Training with Fewer Bits: Unlocking Edge LLMs Training with Stochastic Rounding

Taowen Liu, Marta Andronic, Deniz Gündüz, George A. Constantinides
Department of Electrical and Electronic Engineering
Imperial College London
London, UK
{tony.liu20, marta.andronic18, d.gunduz, g.constantinides}@imperial.ac.uk

Abstract

LLM training is resource-intensive. Quantized training improves computational and memory efficiency but introduces quantization noise, which can hinder convergence and degrade model accuracy. Stochastic Rounding (SR) has emerged as a theoretically attractive alternative to deterministic rounding, offering unbiased gradient estimates. However, its interaction with other training factors—especially batch size—remains underexplored. In this paper, we present a theoretical and empirical study of mini-batch stochastic gradient descent (SGD) with SR, showing that **increased batch sizes can compensate for reduced precision during backpropagation**. Furthermore, we show that quantizing weights and activations impacts gradient variance in distinct ways. Our experiments validate these theoretical insights.

1 Introduction

Training large language models (LLMs) demands significant computational and memory resources. Mixed-precision training, using lower-precision formats, offers a crucial path to efficiency (Micikevicius et al., 2017; Das et al., 2018; Wu et al., 2018; Lee et al.), enabling training on edge devices (Kandala and Varshney, 2024). However, reduced precision introduces quantization noise, potentially hindering convergence. Stochastic rounding (SR) (Croci et al., 2022) provides an attractive, unbiased alternative to deterministic rounding methods for managing this noise. As shown in Figure 1, SR can maintain training stability and performance where round-to-nearest (RTN) fails, especially under aggressive quantization.

Despite SR's advantages, its practical interaction with mini-batch stochastic gradient descent (SGD), particularly the role of batch size in mitigating SR-induced noise, remains underexplored. Key questions persist: Can larger batch sizes—a common variance reduction tool—effectively counteract SR noise? How does this apply distinctly to quantizing shared model weights versus per-sample activations and gradients? And crucially, how does SR's variance impact SGD convergence guarantees? Addressing these is vital for establishing theoretically grounded guidelines for optimal precision, rounding, and batch size selection, especially to unlock aggressive quantization for training LLMs on edge devices.

Although prior work has explored mixed-precision training empirically (Micikevicius et al., 2017; Blake et al., 2023; Peng et al., 2023) and theoretically analyzed aspects like gradient quantization for efficient communication (Xia et al., 2021, 2022, 2025, 2024) or weight quantization (Li et al., 2017), a rigorous understanding of low-precision arithmetic within the backpropagation process itself, combined with the specific variance characteristics introduced by SR in a mini-batch context, remains limited. Chen et al. (2020) analyzed the impact of quantization error but did not explicitly incorporate batch size as a variable interacting with different sources of SR noise.

This paper addresses this gap through a theoretical and empirical analysis of SR within mini-batch SGD. We explicitly model and differentiate the statistical properties of noise from quantizing shared weights versus per-sample activations and gradients during backpropagation. Our core theoretical finding, supported by empirical validation, is that **increased batch sizes can effectively compensate for reduced precision during backpropagation by mitigating SR-induced variance from per-sample operations.** Specifically, we demonstrate that reducing precision in activation and gradient quantization can be offset by a quantifiable increase in batch size—for instance, a 1-bit reduction may be balanced by at most a fourfold batch increase to maintain convergence quality, with practical increases often being milder.

Our main contributions are:

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 14531–14546

- We develop a comprehensive theoretical analysis that explicitly models the impact of mini-batch size on SGD convergence when employing SR for quantizing different components in the training pipeline. This framework distinguishes between noise originating from weight quantization and noise from per-sample activation and gradient quantization.
- Leveraging our framework, we prove that the variance introduced by stochastically rounding per-sample activations and their gradients during backpropagation (specifically, for computing weight gradients like $\widehat{A_{in}^T}\widehat{\nabla A_{out}}$ and input activation gradients like $\widehat{\nabla A_{out}}\widehat{W^T}$) decays inversely with the mini-batch size ($1/b$). This provides a theoretical underpinning for using larger batches to counteract reduced precision in these operations.
- We conduct experiments on both image classification and LLM fine-tuning to validate our theoretical predictions. These experiments demonstrate SR's superiority over RTN and confirm the predicted batch size scaling effect. From these results, we derive practical guidelines quantifying the trade-off, e.g., showing that a 1-bit precision reduction in activation/gradient quantization can be compensated by approximately doubling the batch size in practice to maintain similar convergence behavior.

