Supplementary Materials of "Sub-bit Neural Networks: Learning to Compress and Accelerate Binary Neural Networks"

Yikai Wang[1*] Yi Yang[2] Fuchun Sun[1] Anbang Yao[2]
[1]Beijing National Research Center for Information Science and Technology (BNRist),
State Key Lab on Intelligent Technology and Systems,
Department of Computer Science and Technology, Tsinghua University [2]Intel Corporation
{wangyk17@mails., fcsun@}tsinghua.edu.cn, {yi.b.yang, anbang.yao}@intel.com

A. Training Details

For image classification experiments on both CIFAR10 and ImageNet datasets, networks are trained from scratch. We set the batch size to 256, and use an SGD optimizer with a momentum of 0.9. The weight decay rate is assigned to 10^{-4}. We initialize the learning rate to 0.1 and adopt a cosine learning rate scheduler. Following IR-Net [9], we balance and standardize weights in the forward propagation and adopt the error decay estimator to approximate the sign function in the backward propagation. We select $\mathrm{Hardtanh}$ as the activation function instead of ReLU when we binarize activations [9]. To networks considered in the experiments, we use standard data augmentation strategies following the original papers [8, 10, 13]. On CIFAR10, we train each model on a single V100 GPU with 1000 epochs. On ImageNet, we train each model on four V100 GPUs with 120 epochs for ResNet-18 and ResNet-34.

For object detection, on both PASCAL VOC and MS-COCO datasets, networks are pre-trained on ImageNet [2] classification dataset. Following BiDet [12], the batch size is set to 32, and we adopt the Adam optimizer [5]. The learning rate is initialized to 0.001 which decays twice by multiplying 0.1 at the 6-th and 10-th epoch out of 12 epochs [5]. We adopt four V100 GPUs for training.

B. More Visualizations

To figure out how the subsets refinement module affects the binary kernels during training, we provide more visualization results in Figure 11. We observe that for different bit-width settings, binary kernels tend to be grouped and symmetric, especially in 0.56-bit and 0.44-bit subfigures. Besides, after training, binary kernels in SNNs present similar distributions in the same layers to some extent.

*This research was done when Yikai Wang was an intern at Intel Labs China, supervised by Anbang Yao who is responsible for correspondence.

C. Extending to Bottleneck with 1×1 Kernels

In our main paper, we focus on 3×3 kernels to design our SNNs, as 3×3 convolutional layers usually occupy major parameters and computational costs in many backbones such as VGG-small, ResNet-18, ResNet-34, etc. Yet in larger backbones like ResNet-50, the Bottleneck structure also contains 1×1 convolutional layers of which both the parameters and computations cannot be ignored. Therefore in this part, we introduce how our method can be naturally extended to such architectures.

For a 1×1 convolutional layer, assuming its layer index is i, the weights can be denoted as $\mathbf{w}^i \in \mathbb{R}^{c^i_{out}\cdot c^i_{in}\times1\times1}$. As in modern backbones c^i_{in} is usually divisible by 8, we split $\mathbf{w}^i$ into $c^i_{out}\cdot\frac{c^i_{in}}{8}$ vectors with each vector denoted as $\mathbf{w}^i_c \in \mathbb{R}^{8\times1\times1}$, where $c=1,2,\cdots,c^i_{out}\cdot\frac{c^i_{in}}{8}$. In this case, binarizing $\mathbf{w}^i_c$ is formulated as $\bar{\mathbf{w}}^i_c = \arg\min_{\mathbf{k}\in\mathbb{K}'}\|\mathbf{k}-\mathbf{w}^i_c\|_2^2$, where $\mathbb{K}'=\{\pm1\}^{8\times1\times1}$ and $|\mathbb{K}'|=256$. Again, we sample layer-specific subsets $\mathbb{P}'^i\subset\mathbb{K}'$. Under the circumstance, if we use τ'-bit to represent a vector $\mathbf{w}^i_c$, there is $|\mathbb{P}'^i|=2^{\tau'}$. By using indices $1,2,3,\cdots,2^{\tau'}$ to represent each binarized vector $\bar{\mathbf{w}}^i_c$, we can obtain a compression ratio $\frac{\tau'}{8}$.

To facilitate the understanding, Table 5 illustrates the comparison of the binarization for 3×3 kernels (proposed in our main paper) and for 1×1 kernels.

