degradation, they stiffly rely on the training overhead in quantizing the network.

2.2 Training Acceleration

Many efforts have been made to accelerate the training DNNs. One common way is to exploit the highly-efficient distributed training (Goyal et al., 2017; Jia et al., 2018; You et al., 2018; Akiba et al., 2017). However, the distributed training is towards modern single- and multi-GPU workstations, servers, clusters, and even supercomputers. Therefore, the acceleration comes at expensive training costs and energy consumption. In contrast, our SpQAT aims to reduce training costs.

Gradient pruning targets pruning gradients in the computationally-intensive backpropagation (Sun et al., 2017; Goli & Aamodt, 2020). For example, meProp (Sun et al., 2017) only preserves the top-k gradients in the matrix multiplication output. In contrast to gradient pruning that continuously updates weights in the training process, our SpQAT stops updating these frozen weights forever.

Sparse training imposes sparsity constraints during network training (Mostafa & Wang, 2019; Evci et al., 2020; Liu et al., 2021; Raihan & Aamodt, 2020). By removing network weights in the training, related computations in forward and backward propagations can be reduced. Also, these methods are often designed for full-precision networks. Differently, our SpQAT does not remove any weight and is particularly designed for network quantization.

Another line of work proposed to represent gradients in a low-bit data format such that the costs can be decreased for the hardware-friendly gradient computing (Banner et al., 2018; Zhu et al., 2020). In compliance with extensive analyses on gradient distribution, most current methods focus on designing a suitable quantizer (Zhao et al., 2021). For instance, Lee *et al.* (Lee et al., 2022) implemented low-bit representation of gradients in a searching based style. (Chmiel et al., 2021) reasoned a near-lognormal distribution of gradients and obtained the best clipping value analytically.

Our SpQAT is orthogonal to these methods since we focus on quantizing network in forward pass. These methods can be integrated in our SpQAT to further speed up the network training.

3 METHOD

3.1 PRELIMINARY

A quantizer is adopted in QAT to quantize the full-precision weights and activations to simulate the quantization inference. The quantizer employed in this paper is borrowed from EWGS (Lee et al., 2021b). Concretely, given a full-precision data $\boldsymbol{x}$ (weights or activations), it is first normalized by a scaling coefficient s:

$$\boldsymbol{x_n} = \frac{\boldsymbol{x}}{s}. \tag{1}$$

Then, the normalized data $\boldsymbol{x_n}$ is quantized by the following quantizer:

$$\bar{\boldsymbol{x}} = \frac{1}{\alpha}\lfloor clip(\alpha \boldsymbol{x_n}, l, u)\rceil, \tag{2}$$

where $clip(\boldsymbol{x_n}, l, u) = min\big(max(\boldsymbol{x_n}, l), u\big)$, $\lfloor\cdot\rceil$ rounds its input to the nearest integer, and the result, $\bar{\boldsymbol{x}}$, is the quantization level of $\boldsymbol{x_n}$. The α, l, and u are three constants related to the bit-width B. Explicitly, for the weight quantizer, $\alpha = 2^{B-1}$, $l = -2^{B-1} - 1$, and $u = 2^{B-1} - 1$. As for the activation quantizer, $\alpha = 2^B$, $l = 0$, and $u = 2^B - 1$.

In contrast to EWGS (Lee et al., 2021b), we prevent quantizer output from re-scaling by the coefficient s given that the batch normalization diminishes the scaling effect (Hoffer et al., 2018). Following (Lee et al., 2021b), the scaling coefficient s is initialized to three times of the standard deviation of the full-precision data. For weights, we fix s to stabilize the quantization level of frozen weights. For activations, the scaling coefficient s is set to a trainable parameter like LSQ (Esser et al., 2020). Also, we adopt a layer-wise quantizer for both weights and activations.

3.2 PARTLY SCRATCH-OFF LOTTERY TICKET

The limited capacity of low-bit representation requires accomplishing network quantization in a quantization-aware training (QAT) manner so as to mitigate the performance gap between the quantized network and its full-precision counterpart. However, QAT involves cumbersome training costs since it requires dozens of training epochs (Esser et al., 2020; Lee et al., 2021b; Chen et al., 2021a). The heavy training costs barricades applications of quantized networks. Therefore, an efficient training method is urgent for the QAT community.

To this end, we dive into the training evolution of quantization level for each weight and unearth an interesting phenomenon that many network weights converge to their optimal quantization levels in the early training process. Concretely, we calculate the proportion of quantized weights that reaches the optimal quantization level along with the network training. Fig. 1 provides the results over various datasets and bit-widths. Taking Fig. 1(a) as an instance, for "layer2.conv1" of 2-bit ResNet-20 on CIFAR-100, the optimal quantized network is obtained at around epoch 320. However, about 60% weights reach their optimal quantization levels without any training. And it increases to 80% at epoch 80. A similar observation can be found at other bit-widths. For example, in Fig. 1(d), 20% weights already stay in the optimal quantization level before training a 4-bit ResNet-20 on CIFAR-100 and it becomes 40% at epoch 80.

The above phenomenon indicates that the half-trained quantized network contains a quantized subnet consisting of these well-optimized quantized weights. Alike to the lottery ticket in network pruning (Frankle & Carbin, 2019), we term these weights as partly scratch-off lottery ticket, which means they do not require to be trained from beginning to end. When diving into a deeper analysis, we attribute the existence of the partly scratch-off lottery ticket to two possible characteristics in network quantization. First, weights of the quantized network are often initialized from a pre-trained full-precision model. It can be expected that many weights start from an optimal or sub-optimal state. As a result, several training epochs lead to the convergence of many weights in their optimal quantization level. Second, compared with the full-precision data format, the quantization levels are discrete. Numerous continuous values in an interval are mapped to the same discrete state. Despite

that full-precision weights vary across different training epochs, they remain at the same quantization level. In particular to the lower bit-width case, more partly scratch-off lottery tickets can be observed since each interval becomes much larger. Fig. 1 demonstrates this potential: the 2-bit ResNet-20 manifests a higher ratio of the optimal quantization level than its 4-bit version in the early training process.