2 Related Works

2.1 Mixed-Precision Training

Mixed-precision training has emerged as a crucial approach for efficient LLMs training by quantizing weights, activations, and gradients. Typically, general matrix-matrix multiplication (GEMM) operations are performed in mixed-precision during gradient computation. Early research demonstrated the viability of training with reduced precision using `FP16` (Gupta et al., 2015; Micikevicius et al., 2017) and `INT16` (Das et al., 2018). The field progressed with the development of `FP8`-based methods (Wang et al., 2018b; Banner et al., 2018; Yang et al., 2020) and even 4-bit methods (Sun et al., 2020; Chmiel et al., 2021). Various techniques have been proposed to address the challenges of mixed-precision training, including loss scaling (Micikevicius et al., 2017), precision-aware parameter initialization (Blake et al., 2023), and blockwise quantization (Peng et al., 2023).

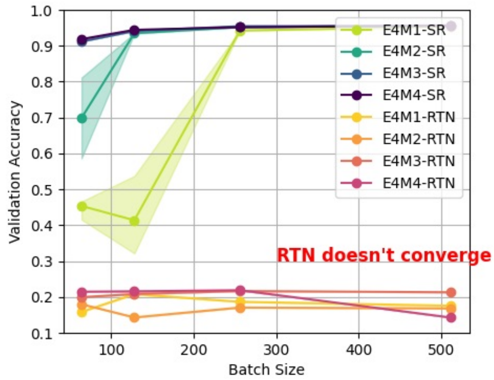


Figure 1: Stochastic rounding (SR) achieves higher accuracy with 8y batch sizes, while round-to-nearest (RTN) fails to converge at the same precision. [1]

2.2 Quantization-Aware Training

Quantization-aware training (QAT) represents another significant direction, focusing on learning quantized weights during training (Hubara et al., 2018). Researchers have explored diverse approaches, including uniform and non-uniform quantization methods (Zhou et al., 2017), mixed-precision quantization (Dong et al., 2019), and learned quantization strategies (Esser et al., 2019).

2.2.1 Stochastic Rounding

Stochastic rounding (SR) (Croci et al., 2022; Xia et al., 2021, 2022, 2025, 2024) is a probabilistic technique used to mitigate the errors introduced by quantization, especially in low-precision arithmetic by providing unbiased estimates of quantized values. Most theoretical analyses of mixed-precision training have focused on gradient quantization for communication efficiency or storage efficiency (Xia et al., 2021, 2022, 2025, 2024). Chen et al. (2020) proposed a framework for analyzing the impact of gradient quantization on convergence properties. SR randomly rounds the value to one of the two nearest levels, with probabilities proportional to the distance of the value from each level. SR can be efficiently implemented in hardware using a pseudo-random number generator such as a linear feedback shift register (LFSR). SR has been adopted in modern hardware implementations such as Graphcore processors (Knowles, 2021). To define SR formally, we first define the threshold quantization function, which can express both round-to-nearest and stochastic rounding.

Definition 1 (**Threshold Quantization Function**).

[1] Validation accuracy of WideResNet-16 on CIFAR-10.

Let x be the number to be quantized, let Δ be the precision level, and let ϵ be the threshold. The threshold quantization function is defined as:

$$\mathbb{Q}_\Delta(x, \epsilon) = \Delta \cdot \begin{cases} \lfloor \frac{x}{\Delta} \rfloor, \textit{when } \frac{x}{\Delta} - \lfloor \frac{x}{\Delta} \rfloor < \epsilon \\ \lfloor \frac{x}{\Delta} + 1 \rfloor, \textit{otherwise} \end{cases} \tag{1}$$

Note: Specifically, when $\Delta = 0$, for any ϵ, define $\mathbb{Q}_0(x, \epsilon) = x$.

Definition 2 (**Round-to-nearest (RTN)**). *Given a real number x and a quantization step size Δ, the RTN quantization of x is defined as:*

$$\hat{x} = \mathbb{Q}_\Delta(x, \frac{1}{2}), \tag{2}$$

where rounding is performed towards positive infinity in the event of a tie.