In Figure 6 of our main paper, we introduce a method that accelerates the 3×3 convolution operations of SNNs. 1×1 convolutional layers of SNNs can also be accelerated by the similar computation sharing.

Table 6 provides experimental results based on ResNet-50, and we again provide the full results for comparison. We adopt three bit-width settings including 0.59-bit, 0.47-bit, and 0.35-bit, which correspond to setting both τ and τ' to $5,4,3$ respectively. We observe that the $0.47/32$-bit setting with ResNet-50 only drops 0.6% in top-1 accuracy compared with the $1/32$-bit counterpart, yet it achieves $2.13\times$ parameter reduction and $5.27\times$ Bit-OPs reduction.

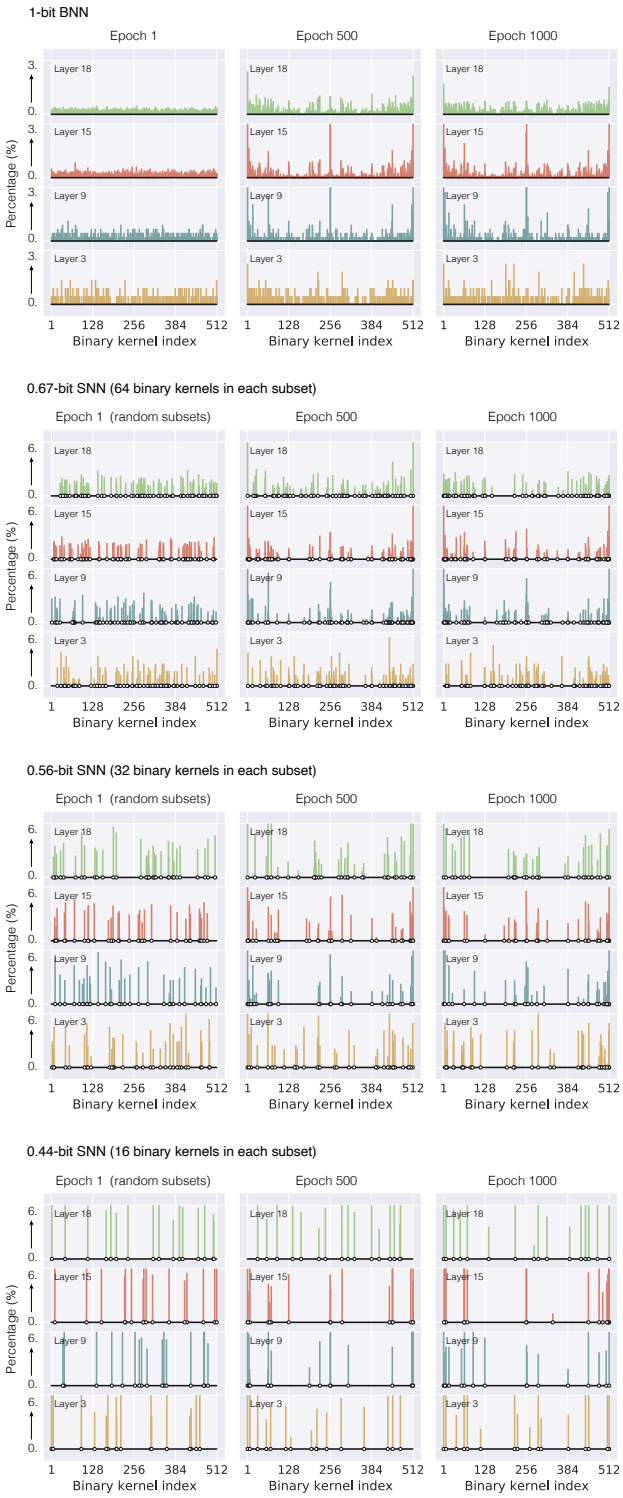

Figure 11. Visualization of how indices and frequencies of binary kernels change during training with subsets refinement. We provide results for 0.67-bit, 0.56-bit, and 0.44-bit SNNs, as well as a 1-bit BNN for comparison. Experiments are performed on CIFAR10 with ResNet-20.

Table 5. Comparison of binarization methods including BNN and SNN for 3×3 and 1×1 convolutional layers, including weights representation, element of a set/subset, set/subset representation, bit numbers of $\mathbf{w}_c^i$ and per weight.

