We thank all four reviewers for their valuable feedback. All suggestions will be included in the final draft.

Reviewer1 On TPR hardware: The reviewer's understanding of even/odd phase is correct—but dedicated MAC units are not needed to handle the additional odd phase. FP4 numbers in the odd phase are simply 0.5× of those in the even phase. Thus, the odd-phase final MAC result is also 0.5× and can therefore be efficiently computed by subtracting 1 from the exponent of the final FP16 result using the same MAC engine. On Sec.4.2: Thank you for finding the typo. To clarify, it is the gap between TPR's Var(dL/dx) and FP32's Var(dL/dx) that increases from layer50 to layer1. Our results on FP4 and "FP4+FP8" indicate that quantization causes the loss of gradient variance in Fig.6a. Also in [35], vanishing gradients are reflected by the loss of gradient variance—so we'll look to study this correlation further. On Depthwise Conv (DW): The variance analysis and the hybrid approaches are not limited to Conv1x1 and can be applied to compact networks as well. In fact, while analyzing MobileNetV2 with DW, we found it optimal to use 4b for Conv1x1 and follow the FP8 work recommendations to use FP16 for DW layers[15]. FP16 DW does not impact latency much since it is <3% of total FLOPs and is hard to accelerate due to limited data reuse. For MobilenetV2, we achieve 1.2% degradation on CIFAR10 and 3.0% degradation on ImageNet using FP4. These new accuracy results are especially promising—given that just INT4 inference in MobilenetV2 remains challenging [23,24]. On SOTA models: To our knowledge, this is the first time a SOTA-size model has ever been trained in 4b without major convergence issues. To show the robustness of our approach over a larger model set, we trained a large NLP task with transformer_base model on WMT De-En and obtained a good BLEU score with 2 points off the baseline (25.4 vs 27.5). Overall, we believe this work lays a solid foundation upon which future FP4 innovations can be added on to address the remaining accuracy losses. On hardware area/power: The area projection is based on insights derived from multiple previous AI Hardware and Low-precision FPU designs (published in Symp. on VLSI circuit). In our estimates, a MAC unit that performs 4-way INT4×FP4 inner products consumes 55% of the area of the FP16 FPU while providing 4× throughput, yielding a total compute density improvement of 7.3×. Compared to FP16 FPUs, the 4b unit has simpler shift-based multipliers thanks to the power-of-2 FP4 numbers. It also benefits from the absence of addend aligners, narrower adders, and a simpler normalizer. Additionally, the simpler datapath requires fewer executing stages and saves flip-flop area.

Reviewer2 On 4b overhead: We've estimated that the overheads for 4b training to be <5% of the total GEMM ops. In back-propagation, all collection approaches cost $\mathcal{O}(1)$ FLOPs/gradient element. Compared with HFP8[15] and S2FP8, our FP4 conversion is actually simpler due to the absence of mantissa rounding—and can be done with two back-to-back instructions(1 MUL+1 custom bit OP). The overhead of statistics collection for layer-wise scaling is also comparable to that in S2FP8. Specifically, we expect ~10 FP16 FLOPs/gradient for PACT BWD(2), Radix Conversion(3), Two-phase Rounding(3), and Layer-wise Scaling(2) overheads. These overheads are much smaller than $\mathcal{O}(k_i \times k_j \times channel)$/gradient in convolution GEMMs (e.g. In ResNet50, the effective GEMM FLOPs is 642 per gradient element). Therefore, **with the majority of FLOPs spent on GEMM, 4b training retains significant advantage over HFP8 and S2FP8 training due to the throughput and power & area boost in going from 8b to 4b GEMM**. With additional optimization from our compiler [published in IEEE Micro], 4b ResNet50 training can yield at least 60-80% higher throughput vs. HFP8 training along with a 42% area and power saving. On conversion hardware: The conversion between radix-2 and radix-4 is remarkably simple for FP4. Due to the absence of mantissa bits, this can be done through simple bit operations on exponenst - which rounds down the FP16 exponent to even or odd value (and handles overflow and underflow) through common clipping and rounding operations.

Reviewer3 On Conv1x1: Our mixed precision approach is not limited to only Conv1x1, which is chosen in the context of ResNet50. For compact models like ShuffleNet and MobileNet, Depthwise convolution layers will be in higher precision (see line 8). Although the Conv3x3 layers have 2× FLOPs than the bottom-conv1x1, they yield less accuracy improvements when cast in FP8 (74.7 vs. 75.0). While the performance is determined by FLOPs instead of gradient size, bottom-conv1x1 has 4× output gradients than conv3x3 and top-conv1x1, explaining its efficiency in accuracy gain when cast in FP8. Tradeoff of TPR: Please see line 2. On optimal radix: Radix-2 does not have enough range for gradients. Our experiments with radix-8 on BWD and/or UPDATE GEMM show that training cannot converge due to poor representation. In addition, for hardware implementation, radix-8 cannot adopt TPR to enhance its representation due to the non-integral exponent of dividing 2^3 into two phases.On contribution of format: In the context of radix-4, we propose efficient rounding schemes that minimize the MSE of gradient quantization. In addition, we propose the hybrid use of INT4 for FWD and FP4 for BWD.On channel-level quantization: We limit the quantization granularity to per layer for hardware efficiency. If each output-channel has a different scale, reduction over the channel dimension will be hardware inefficient. On cancellation: Fig.4a shows that the same gradient will most likely be quantized towards opposite directions in each phase. This is important to get a lower expected quantization error $\mathbb{E}(Q_{err})$, which is the sum of mean errors over the entire ensemble from each phase **divided by 2**. The cancellation contributes to a lower average expected value but individual values could certainly be skewed higher or lower.

Reviewer4 On empirical validation: We agree with the reviewer and would like to point that we have also evaluated other models (please refer to lines 12&16). In future, we plan to utilize AutoML techniques to make FP4 broadly applicable. On bias: The numerical bias caused by quantization is random and not expected to result in fairness issues — especially since it acts at lower levels than the data/target/model choices and is therefore not directly susceptible to human intervention.We do plan to validate this further as part of future work.