We thank all reviewers for their insightful comments, and have addressed them below.

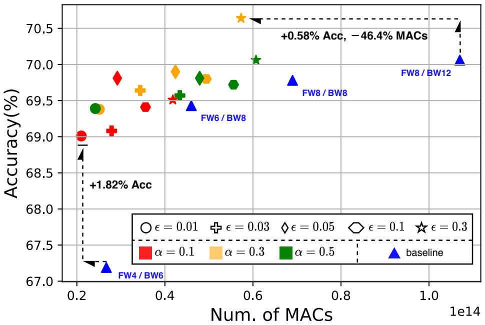
[R1] **FracTrain on larger and deeper models:** Thanks for the advice which helps to strengthen our evaluation. Given the limited time, we apply FracTrain to ResNet-110/ResNet-164 on CIFAR-10/CIFAR-100 and find that again FracTrain consistently outperforms the FW8/BW8 baseline with 38.69%-67.3% computational savings under a lightly higher accuracy (+0.05%- +0.25%).

Method	ResNet-110		ResNet-164	
	CIFAR-10	CIFAR-100	CIFAR-10	CIFAR-100
FW8/BW8	93.38	72.11	93.72	74.55
FracTrain	93.51	72.19	93.77	74.8
Comp. Saving	67.3%	45.17%	38.69%	43.6%

[R1] **MP on BitFusion and Integer on integer hardware:** We will clarify the evaluation metrics in the final version. We used Bit-Fusion for both the MP and integer-only systems to keep the hardware parameters (e.g., dataflows) the same for a fair comparison. We have conducted experiments to address your suggestion by comparing FracTrain on Bit-Fusion and FW8/BW8 on an integer-only hardware Eyeriss [Y. Chen, ISCA'16] based on the simulator in Tetris [M. Gao, ASPLOS'17]: for ResNet-38/74 on CIFAR-100 (accuracy in Table 3), FracTrain on Bit-Fusion still outperforms FW8/BW8 on Eyeriss with +65.8%/+69.8% energy savings and +72.6%/+68.2% latency savings, when both adopting the same unit energy and memory size as in Bit-Fusion for a fair comparison.

[R1] **Bit choices of DFQ:** It is in fact DFQ's advantage to allow adaptive allocation of higher precision to important layers/inputs and lower precision to unimportant ones, and thus enable a larger range of precision choices over static quantization, given the same computational cost. Furthermore, we follow your suggestion, and limit BW in DFQ no more than 8 bits: compared with FW8/BW8 on (1) ResNet-74@CIFAR-10 (93.04%) and (2) ResNet-74@CIFAR-100 (71.01%), DFQ still achieves slightly higher accuracy (+93.11%/71.11%) with +37.3%/+43.7% computational savings.

[R3] **Sensitivity to hyper-params in PFQ:** Figure 2 in the appendix shows PFQ's insensitivity to its precision schedule hyper-params under three different precision schedule strategies. Furthermore, we perform your suggested ablation study to evaluate PFQ-FW(3,4,6,8)/BW(6,6,8,8) on ResNet-38@CIFAR-100 under various ϵ (different shapes) and α (different colors): We can see that a good accuracy-efficiency trade-off can be found in a large range of settings compared with static baselines, showing PFQ's insensitivity to hyper-params. It is intuitive that (1) ϵ and α control the accuracy-efficiency trade-off, and (2) a larger ϵ and α (i.e., faster precision increase) lead to higher training cost and higher accuracy.



[R3] **Modifications on Bit-Fusion:** We did not modify the BitFusion RTL. As the backpropagation can be viewed as two convolution processes (for computing error and gradient, respectively), we estimate energy by executing the three convolution processes of training sequentially in BitFusion. The reuse patterns optimized by BitFusion is output-stationary for both gradients/activations.

[R3, R4] **How MACs are calculated:** Inspired by the computation complexity determined by precision in Sec-2.1 of [30], we calculate the effective MACs of the dot product between a and b using (# of $MACs$)$*Bit_a/32*Bit_b/32$, following [J. Shen, AAAI'20], which is in proportional to bit operations. We will clarify this in the final version.

[R3, R4] **ML accelerators to support FracTrain:** Both dedicated ASIC (e.g., [H. Yoo, ISSCC'19] and [H. Yoo, JICS'20]) or FPGA accelerators (e.g., EDD [Y. Li, DAC'20]) can help exploit FracTrain's best potential by making use of its lower average precision to save both data movement and computation costs during training. After the submission, we have proceeded to implement FracTrain on FPGA to evaluate its real-hardware benefits, following the design in EDD, which adopts a recursive architecture for mix precision networks (i.e., the same computation unit is reused by different precisions) and a dynamic logic to perform dynamic schedule. Evaluated ResNet-38/ResNet-74@CIFAR-100 on Xilinx ZC706 (accuracy in Table 3), FracTrain leads to 34.9%/36.6% savings in latency and 30.3%/24.9% savings in energy compared with FW8/BW8. We will clarify this experiment in the final version.

[R3] **Support by bit-parallel accelerators:** Since the precision granularity in DFQ is block/layer-wise (e.g., a block with several layers in ResNet will use the same precision), bit-parallel is feasible within each block/layer (see our answer and FPGA implementation right above this one).

[R4] **Dataset split:** We use standard train/test datasets (50000 vs. 10000) for CIFAR-10/100 WITHOUT any special split. The training trajectories in Figure 5 visualize the evolution of test accuracy during the whole training process, where the x-axis captures the total computational cost to reach the current epoch instead of the number of epoch. We will clarify this point in the final version.

[R4] **Granularity of the RNN controller:** We use per mini-batch for hardware-friendly run-time quantization in both training and inference. We will clarify this in the final version.

[R1, R3, R4] **Typos and missing references:** Thanks a lot for pointing out! We will ensure that they are addressed and proofread more carefully before camera-ready.