		3×3 kernel size	1×1 kernel size
Weights		$\mathbf{w}^i \in \mathbb{R}^{c_{out}^i \cdot c_{in}^i \times 3 \times 3}$	$\mathbf{w}^i \in \mathbb{R}^{c_{out}^i \cdot c_{in}^i \times 1 \times 1}$
Element of a set/subset		$\mathbf{w}_c^i \in \mathbb{R}^{1\times 3\times 3}$ (Kernel)	$\mathbf{w}_c^i \in \mathbb{R}^{8\times 1\times 1}$ (Vector)
Number of units		$c_{out}^i \cdot c_{in}^i$	$c_{out}^i \cdot \frac{c_{in}^i}{8}$
BNN	Full set	$\mathbb{K} = \{\pm 1\}^{1\times 3\times 3}$	$\mathbb{K}' = \{\pm 1\}^{8\times 1\times 1}$
	Bits of $\mathbf{w}_c^i$	9-bit	8-bit
	Bits per weight	1-bit	1-bit
SNN	Subset	$\mathbb{P}^i \subset \mathbb{K}$, $\lvert\mathbb{P}^i\rvert = 2^\tau$	$\mathbb{P}'^i \subset \mathbb{K}'$, $\lvert\mathbb{P}'^i\rvert = 2^{\tau'}$
	Bits of $\mathbf{w}_c^i$	τ-bit, $1 \le \tau < 9$	τ'-bit, $1 \le \tau < 8$
	Bits per weight	$\frac{\tau}{9}$-bit	$\frac{\tau'}{8}$-bit

Table 6. Results on the CIFAR10 dataset to verify our method on ResNet-50 which contains 1×1 convolutional layers. Single-crop testing with 32×32 crop size is adopted, and each result of our method is the average of three runs. Numbers highlighted in green are reduction ratios over BNN counterparts. * indicates our implemented results.

Method	Bit-width (W/A)	#Params (Mbit)	Bit-OPs (G)	Top-1 Acc. (%)
ResNet-50 (Extending to Bottleneck with 1×1 convolutional kernels)				
Full precision	32/32	750.26	78.12	95.4
IR-Net* [9]	1/1	23.45	1.221	93.2
Vanilla-SNN | SNN	0.59/1	13.87 (1.7×)	0.333 (3.7×)	92.1 | 92.9
Vanilla-SNN | SNN	0.47/1	11.09 (2.1×)	0.239 (5.1×)	91.4 | 92.4
Vanilla-SNN | SNN	0.35/1	8.321 (2.8×)	0.191 (6.4×)	91.0 | 92.1
IR-Net* [9]	1/32	23.45	39.06	95.1
Vanilla-SNN | SNN	0.59/32	13.87 (1.7×)	10.67 (3.7×)	94.4 | 95.1
Vanilla-SNN | SNN	0.47/32	11.09 (2.1×)	7.640 (5.1×)	93.8 | 94.5
Vanilla-SNN | SNN	0.35/32	8.321 (2.8×)	6.127 (6.4×)	93.5 | 94.0

D. Evaluation on Object Detection

To prove the generalization of our SNNs on object detection, we further perform experiments on PASCAL VOC [3] and MS-COCO 2014 [6] datasets. PASCAL VOC contains images within 20 different categories. The same with [12], we train our SNNs using VOC 2007 and VOC 2012 trainval-sets (16k images) and evaluate on VOC 2007 test-set (5k images). MS-COCO 2014 dataset contains images within 80 different categories. We follow the popular "trainval35k" and "minival" data split method [1, 12], which sets up training with the combination of the training set (80k images) as well as sampled images from the validation set (35k images), and adopts the remaining 5k images in the validation set for testing.

We choose two typical pipelines including SSD300 [7] and Faster R-CNN [11], using VGG16 and ResNet-18 as backbones respectively. Following the standard evaluation metrics [6], we report the average precision (AP) for IoU$\in [0.5{:}0.05{:}0.95]$, denoted as mAP, and AP_{50}, AP_{75} as well.

For a fair comparison, we follow the same training settings with BiDet [12], which achieves state-of-the-art binarization performance for the object detection task. Evalu-

Table 7. Object detection results on the PASCAL VOC and MS-COCO 2014 datasets. Single-crop testing with crop size 300×300 for SSD300, and 600×1000 for Faster R-CNN. We follow the training details and techniques in BiDet [12], which could be the reference for our 1-bit baselines.