We realize the unnecessary expense of updating these tickets during most part of the training period since they already fall into their optimal quantization levels. The dense gradient computing for these tickets can be eliminated. Given that the partly scratch-off lottery ticket increases as network training, we can create a gradual weight gradient sparsity where more and more weights are frozen if reaching the optimal. Therefore, the factuality of partly scratch-off lottery tickets permits the possibility of efficient QAT training if we can well locate these tickets.

3.3 SPARSE QUANTIZATION-AWARE TRAINING

In this subsection, we introduce SpQAT, an efficient sparse QAT method, to discover the partly scratch-off lottery. By gradually pulling out weights from network training, the proposed SpQAT creates a sparse weight gradient and then shrinks the training costs of quantized networks.

Specifically, our SpQAT deploys a simple-yet-effective heuristic rule to find the partly scratch-off lottery ticket. At first, we train the quantized network for E_{wm} epochs as the warmup stage. Then, the partly scratch-off lottery comprises weights whose EMA distance is lower than a pre-defined threshold t. Herein, the EMA distance is the exponential moving average distance between the normalized weight $\boldsymbol{w_n}$ and its corresponding quantization level. At the i-th training iteration, the EMA distance of the given $\boldsymbol{w_n}$ is defined as:

$$D^i_{\boldsymbol{w_n}} = mD^{i-1}_{\boldsymbol{w_n}} + (1-m)D_{\boldsymbol{w_n}}, \tag{3}$$

where m is the momentum of the EMA function, and $D_{\boldsymbol{w_n}}$ is the distance between $\boldsymbol{w_n}$ and its corresponding quantization level. Note that $D^i_{\boldsymbol{w_n}}$ will be reset to the quantization interval $\Delta^B = \frac{2}{2^B}$ once the quantization level of $\boldsymbol{w_n}$ is changed. We then compare $D^i_{\boldsymbol{w_n}}$ with a threshold t to determine whether $\boldsymbol{w_n}$ should be frozen. If $D^i_{\boldsymbol{w_n}}$ is lower than the threshold t, $\boldsymbol{w_n}$ will be frozen, meaning that zeroing out the gradient of $\boldsymbol{w_n}$. Otherwise, $\boldsymbol{w_n}$ is continuously updated. The threshold t is computed by obtained Δ^B with a rate p:

$$t = \Delta^B \cdot p. \tag{4}$$

The constant quantization interval Δ^B is determined by the bit-width B. We adjust the rate p to control the ease of frozen weights. The selection of p provides a trade-off between the performance and training costs reduction. For example, a small p returns a small threshold t, which means a weight has to be very close to its quantization level if it is frozen. As a result, the identification of the partly scratch-off lottery is more accurate that is able to retain the performance. However, it is more difficult to freeze weights and more weights are prone to be updated. In this case, the weight gradient sparsity is limited and the reduced training costs are also limited. In contrast, a large p indicates a large threshold t. Subsequently, weights are more likely to be misidentified as the partly scratch-off lottery since the condition is loose. This misidentification will lead to performance degradation. However, it would be easier to freeze weights and achieve a higher weight gradient sparsity, so that the training costs can be reduced significantly.

To adjust the value of p, we design three strategies including fixing, linear-growth, and sine-growth. The fixing strategy indicates the rate p is set to a constant c that ranges from 0 to 1. In linear-growth strategy and sine-growth strategy, the rate p gradually increases in a linear manner and a sine manner when the warmup stage finishes, respectively. Supposing the total training epoch is E, the current iteration is i, and an epoch has T iterations. The linear-growth strategy is defined as:

$$p = \frac{i - T \cdot E_{wm}}{T \cdot E - T \cdot E_{wm}} \cdot \mathbb{I}(i > T \cdot E_{wm}), \tag{5}$$

where $\mathbb{I}()$ is the indicator function. The sine-growth is defined as:

$$p = \sin\big(\frac{i - T \times E_{wm}}{T \cdot E - T \cdot E_{wm}} \cdot \frac{\pi}{2}\big) \cdot \mathbb{I}(i > T \cdot E_{wm}). \tag{6}$$

Both linear-growth strategy and sine-growth strategy increase the rate p from 0 to 1.

Table 1: Results of ResNet-20 on CIFAR-100/10. "Avg. of WGS" indicates the average weight gradient sparsity. "RD. FLOPs of BP" indicates the reduction in FLOPs of backward propagation.

