Bitwidth-Adaptive Quantization-Aware Neural Network Training: A Meta-Learning Approach - Appendix

July 20, 2022

Why does combining meta-learning with QAT work?

Previous approaches (AdaBits/ApDNN) iteratively experience forward and backward propagations with varying bitwidth, which is similar to the inner-loop in MAML, leading to the intuition that combining the process with meta learning can enhance efficiency in a non-few-shot scenario and expand bitwidth-adaptive quantization to the few-shot scenario while also improving performance.

More details on quantization.

We provide Table 6 including every dataset - model architecture pair of our experiments, corresponding bitwidth candidates and (fake) quantization method.

Table 6: Comparison of accuracy (%) between possible PN scheme designs.

Experiment (Dataset, Model architecture)	Omniglot, 5-layer CNN MiniImageNet, 5-layer CNN Omniglot, 4-layer CNN MiniImageNet, 4-layer CNN	CIFAR-10, MobileNet-v2	CIFAR-10, Pre-activation ResNet-20 SVHN, 8-layer CNN
Bitwidth candidates in test (except $b_w \neq b_a$ cases) = Bitwidth candidates in MEBQAT training	(2, 2), (2, 3), (2, 4), (2, 5), (2, 6), (2, 7), (2, 8), (2, 16), (2, FP), (3, 2), (3, 3), (3, 4), (3, 5), (3, 6), (3, 7), (3, 8), (3, 16), (3, FP), (4, 2), (4, 3), (4, 4), (4, 5), (4, 6), (4, 7), (4, 8), (4, 16), (4, FP), (5, 2), (5, 3), (5, 4), (5, 5), (5, 6), (5, 7), (5, 8), (5, 16), (5, FP), (6, 2), (6, 3), (6, 4), (6, 5), (6, 6), (6, 7), (6, 8), (6, 16), (6, FP), (7, 2), (7, 3), (7, 4), (7, 5), (7, 6), (7, 7), (7, 8), (7, 16), (7, FP), (8, 2), (8, 3), (8, 4), (8, 5), (8, 6), (8, 7), (8, 8), (8, 16), (8, FP), (16, 2), (16, 3), (16, 4), (16, 5), (16, 6), (16, 7), (16, 8), (16, 16), (16, FP), (FP, FP)		(1, 1), (1, FP), (2, 2), (2, 3), (2, 4), (2, 5), (2, 6), (2, 7), (2, 8), (2, 16), (2, FP), (3, 2), (3, 3), (3, 4), (3, 5), (3, 6), (3, 7), (3, 8), (3, 16), (3, FP), (4, 2), (4, 3), (4, 4), (4, 5), (4, 6), (4, 7), (4, 8), (4, 16), (4, FP), (5, 2), (5, 3), (5, 4), (5, 5), (5, 6), (5, 7), (5, 8), (5, 16), (5, FP), (6, 2), (6, 3), (6, 4), (6, 5), (6, 6), (6, 7), (6, 8), (6, 16), (6, FP), (7, 2), (7, 3), (7, 4), (7, 5), (7, 6), (7, 7), (7, 8), (7, 16), (7, FP), (8, 2), (8, 3), (8, 4), (8, 5), (8, 6), (8, 7), (8, 8), (8, 16), (8, FP), (16, 2), (16, 3), (16, 4), (16, 5), (16, 6), (16, 7), (16, 8), (16, 16), (16, FP), (FP, FP)
(Fake) quantization method	Learned Step size Quantization (LSQ)	Scale-Adjusted Training (SAT)	DoReFa-Net

Comparison with other possible scheme designs on PN framework.

Table 7: Comparison of accuracy (%) between possible PN scheme designs.

Method	MiniImageNet 5-way 1-shot, 4-layer CNN		MiniImageNet 5-way 5-shot, 4-layer CNN	
	$(b_w, b_a) = (2, 2)$	$(b_w, b_a) = (\mathrm{FP}, \mathrm{FP})$	$(b_w, b_a) = (2, 2)$	$(b_w, b_a) = (\mathrm{FP}, \mathrm{FP})$
No inner-/outer-loop, jointly-varying update	44.61	46.50	59.51	65.52
Jointly-varying inner-loop	46.63	47.24	64.75	66.17
MEBQAT-PN (bitwidth-varying inner-loop, data-varying outer-loop)	47.66	48.33	65.34	66.03

Originally, in a bitwidth-class joint adaptation scenario, PN does not utilize the concept of inner-/outer-loops (as conventional training in a bitwidth adaptation scenario does). Thus, for MEBQAT-PN (third row of Table 7), we create a new inner loop with varying bitwidths (as we do for MEBQAT). Yet, the PN scheme design can be different such as PN model update with jointly-varying bitwidth and class (first row of Table 7) or creating a new jointly-varying inner loop (second row of Table 7). Table 7 shows that our MEBQAT-PN design achieves performance comparable to or outperforming the others.