Method	Bit-width (W/A)	VOC mAP (%)	MS-COCO 2014 mAP (%)	MS-COCO 2014 AP_{50} (%)	MS-COCO 2014 AP_{75} (%)
VGG16, SSD					
Full precision	32/32	72.4	23.2	41.2	23.4
BNN [4]	1/1	42.0	6.2	15.9	3.8
XNOR [10]	1/1	50.2	8.1	19.5	5.6
Bi-Real [8]	1/1	63.8	11.2	26.0	8.3
BiDet [12]	1/1	66.0	13.2	28.3	10.5
Vanilla-SNN | SNN	0.67/1	64.0 | 65.1	12.1 | 12.8	27.4 | 27.9	9.4 | 10.1
Vanilla-SNN | SNN	0.56/1	63.3 | 64.2	11.2 | 11.9	26.3 | 27.1	8.9 | 9.5
Vanilla-SNN | SNN	0.44/1	61.8 | 62.9	10.1 | 11.0	24.9 | 25.6	8.0 | 8.7
ResNet-18, Faster R-CNN					
Full precision	32/32	74.5	26.0	44.8	27.2
BNN [4]	1/1	35.6	5.6	14.3	2.6
XNOR [10]	1/1	48.4	10.4	21.6	8.8
Bi-Real [8]	1/1	58.2	14.4	29.0	13.4
BiDet [12]	1/1	59.5	15.7	31.0	14.4
Vanilla-SNN | SNN	0.67/1	57.6 | 58.8	14.3 | 15.1	29.6 | 30.5	13.3 | 13.8
Vanilla-SNN | SNN	0.56/1	56.9 | 57.9	13.5 | 14.3	29.0 | 29.9	12.4 | 13.2
Vanilla-SNN | SNN	0.44/1	55.2 | 56.4	12.4 | 13.3	27.7 | 28.7	11.8 | 12.5

ation results on both datasets are reported in Table 7, and we also consider 0.67-bit, 0.56-bit, and 0.44-bit settings for our SNNs. Regarding Faster-RCNN on the PASCAL VOC dataset, reducing the bit-width to 0.56, mAP has a slight drop of 1.6%; using 0.67-bit on the COCO dataset achieves very close performance compared with 1-bit BiDet, with only 0.6% mAP drop. These evaluation results again prove the effectiveness of our proposed method.

E. Other Possible Baselines

Besides the methods discussed and compared in the experiments of the main paper, we also consider another two possible methods as baselines. 1) Baseline A: Since each binary kernel is assigned an index from 1 to 512, a straightforward solution is to simply select binary kernels uniformly with the same index interval. 2) Baseline B: At every iteration or every 10 iterations, we select the top 2^τ most frequent binary kernels from the corresponding 1-bit BNNs as the subset for each layer. Results of both baselines and our method are provided in Table 8. Here, experiments are conducted on ImageNet with ResNet-18. We find that Baseline A achieves much lower performance than our proposed method, even Vanilla-SNN. We speculate that uniformly selecting binary kernels impacts the network representation ability as subsets in different layers are the same. In contrast, Baseline B seems to be a stronger baseline and it also surpasses Vanilla-SNN. However, the counting and sorting processes slow down the training. The training speed issue of Baseline B could be alleviated by updating subsets every 10 training iterations instead of 1, but accordingly the accuracy slightly drops. These results verify that our SNN is superior in both performance and training efficiency.

Table 8. Results on the ImageNet dataset with ResNet-18 to verify two possible baseline methods described in Sec. E. "1 iteration" and "10 iterations" indicate updating binary kernels every iteration and every 10 iterations, respectively.

Method	Bit-width (W/A)	Top-1 Acc. (%)
Baseline A	0.56/32	60.5
Baseline B	0.56/32	62.7
Baseline B-10	0.56/32	62.5
Vanilla-SNN | SNN	0.56/32	62.8 | 63.4