Datasets	Networks	W/A	Mode	Acc. (%)	Avg. of WGS	RD. FLOPs of BP
CIFAR-100	ResNet-20 (FP: 65.24%)	2/2	Baseline	62.99	0	0%
			SpQAT	**64.20**	0.58	29.0%↓
		3/3	Baseline	66.19	0	0%
			SpQAT	**66.90**	0.58	29.0%↓
		4/4	Baseline	66.71	0	0%
			SpQAT	**67.00**	0.65	32.5%↓
CIFAR-10	ResNet-20 (FP: 91.78%)	2/2	Baseline	90.27	0	0%
			SpQAT	**90.48**	0.66	33.0%↓
		3/3	Baseline	91.61	0	0%
			SpQAT	91.59	0.65	32.5%↓
		4/4	Baseline	91.88	0	0%
			SpQAT	91.83	0.68	34.0%↓

4 EXPERIMENTATION

4.1 EXPERIMENTAL SETTINGS

4.1.1 CLASSIFICATION TASK

We quantize ResNet-20 (He et al., 2016) for CIFAR-100/10 (Krizhevsky, 2009), ResNet-18, ResNet-50 (He et al., 2016), MobileNetV1 (Howard et al., 2017), and MobileNetV2 (Sandler et al., 2018) for ImageNet (Russakovsky et al., 2015). All code are implemented with Pytorch (Paszke et al., 2019). For experiments on CIFAR-100/10, we chose to quantize all layers of ResNet-20 into 2, 3, 4-bit and the last linear layer won't be fixed. As for experiments on challenging ImageNet, we retain the first and last layer as full-precision and chose to quantize the networks into 2, 3, 4, 6, 8-bit.

The quantized network is trained with the cross-entropy loss. The gradient of the round function is set to 1 by using the straight-through estimator (STE) (Courbariaux et al., 2016). The SGD optimizer is adopted with a momentum of 0.9. For CIFAR-100/10, the initial learning rate, batch size, weight decay, and total training epochs are set to 0.1, 256, 0.0001, and 400, respectively. The data augmentation consists of "random crop" and "random horizontal flip". For ImageNet, the initial learning rate, batch size, weight decay, and total training epochs are set to 0.01, 128, 0.0005, and 80, respectively. The learning rate is decayed by a factor of 0.1 for every 100 epochs on CIFAR-100/10, and every 20 epochs on ImageNet. The standard data augmentation is used including "random resize and crop", and "random horizontal flip". The pre-trained MobileNetV1 is obtained from *pytorchcv* and other models are downloaded from the *torchvision*. For all experiments, we employ the linear-growth strategy for it achieves a good balance between sparsity and accuracy. For the hyper-parameters, we empirically set m as 0.99 for CIFAR-100/10, and as 0.9999 for ImageNet without further search. The E_{wm} is set as 80 for CIFAR100/10, and as 16 for ImageNet by using the grid search. Despite they might not be optimal for all networks, we find these values already have provided satisfactory performance. The ablation study is provided in Appendix A.2.

To measure the reduced computational training costs, we report the average weight gradient sparsity, which is the sum of the weight gradient sparsity per iteration divided by the total number of iterations. The weight gradient sparsity per iteration is the ratio of the number of fixed weights to the number of all weights at the current iteration. We also employ the overall reduction in FLOPs of backward propagation as the metric. This metric is equal to half of the average weight gradient sparsity since SpQAT only eliminates the computation of weight gradient.

4.1.2 DETECTION TASK

To further demonstrate the generalization ability of the proposed SpQAT, we apply it to detection task. We employ the well-known open-source framework MMDetection (Chen et al., 2019) and choose to quantize Faster R-CNN (Ren et al., 2016) to 6, 8-bit. The experiments are conducted on Pascal VOC (Everingham et al., 2010). The implementation details are provided in Appendix A.3.

Table 2: Results of ResNet-18 and ResNet-50 on ImageNet.

Datasets	Networks	W/A	Mode	Acc. (%)	Avg. of WGS	RD. FLOPs of BP
ImageNet	ResNet-18 (FP: 69.64%)	2/2	Baseline	63.08	0	0%
			SpQAT	**64.92**	0.52	26.0%↓
		3/3	Baseline	68.37	0	0%
			SpQAT	**69.00**	0.51	25.5%↓
		4/4	Baseline	70.15	0	0%
			SpQAT	**70.21**	0.50	25.0%↓
		6/6	Baseline	70.90	0	0%
			SpQAT	70.71	0.39	19.5%↓
		8/8	Baseline	70.87	0	0%
			SpQAT	70.84	0.22	11.0%↓
	ResNet-50 (FP: 76.15%)	2/2	Baseline	69.67	0	0%
			SpQAT	**70.42**	0.52	26.0%↓
		3/3	Baseline	75.28	0	0%
			SpQAT	**75.66**	0.50	25.0%↓
		4/4	Baseline	76.30	0	0%
			SpQAT	**76.66**	0.47	23.5%↓
		6/6	Baseline	76.55	0	0%
			SpQAT	**76.71**	0.37	18.5%↓
		8/8	Baseline	76.48	0	0%
			SpQAT	**76.51**	0.21	10.5%↓

4.2 Results of Classification task

4.2.1 CIFAR-100/10

We first analyze the performance on CIFAR-100/10 by comparing the SpQAT against the baseline. The results are presented in Tab. 1. On CIFAR-100, the proposed SpQAR achieves around 0.6 average weight gradient sparsity and 30% backward propagation FLOPs reduction, but still consistently achieving better accuracy compared with the baseline, especially in 2, 3-bit. Specifically, SpQAT improves the performance by 1.21%, 0.71%, and 0.29% for 2, 3, 4-bit, respectively. For CIFAR-10, our method also yields better or comparable performance compared to the baseline, accompanied by more than 30% reduction in backward propagation FLOPs. These results well demonstrate the superiority of the proposed SpQAT.

4.2.2 ImageNet

We then conduct the experiments on the challenging large-scale ImageNet. In particular, we provide the results of four standard networks including ResNet-18/50 and MobileNetV1/V2.