Experiments on more bitwidth settings.

Comparison of accuracy on more bitwidth settings where $b_w \neq b_a$ is on Table 8. Our proposed scheme has performance comparable to or outperforming the compared schemes, even with a single model.

Table 8: Comparison of accuracy (%) on more bitwidth settings where $b_w \neq b_a$.

Method	CIFAR-10, Pre-activation ResNet-20	SVHN, 8-layer CNN
	$(b_w, b_a) = (7, 5)$	$(b_w, b_a) = (5, 16)$
QAT	92.66 (±0.157)	97.31 (±0.077)
MEBQAT	92.82 (±0.169)	97.64 (±0.051)
Method	Omniglot 20-way 1-shot, 5-layer CNN	Omniglot 20-way 5-shot, 5-layer CNN
	$(b_w, b_a) = (3, 8)$	$(b_w, b_a) = (16, \mathrm{FP})$
FOMAML	78.60	97.47
FOMAML+QAT	66.38	97.46
MEBQAT-MAML	91.86	97.88
Method	MiniImageNet 5-way 1-shot, 5-layer CNN	MiniImageNet 5-way 5-shot, 5-layer CNN
	$(b_w, b_a) = (8, 2)$	$(b_w, b_a) = (2, 4)$
FOMAML	46.23	49.19
FOMAML+QAT	48.61	60.87
MEBQAT-MAML	47.14	62.38
Method	Omniglot 20-way 1-shot, 4-layer CNN	Omniglot 20-way 5-shot, 4-layer CNN
	$(b_w, b_a) = (2, 8)$	$(b_w, b_a) = (8, 3)$
PN	53.15	98.57
PN+QAT	95.73	98.81
MEBQAT-PN	95.12	98.57
Method	MiniImageNet 5-way 1-shot, 4-layer CNN	MiniImageNet 5-way 5-shot, 4-layer CNN
	$(b_w, b_a) = (8, \mathrm{FP})$	$(b_w, b_a) = (2, 3)$
PN	49.47	31.07
PN+QAT	50.01	66.80
MEBQAT-PN	49.68	65.60

Adaptation & inference phase.

Figure 3 illustrates the adaptation-and-inference phase in MEBQAT, MEBQAT-MAML, and MEBQAT-PN.

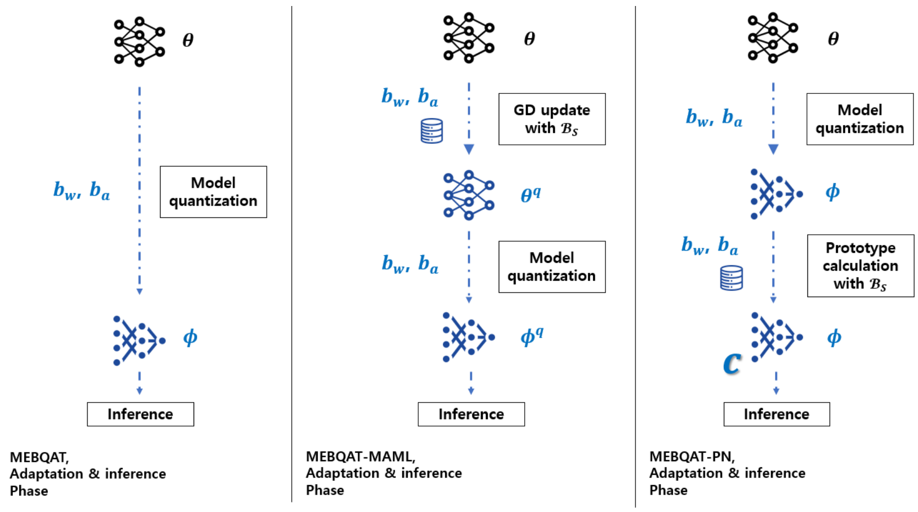

Figure 3: Illustration of adaptation-and-inference phase in MEBQAT, MEBQAT-MAML, and MEBQAT-PN. GD stands for Gradient Descent. **c** denotes prototypes. $\mathcal{B}_S$ represents a support set.