Definition 3 (**Stochastic Rounding (SR)**). *Given a real number x, a quantization step size Δ, and a randomly sampled threshold $\epsilon \sim U[0, 1]$, the SR quantization of x is defined as:*

$$\hat{x} = \mathbb{Q}_\Delta(x, \epsilon). \tag{3}$$

A matrix stochastic rounding is an element-wise stochastic rounding with the thresholds drawn independently for each element.

SR provides an unbiased estimator of the true arithmetic result, though this accuracy comes at the cost of increased variance compared to deterministic methods. For example, consider summing the value 0.7 exactly ten times using integer arithmetic. Deterministic RTN consistently rounds 0.7 up to 1, yielding a fixed sum of 10. This deterministic approach introduces a bias of $+3$, as the exact sum should be 7, but it exhibits zero variance. In contrast, SR probabilistically rounds 0.7 either up to 1 with a 70% probability or down to 0 with a 30% probability. Consequently, repeated summations using SR yield varying outcomes such as 6, 7, or 8. Crucially, the expected sum over many trials converges precisely to the unbiased value of 7.

2.2.2 Mixed-Precision Matrix Multiplication

In LLMs training, the most arithmetic-intensive operation is matrix multiplication in linear or convolution layers. In accelerators like GPUs, the mixed-precision multiplication performs the scalar multiplication operations in low precision and the accumulation in high precision. Mathematically, this is equivalent to full-precision matrix multiplication of two quantized matrices. $\hat{A} = \mathbb{Q}_\Delta(A, \epsilon_A), \hat{B} = \mathbb{Q}_\Delta(B, \epsilon_B)$. Therefore, AB is approximated through $\hat{A}\hat{B}$.

3 A General Framework for Mixed-Precision SGD

In mixed-precision SGD, matrix multiplications are performed using lower precision. This involves quantizing the layer's input activations, its weight matrix, and the gradients backpropagated from the subsequent layer before they are used in the forward and backward pass matrix multiplications. The choice of quantization parameters (precision levels Δ and rounding thresholds ϵ) for each of these affects training dynamics.

Our analysis focuses on the quantization within linear layers (including convolutional layers viewed as matrix multiplications), as they typically dominate the computational cost (Joshi et al., 2020). Consider a network with n linear layers, indexed $i = 1, \dots, n$. The weights for linear layers are $\boldsymbol{w} = W_1, \dots, W_n$. The input activation to layer i is A_{i-1}, the weight is W_i, and the output activation is A_i. x and y are the network input and label, respectively. For simplicity, we exclude the bias. We denote ∇A_i as the gradient of the loss L with respect to A_i. We also denote $f(\boldsymbol{w}) = \mathbb{E}_{x,y}[L(\boldsymbol{w}, x, y)]$.

For each layer, 5 quantization operations are involved. The exact quantization operation behavior is exactly defined through quantization thresholds ϵ and quantization precision Δ. We define $\Delta_i = \{\Delta_A^{\text{fwd}}, \Delta_W^{\text{fwd}}, \Delta_A^{\text{bwd}}, \Delta_W^{\text{bwd}}, \Delta_{\nabla A}^{\text{bwd}}\}$ and $\epsilon_i = \{\epsilon_A^{\text{fwd}}, \epsilon_W^{\text{fwd}}, \epsilon_A^{\text{bwd}}, \epsilon_W^{\text{bwd}}, \epsilon_{\nabla A}^{\text{bwd}}\}$. Let $\boldsymbol{\Delta} = \{\Delta_i\}_{i=1}^n$ and $\boldsymbol{\epsilon} = \{\epsilon_i\}_{i=1}^n$ represent the collection of all quantization parameters for the entire network.

Definition 4 (**Gradient Approximation**). *The gradient approximation $g(\boldsymbol{w}, x, y, \boldsymbol{\Delta}, \boldsymbol{\epsilon})$ for a sample (x, y) is computed via the forward and backward passes described in Algorithms 1,2 and Figure 2.*