ResNet-18/50. The results of ResNet-18 and ResNet-50 are presented in Tab. 2. For 6, 8-bit ResNet-18, the results show that our SpQAT eliminates 10% and 19.5% backward propagation FLOPs with negligible performance degradation. When it comes to 2, 3, and 4-bit, SpQAT obtains 50% average weight gradient sparsity and reduces the backward propagation FLOPs by 25%, while still exhibiting better accuracy than the baseline. For example, our SpQAT improves the accuracy by 1.84% for 2-bit ResNet-18. As for ResNet-50, we also observe that our SpQAT consistently outperforms the baseline on all bit-widths. Taking 2, 3, and 4-bit as examples, the backward propagation FLOPs are respectively reduced by 23.5%, 25%, and 25%, respectively, but the performance gains of our SpQAT are 0.75%, 0.38%, and 0.36%, respectively.

MobileNetV1/V2. We also present the results of MobileNetV1 and MobileNetV2 in Tab. 2. It can be find that SpQAT maintains better accuracy than the baseline on MobileNetV1. For 8-bit, our SpQAT reduce 8.5% backward propagation FLOPs and slightly outperform the baseline by 0.11%. As the bit-width goes down, the reduction of backward propagation FLOPs consistently increases. However, the performance superiority of our SpQAT becomes more siginificant. For instance, compared with the baseline, our SpQAT obtains 19.29%, 12.07%, and 3.47% accuracy improvements and eliminates 19.5%, 18.5%, 18.5% backward propagation FLOPs on 2, 3, and 4-

Table 3: Results of MobileNetV1 and MobileNetV2 on ImageNet.

Datasets	Networks	W/A	Mode	Acc. (%)	Avg. of WGS	RD. FLOPs of BP
ImageNet	MobileNetV1 (FP: 73.33%)	2/2	Baseline	35.71	0	0%
			SpQAT	**55.00**	0.39	19.5%↓
		3/3	Baseline	55.78	0	0%
			SpQAT	**67.85**	0.37	18.5%↓
		4/4	Baseline	68.09	0	0%
			SpQAT	**71.56**	0.37	18.5%↓
		6/6	Baseline	73.17	0	0%
			SpQAT	**73.51**	0.30	15.0%↓
		8/8	Baseline	73.56	0	0%
			SpQAT	**73.67**	0.17	8.5%↓
	MobileNetV2 (FP: 71.83%)	2/2	Baseline	25.15	0	0%
			SpQAT	**40.39**	0.34	17.0%↓
		3/3	Baseline	48.56	0	0%
			SpQAT	**57.75**	0.30	15.0%↓
		4/4	Baseline	60.74	0	0%
			SpQAT	**64.68**	0.29	14.5%↓
		6/6	Baseline	68.30	0	0%
			SpQAT	**68.55**	0.25	12.5%↓
		8/8	Baseline	69.57	0	0%
			SpQAT	69.35	0.16	8.0%↓

bit, respectively. The effectiveness of the proposed SpQAT is also demonstrated by the results of MobileNetV2. In particular, our SpQAT enjoys a higher performance compared with the baseline on all bit-widths except for 8-bit, where SpQAT is only slightly lower. Similar to MobileNetV1, for low bit-widths such as 2, 3, and 4-bit, the elimination of backward propagation FLOPs increases, while noticeable performance gains appear. On 2, 3, and 4-bit, the FLOPs of backward propagation decrease 17% 15% and 14.5%, and the performance gains are 15.24%, 9.19%, and 3.94%.

The performance improvements on low bit-widths can be attributed to the fact that the proposed SpQAT well solving the weight oscillations problem that is especially obvious for low bit-widths (Nagel et al., 2022). The weight oscillations problem refers to the quantized weight periodically oscillating between two quantization levels, which leads to unstable training and inferior performance. Although the original idea is to reduce training costs, our SpQAT subtly alleviates this problem by freezing weights during the training period and thus improves performance by a large margin.

4.2.3 ILLUSTRATIONS ON WEIGHT GRADIENT SPARSITY

We present the curve of the weight gradient sparsity of quantized weights *w.r.t.* training epochs in Fig. 3. It can be seen that the proposed SpQAT results in a gradual sparsity. The sparsity gradually increases until it reaches an extremely high magnitude. Take Fig. 3(a) as an example, for ResNet-20 on CIFAR-100, the weight gradient sparsity increases rapidly after the warmup stage finishes. After epoch 240, the sparsity is close to 100%, which means most quantized weights are frozen. As shown in Fig. 3(b), the results of ResNet-20 on CIFAR-10 are similar to CIFAR-100. As for the large-scale ImageNet, we observe that the sparsity starts to raise from epoch 20 to 45, depending on the bit-width. Then, the sparsity rapidly increases to a very high level. Concretely, Fig. 3(c) provides the results of ResNet-18. Despite the raise point of sparsity increase being different, our SpQAT still achieves over 80% at epoch 50 in general. In particular, it can exceed 90% at epoch 55. Fig. 3(c)-Fig. 3(f) present the sparsity of ResNet-50, MobileNetV1, and MobileNetV2. Despite the raise points of the sparsity of bit-widths being different, a high sparsity can be achieved at epoch 50. For instance, in Fig. 3(e), the sparsity of 2-bit ResNet-50 starts to increase quickly at epoch 20. While for 6-bit ResNet-50, its raise point of sparsity increase is epoch 40. After the raise point, the sparsity raises quickly, and ResNet-50 of all bit-widths exceed 80% sparsity at epoch 50. Fig. 3 demonstrates our SpQAT is able to create an extreme high sparsity after the middle of the training period.