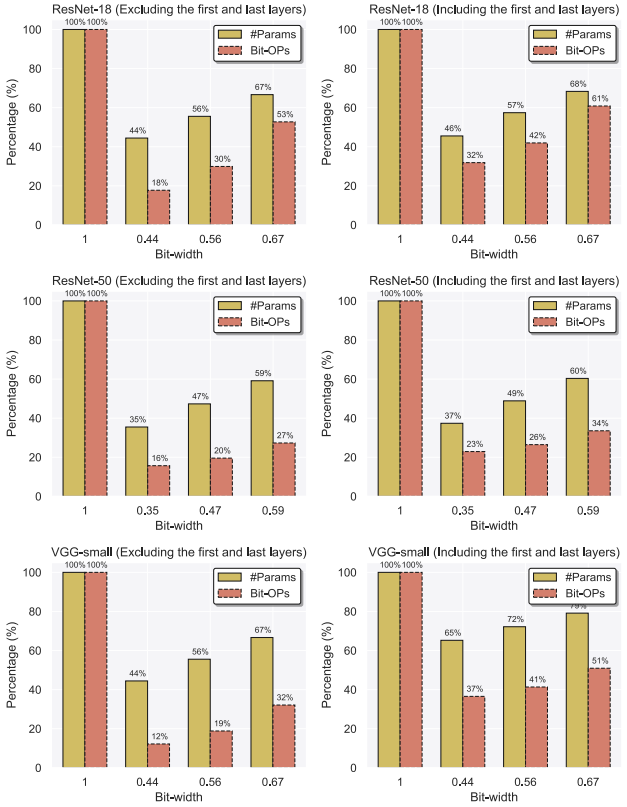

Figure 12. Reduction ratios *w.r.t.* parameters and bit-wise operations compared with 1-bit BNNs. Subfigures in the left column provide the comparison when excluding the first and last layers. Subfigures in the right column provide the comparison when including the first and last layers with full-precision weights. Experiments are conducted on CIFAR10 with ResNet-18, ResNet-50, and VGG-small. Activations are not binarized in the models.

F. Compression and Acceleration Including the First and Last layers

Regarding the common paradigm [8, 9, 10] of network binarization, weights in the first layer and last layer are full-precision weights, and in other layers are binarized. Our comparisons for parameters and bit-wise operations so far have excluded the first and last layers. To further verify the efficiency of our method when considering the whole network, Figure 12 provides the comparison when excluding and including the first and last layers. We observe that our

Figure 13. Timeline comparison between a 0.56-bit SNN and a 1-bit BNN, based on ResNet-18. This visualization is a supplement to Table 4 of our main paper. The running time is evaluated with the 224×224 image input, and a hardware configuration of Intel® CoFluent™ 64PEs@1GHz. Zoom in for the best view.

method can as well achieve good compression and acceleration performance even taking the first and last layers (with full-precision weights) into consideration. Note that VGG-small does not have a global average pooling layer, and thus its fully connected layer occupies more parameters and operations than the other two backbones.

G. Other Details of Practical Deployment

In Sec. 5.3 of our main paper, we conduct practical deployment and compare the total running time of 0.56-bit SNNs and 1-bit BNNs. Here, we provide Figure 13 which depicts the evaluation timelines of both models based on ResNet-18. Different timeline densities indicate different stages of ResNet-18. We observe that our SNN can achieve high acceleration ratios in the last three stages where the channel numbers are large, which is consistent with our analysis in Sec. 4.5 of the main paper.

References

[1] Sean Bell, C. Lawrence Zitnick, Kavita Bala, and Ross B. Girshick. Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks. In *CVPR*, 2016. 2

[2] Jia Deng, Wei Dong, Richard Socher, Li Jia Li, Kai Li, and Fei Fei Li. Imagenet: a large-scale hierarchical image database. In *CVPR*, 2009. 1

[3] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *IJCV*, 2010. 2

[4] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In *NeurIPS*, 2016. 3

[5] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 1

[6] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *ECCV*, 2014. 2

[7] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. In *ECCV*, 2016. 2

[8] Zechun Liu, Wenhan Luo, Baoyuan Wu, Xin Yang, Wei Liu, and Kwang-Ting Cheng. Bi-real net: Binarizing deep network towards real-network performance. In *IJCV*, 2020. 1, 3

[9] Haotong Qin, Ruihao Gong, Xianglong Liu, Mingzhu Shen, Ziran Wei, Fengwei Yu, and Jingkuan Song. Forward and backward information retention for accurate binary neural networks. In *CVPR*, 2020. 1, 2, 3

[10] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *ECCV*, 2016. 1, 3

[11] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 2

[12] Ziwei Wang, Ziyi Wu, Jiwen Lu, and Jie Zhou. Bidet: An efficient binarized object detector. In *CVPR*, 2020. 1, 2, 3

[13] Dongqing Zhang, Jiaolong Yang, Dongqiangzi Ye, and Gang Hua. Lq-nets: Learned quantization for highly accurate and compact deep neural networks. In *ECCV*, 2018. 1