3.1 Mixed-Precision SGD with Deterministic Rounding

One approach is to use round-to-nearest (RTN) rounding. This corresponds to setting all quantization thresholds $\boldsymbol{\epsilon}$ to 0.5. The approximate gradient is then computed as $g(\boldsymbol{w}, x, y, \boldsymbol{\Delta}, \boldsymbol{\epsilon}_{0.5})$, where $\boldsymbol{\Delta}$ contains the desired low-precision levels. A drawback of this approach is that deterministic rounding introduces systematic bias. Consequently, the resulting approximate gradient $g(\boldsymbol{w}, x, y, \boldsymbol{\Delta}, \boldsymbol{\epsilon}_{0.5})$ is usually a *biased* estimator of the true gradient

Algorithm 1 Mixed-Precision Forward Pass

function F($A_{in}, W, \Delta_A^{\text{fwd}}, \epsilon_A^{\text{fwd}}, \Delta_W^{\text{fwd}}, \epsilon_W^{\text{fwd}}$)
  $\widehat{A}_{in} \leftarrow \mathbb{Q}_{\Delta_A^{\text{fwd}}}(A_{in}, \epsilon_A^{\text{fwd}})$
  $\widehat{W} \leftarrow \mathbb{Q}_{\Delta_W^{\text{fwd}}}(W, \epsilon_W^{\text{fwd}})$
  $A_{out} \leftarrow \widehat{A}_{in}\widehat{W}$
  return A_{out}
end function

Note: Forward and backward passes are deterministic for fixed $\boldsymbol{\epsilon}$. Stochasticity can arise if thresholds ϵ are sampled, otherwise round-to-nearest is applied.

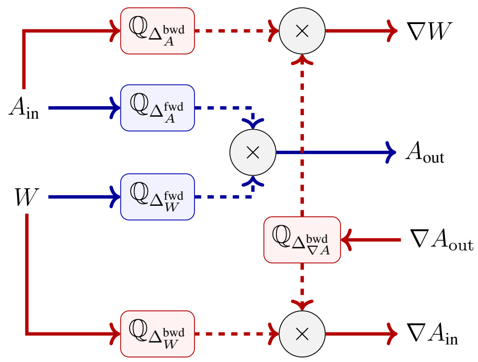


Figure 2: The forward pass is in blue, and the backward pass in red. Solid arrows represent data flow, while dashed arrows indicate the flow of quantized values.

$\nabla_{\boldsymbol{w}} L(\boldsymbol{w}, x, y)$, unless extreme cases, for example, the intermediate values are exactly lying on the quantization points. This means that even when averaged over the data distribution, the expected approximate gradient may not equal the true expected gradient:

$$\mathbb{E}_{x,y}\left[g(\boldsymbol{w}, x, y, \boldsymbol{\Delta}, \boldsymbol{\epsilon}_{0.5})\right] \neq \mathbb{E}_{x,y}\left[\nabla_{\boldsymbol{w}} L(\boldsymbol{w}, x, y)\right]. \tag{4}$$

This lack of unbiasedness complicates theoretical convergence analysis, as many standard SGD proofs rely on the assumption of an unbiased gradient estimator.

3.2 Weight-Only QAT

In a weight-only QAT setup (Jacob et al., 2018), the weight is quantized using the same threshold and precision in both forward and backward passes.

In our framework $g(\boldsymbol{w}, x, y, \boldsymbol{\Delta}, \boldsymbol{\epsilon})$, this strategy corresponds to the following settings: quantize weight $\widehat{\boldsymbol{w}} = \widehat{\boldsymbol{w}}^{\text{fwd}} = \widehat{\boldsymbol{w}}^{\text{bwd}} = \mathbb{Q}_{\Delta \boldsymbol{w}}(\boldsymbol{w}, \epsilon_{\boldsymbol{w}})$ and keep the activations/gradients in high precision $\Delta_A^{\text{fwd}} = \Delta_A^{\text{bwd}} = \Delta_{\nabla A}^{\text{bwd}} = 0$. While the resulting solution may differ from that obtained with full-precision SGD, it is known that SGD for weight-only QAT can converge near a stationary point (Li et al., 2017). The quantization threshold can either be selected deterministically or stochastically.