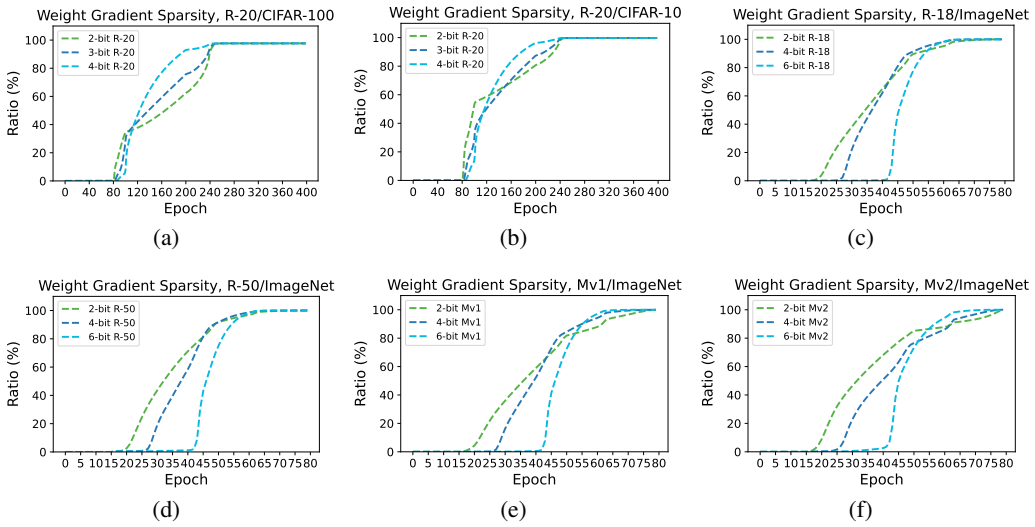


Figure 3: The weight gradient sparsity of quantized layers *w.r.t.* training epochs. "WBAB" indicates the weights and activations are quantized to B-bit. "R-20", "R-18", "R-50", "Mv1", and "Mv2" indicate ResNet-20, ResNet-18, ResNet-50, MobileNetV1, and MobileNetV2, respectively.

Table 4: Results of Faster R-CNN on Pascal VOC. "Avg. of WGS" indicates the average weight gradient sparsity. "RD. FLOPs of BP" indicates the reduction in FLOPs of backward propagation.

Datasets	Networks	W/A	Mode	Acc. (%)	Avg. of WGS	RD. FLOPs of BP
Pascal VOC	Faster R-CNN (FP: 0.804)	6/6	Baseline	0.758	0	0%
			SpQAT	0.754	0.45	22.5%↓
		8/8	Baseline	0.762	0	0%
			SpQAT	**0.763**	0.17	8.5%↓

4.3 Quantitative results of detection task

The quantitative results of detection task are presented in Tab. 4. We find that the performance of our SpQAT is on par with the normal training even though the training costs are eliminated a lot. Concretely, for 8-bit Faster R-CNN, SpQAT provides 0.763 mAP while reducing the FLOPs of backward propagation by 8.5% and obtaining 0.17 average weight gradient sparsity. In contrast, the normal training leads to 0.762 mAP. As for 6-bit Faster R-CNN, our SpQAT is able to eliminate 22.5% backward propagation FLOPs and merely results in a negligible performance drop (0.754 *v.s.* 0.758).

5 Conclusion

In this paper, we investigate reducing the training costs of quantization-aware training (QAT) methods for the first time. We discover a straightforward-yet-valuable observation that a large portion of quantized weights, named as the partly scratch-off lottery ticket, converge to the optimal quantization level after a few training epochs. Inspired by this, we develop an efficient sparse QAT method, dubbed SpQAT, to gradually freeze these weights during the training period and thus reduce tedious weight gradient calculations. Specifically, a weight is frozen once the distance between it and its corresponding quantization level is lower than a threshold, which is controlled by the proposed strategy. The introduced SpQAT is simple but extremely effective in accurately identifying the partly scratch-off lottery ticket and leading to a sparse weight gradient. Extensive experiments on various networks, bit-widths, and datasets demonstrate that the proposed SpQAT achieves 20%-60% weight gradient sparsity in general while still exhibiting comparable or even better performance than the compared baseline.

REFERENCES

Takuya Akiba, Shuji Suzuki, and Keisuke Fukuda. Extremely large minibatch sgd: Training resnet-50 on imagenet in 15 minutes. *arXiv preprint arXiv:1711.04325*, 2017.

Ron Banner, Itay Hubara, Elad Hoffer, and Daniel Soudry. Scalable methods for 8-bit training of neural networks. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5151–5159, 2018.

Ron Banner, Yury Nahshan, Daniel Soudry, et al. Post training 4-bit quantization of convolutional networks for rapid-deployment. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 7950–7958, 2019.

Adrian Bulat, Brais Martinez, and Georgios Tzimiropoulos. High-capacity expert binary networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

Peng Chen, Jing Liu, Bohan Zhuang, Mingkui Tan, and Chunhua Shen. Aqd: Towards accurate quantized object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 104–113, 2021a.

Weihan Chen, Peisong Wang, and Jian Cheng. Towards mixed-precision quantization of neural networks via constrained optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5350–5359, 2021b.

Brian Chmiel, Liad Ben-Uri, Moran Shkolnik, Elad Hoffer, Ron Banner, and Daniel Soudry. Neural gradients are near-lognormal: improved quantized and sparse training. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

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.

Zhen Dong, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Hawq: Hessian aware quantization of neural networks with mixed-precision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 293–302, 2019.

Steven K. Esser, Jeffrey L. McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S. Modha. Learned step size quantization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. Rigging the lottery: Making all tickets winners. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 2943–2952. PMLR, 2020.

M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

Negar Goli and Tor M Aamodt. Resprop: Reuse sparsified backpropagation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1548–1558, 2020.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4852–4861, 2019.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

Song Han, Jeff Pool, John Tran, William J Dally, et al. Learning both weights and connections for efficient neural network. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 1135–1143, 2015.

Tiantian Han, Dong Li, Ji Liu, Lu Tian, and Yi Shan. Improving low-precision network quantization via bin regularization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5261–5270, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Elad Hoffer, Ron Banner, Itay Golan, and Daniel Soudry. Norm matters: efficient and accurate normalization schemes in deep networks. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 2164–2174, 2018.

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.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2704–2713, 2018.

Xianyan Jia, Shutao Song, Wei He, Yangzihao Wang, Haidong Rong, Feihu Zhou, Liqiang Xie, Zhenyu Guo, Yuanzhou Yang, Liwei Yu, et al. Highly scalable deep learning training system with mixed-precision: Training imagenet in four minutes. *arXiv preprint arXiv:1807.11205*, 2018.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4350–4359, 2019.

Dohyung Kim, Junghyup Lee, and Bumsub Ham. Distance-aware quantization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5271–5280, 2021.

Raghuraman Krishnamoorthi. Quantizing deep convolutional networks for efficient inference: A whitepaper. *arXiv preprint arXiv:1806.08342*, 2018.

Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 2009.

Jung Hyun Lee, Jihun Yun, Sung Ju Hwang, and Eunho Yang. Cluster-promoting quantization with bit-drop for minimizing network quantization loss. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5370–5379, 2021a.

Junghyup Lee, Dohyung Kim, and Bumsub Ham. Network quantization with element-wise gradient scaling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6448–6457, 2021b.

Sunwoo Lee, Jeongwoo Park, and Dongsuk Jeon. Toward efficient low-precision training: Data format optimization and hysteresis quantization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

Yuhang Li, Xin Dong, and Wei Wang. Additive powers-of-two quantization: An efficient non-uniform discretization for neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

Mingbao Lin, Rongrong Ji, Yan Wang, Yichen Zhang, Baochang Zhang, Yonghong Tian, and Ling Shao. Hrank: Filter pruning using high-rank feature map. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1529–1538, 2020a.

Mingbao Lin, Rongrong Ji, Zihan Xu, Baochang Zhang, Yan Wang, Yongjian Wu, Feiyue Huang, and Chia-Wen Lin. Rotated binary neural network. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 7474–7485, 2020b.

Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2117–2125, 2017.

Shiwei Liu, Tianlong Chen, Xiaohan Chen, Zahra Atashgahi, Lu Yin, Huanyu Kou, Li Shen, Mykola Pechenizkiy, Zhangyang Wang, and Decebal Constantin Mocanu. Sparse training via boosting pruning plasticity with neuroregeneration. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 9908–9922, 2021.

Zechun Liu, Kwang-Ting Cheng, Dong Huang, Eric P Xing, and Zhiqiang Shen. Nonuniform-to-uniform quantization: Towards accurate quantization via generalized straight-through estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4942–4952, 2022.

Hesham Mostafa and Xin Wang. Parameter efficient training of deep convolutional neural networks by dynamic sparse reparameterization. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 4646–4655. PMLR, 2019.

Markus Nagel, Marios Fournarakis, Yelysei Bondarenko, and Tijmen Blankevoort. Overcoming oscillations in quantization-aware training. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 16318–16330, 2022.

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. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 8026–8037, 2019.

Md Aamir Raihan and Tor Aamodt. Sparse weight activation training. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 15625–15638, 2020.

S Ren, K He, R Girshick, and J Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 39(6):1137–1149, 2016.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115:211–252, 2015.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4510–4520, 2018.

Xu Sun, Xuancheng Ren, Shuming Ma, and Houfeng Wang. meprop: Sparsified back propagation for accelerated deep learning with reduced overfitting. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 3299–3308. PMLR, 2017.

Linjie Yang and Qing Jin. Fracbits: Mixed precision quantization via fractional bit-widths. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 35, pp. 10612–10620, 2021.

Yang You, Zhao Zhang, Cho-Jui Hsieh, James Demmel, and Kurt Keutzer. Imagenet training in minutes. In *Proceedings of the 47th International Conference on Parallel Processing*, pp. 1–10, 2018.

Zhaoyang Zhang, Wenqi Shao, Jinwei Gu, Xiaogang Wang, and Ping Luo. Differentiable dynamic quantization with mixed precision and adaptive resolution. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 12546–12556. PMLR, 2021.

Kang Zhao, Sida Huang, Pan Pan, Yinghan Li, Yingya Zhang, Zhenyu Gu, and Yinghui Xu. Distribution adaptive int8 quantization for training cnns. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 35, pp. 3483–3491, 2021.

Yunshan Zhong, Mingbao Lin, Gongrui Nan, Jianzhuang Liu, Baochang Zhang, Yonghong Tian, and Rongrong Ji. Intraq: Learning synthetic images with intra-class heterogeneity for zero-shot network quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12339–12348, 2022.

Aojun Zhou, Anbang Yao, Yiwen Guo, Lin Xu, and Yurong Chen. Incremental network quantization: Towards lossless cnns with low-precision weights. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

Feng Zhu, Ruihao Gong, Fengwei Yu, Xianglong Liu, Yanfei Wang, Zhelong Li, Xiuqi Yang, and Junjie Yan. Towards unified int8 training for convolutional neural network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1969–1979, 2020.

Bohan Zhuang, Lingqiao Liu, Mingkui Tan, Chunhua Shen, and Ian Reid. Training quantized neural networks with a full-precision auxiliary module. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1488–1497, 2020.

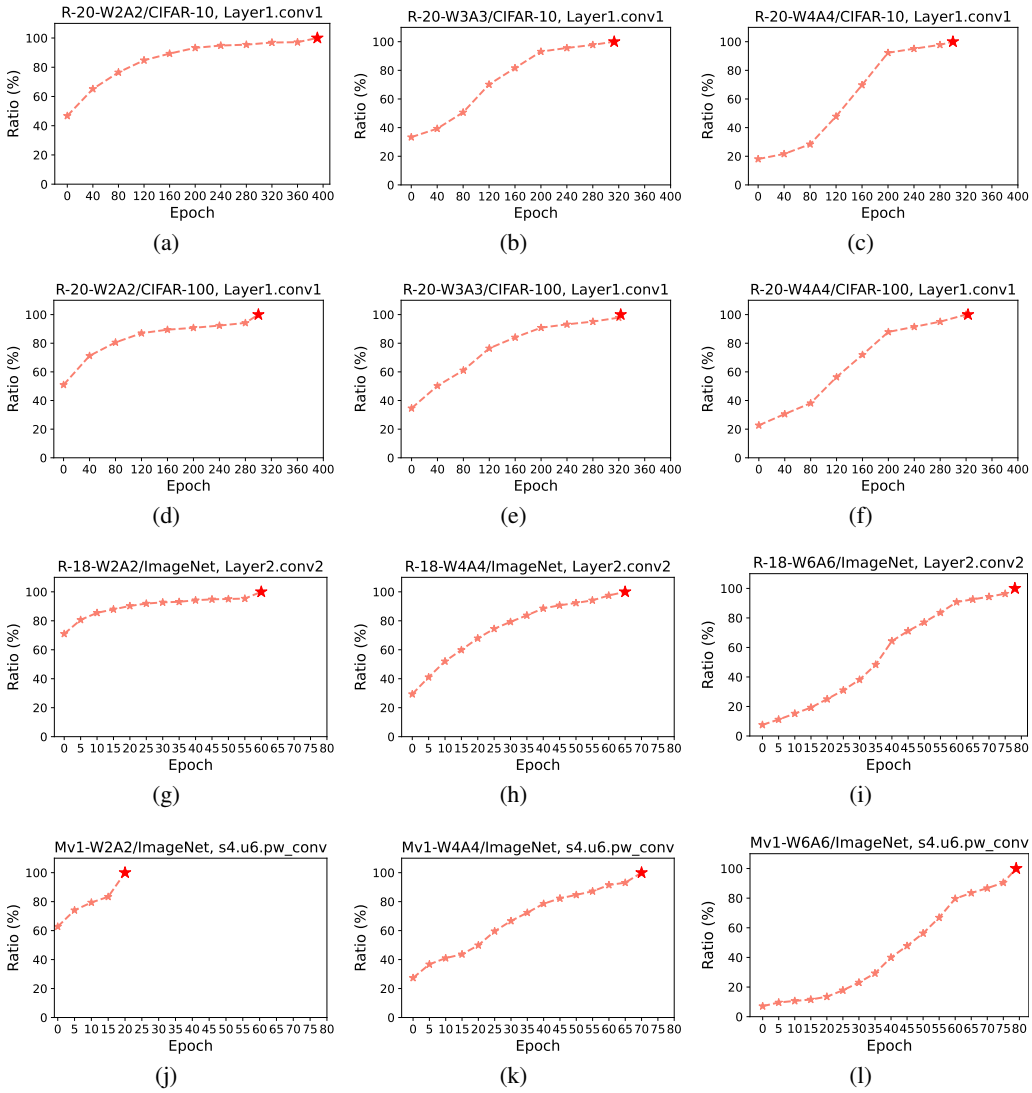


Figure 4: The ratio of the weights reaching the optimal quantization level *w.r.t.* training epochs. The red pentagon denotes the best model. "WBAB" indicates the weights and activations are quantized to B-bit. "R-20", "R-18", and "mv1" indicate ResNet-20, ResNet-18, and MobileNetV1, respectively.

A Appendix

A.1 More Illustration of Partly Scratch-off Lottery

In this subsection, we provide more illustrations of the partly scratch-off lottery ticket. As presented in Fig. 4, we provide the results of 2, 3, 4-bit ResNet-20 on CIFAR-100 in Fig. 4(a) - Fig. 4(c), 2, 3, 4-bit ResNet-20 on CIFAR-10 in Fig. 4(d) - Fig. 4(f), 2, 4, 6-bit ResNet-18 on ImageNet in Fig. 4(g) - Fig. 4(i), and 2, 4, 6-bit MobileNetV1 on ImageNet in Fig. 4(j) - Fig. 4(l).

As can be seen, all illustrations clearly demonstrate the existence of the partly scratch-off lottery ticket. For example, as shown in Fig. 4(a), we can find that over 40% weights already reach the optimal quantization level without training for "layer1.conv1" of 2-bit ResNet-20 on CIRAR-10. After being trained for 80 epochs, abount 80% weights reach the optimal quantization level. For other bit-widths, we also observe similar phenomena. As shown in Fig. 4(i), for 6-bit ResNet-18 on ImageNet, over 20% weights reach the optimal quantization level after 10 epochs. The ratio of the partly scratch-off lottery also increases rapidly as the training epoch increases. As presented

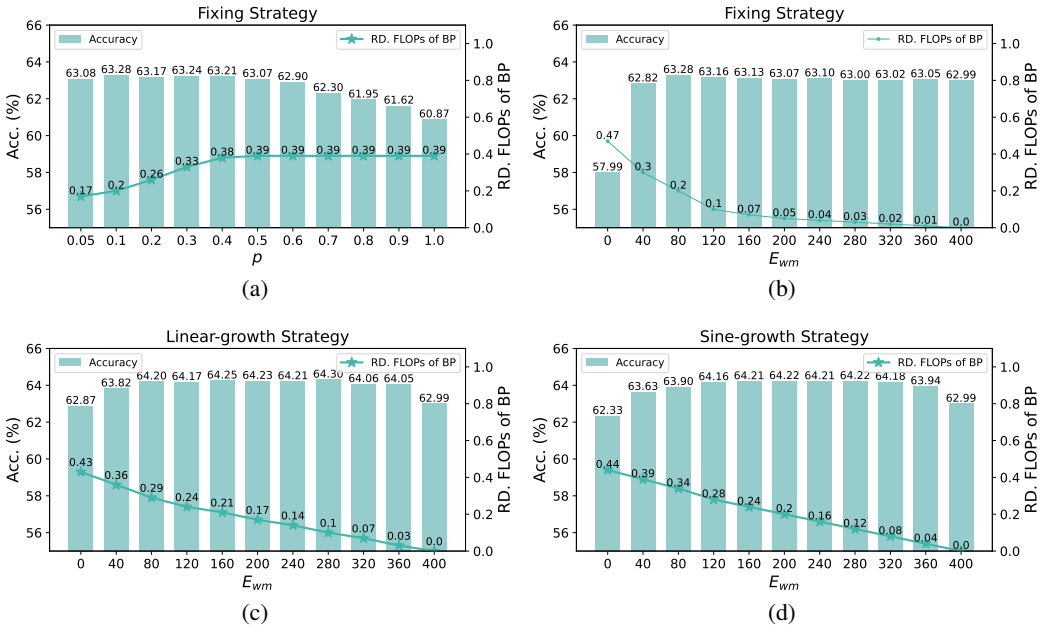


Figure 5: Influence of the hyper-parameters on the top-1 accuracy of 2-bit ResNet-20 on CIFAR-100. (a) and (b) provide the ablation studies of the fixing strategy. (c) and (d) provide the ablation studies of the linear-growth strategy and sine-growth strategy, respectively. "RD. FLOPs of BP" indicates the reduction in FLOPs of backward propagation.

in Fig. 4(j) - Fig. 4(l), the results of MobileNetV1 exhibit the same trend as other networks. For instance, without any training, the 2-bit MobileNetV1 still has over 60% weights being the optimal quantization level as shown in Fig. 4(j). As for 4-bit MobileNetV1, about 40% weights are able to reach the optimal quantization level after only 10 epochs of training as illustrated in Fig. 4(k).

A.2 Ablation Study

In this section, we first conduct ablation studies of the proposed three strategies and hyper-parameters of our SpQAT. All experiments are conducted by quantizing ResNet-20 to 2-bit on CIFAR-100. The top-1 accuracy and the reduction in FLOPs of backward propagation are reported.

Fixing Strategy. The fixing strategy has two hyper-parameters including p and E_{wm}. Fig. 5(a) and Fig. 5(b) respectively provide the top-1 accuracy of 2-bit ResNet-20 on CIFAR100 *w.r.t* p and E_{wm}. From Fig. 5(a), it can be observe that the reduction in FLOPs of backward propagation increases along with the increase of p and the best accuracy is obtain when $p = 0.1$. As presented in Fig. 5(b), when $E_{wm} = 80$, a good trade-off between accuracy and backward propagation FLOPs reduction is achieved. After that, continued to increase E_{wm} won't lead to great gains.

Linear-growth Strategy. As illustrated in Fig. 5(c), as the warmup epochs E_{wm} increases, the backward propagation FLOPs reduction consistently decrease. The performance reaches 64.20% and the FLOPs reduction is 0.29 at $E_{wm} = 80$. When $E_{wm} > 80$, the accuracy has not been greatly improved or even decreased.

Sine-growth Strategy. From Fig. 5(d), we can find that the accuracy is 64.16% when $E_{wm} = 120$. After that, the performance improvement is marginal or negative despite the backward propagation FLOPs reduction decreasing rapidly.

Comparing these three strategies, we can observe that the linear-growth strategy and the sine-growth strategy obtain a better performance than the fixing strategy. Moreover, the linear-growth strategy enjoys higher FLOPs reduction than the sine-growth when they have comparable accuracy. Thus, we consider it achieves the best trade-off between performance and the backward propagation FLOPs reduction, and thus employ this strategy in all following experiments.

A.3 EXPERIMENTAL SETTINGS OF DETECTION TASK

All hyper-parameters, data-processing operation, and training scheduler are consistent with the default setting of MMDetection (Chen et al., 2019) except the initial learning is divided by 10 to stabilize the training process of the quantized network. For the results of Pascal VOC, the network is trained with the train set and Val set of Pascal VOC-2007 and Pascal VOC-2012, and is tested on the test set of Pascal VOC-2007. Related settings can be found in MMDetection. For the settings of the SpQAT, we use linear-growth strategy and the ema momentum m is set to 0.9999 based on the empirical conclusion on ImageNet. We report mAP as the evaluation metric.

We employ a Faster R-CNN with a ResNet-50 as the backbone and feature pyramid networks (FPN) (Lin et al., 2017). We quantize all layers of the Faster R-CNN. The convolutional layer is quantized by using the quantizer in Sec. 3.1. For quantizing the MLP layer, the output of Eq. (2) will be re-scale by the scaling coefficient s since there is no batch normalization layer after the MLP layer. In this case, we set the scaling coefficient s as a trainable parameter. The quantized networks are initialized from the pre-trained model provided by the MMDetection.