Algorithm 2 Mixed-Precision Backward Pass

function B($A_{in}, W, \nabla A_{out}, \Delta_A^{\text{bwd}}, \epsilon_A^{\text{bwd}}, \ldots$)
  $\widehat{A}_{in} \leftarrow \mathbb{Q}_{\Delta_A^{\text{bwd}}}(A_{in}, \epsilon_A^{\text{bwd}})$
  $\widehat{W} \leftarrow \mathbb{Q}_{\Delta_W^{\text{bwd}}}(W, \epsilon_W^{\text{bwd}})$
  $\widehat{\nabla}_{A_{out}} \leftarrow \mathbb{Q}_{\Delta_{\nabla A}^{\text{bwd}}}(\nabla A_{out}, \epsilon_{\nabla A}^{\text{bwd}})$
  $\nabla A_{in} \leftarrow \widehat{\nabla}_{A_{out}} \widehat{W}^T$
  $\nabla W \leftarrow \widehat{A}_{in}^T \widehat{\nabla}_{A_{out}}$
  return $\nabla A_{in}, \nabla W$
end function

Mathematically, the gradient approximation g with this setup is equivalent to evaluating the loss with quantized weights, and then computing gradients with respect to the quantized weights.

$$g(\boldsymbol{w}, x, y, \boldsymbol{\Delta}, \boldsymbol{\epsilon}) = \nabla_{\widehat{\boldsymbol{w}}_t} L(\widehat{\boldsymbol{w}}_t, x, y) \tag{5}$$

It is also worth noting that only weights are quantized; the potential for hardware acceleration from mixed-precision units is limited. This motivates our exploration of stochastic rounding for other components in the following section.

3.3 Stochastic Rounding Mixed-Precision SGD for QAT Objectives

Our goal is to compute an estimate $g(\boldsymbol{w}, x, y, \boldsymbol{\Delta}, \boldsymbol{\epsilon})$ of the QAT gradient $\nabla_{\widehat{\boldsymbol{w}}} L(\widehat{\boldsymbol{w}}, x, y)$ such that the estimate is unbiased and performing computation in low precision. SR can be employed in the backward pass (Chen et al., 2020).

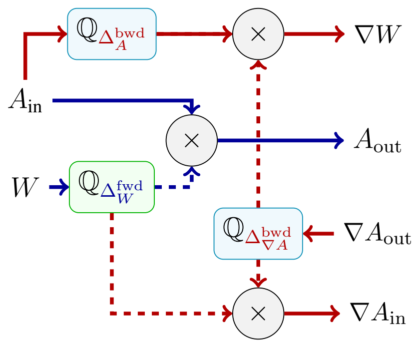


Figure 3: Stochastic Rounding Mixed-Precision SGD for QAT Objectives: Weight quantization is shared across the forward and backward passes. Activation is kept with high precision in the forward pass.

To achieve this, weights in both forward and backward pass, as well as activations/gradients in backward pass, are quantized in low precision $\Delta_W^{\text{fwd}} = \Delta_W^{\text{bwd}} = \Delta_A^{\text{bwd}} = \Delta_{\nabla A}^{\text{bwd}} = \Delta$. Forward pass activation remains high precision to ensure convergence $\Delta_A^{\text{fwd}} = 0$. Activations/gradients are stochastically quantized; ϵ_A^{bwd} and $\epsilon_{\nabla A}^{\text{bwd}}$ are sampled from $U[0, 1]$. Weight quantization threshold

$\epsilon_{\boldsymbol{w}}$ can be either stochastic or deterministic, which is discussed later in section 4.1. Denote $(\boldsymbol{\Delta}^*, \boldsymbol{\epsilon}^*)$ to be the quantization parameters. This approach is captured by figure 3. Under these conditions, the resulting approximate gradient $g(\boldsymbol{w}, x, y, \boldsymbol{\Delta}^*, \boldsymbol{\epsilon}^*)$ is an unbiased estimator of $\nabla_{\widehat{\boldsymbol{w}}} L(\widehat{\boldsymbol{w}}, x, y)$:

$$\mathbb{E}_{\epsilon_A^{\text{bwd}}, \epsilon_{\nabla A}^{\text{bwd}}} \left[g(\boldsymbol{w}, x, y, \boldsymbol{\Delta}^*, \boldsymbol{\epsilon}^*) \right] = \nabla_{\widehat{\boldsymbol{w}}} L(\widehat{\boldsymbol{w}}, x, y). \tag{6}$$

Consequently, when taking expectation over the data distribution, the expected approximate gradient equals the expected QAT gradient:

$$\begin{aligned} &\mathbb{E}_{x,y,\epsilon_A^{\text{bwd}}, \epsilon_{\nabla A}^{\text{bwd}}} \left[g(\boldsymbol{w}, x, y, \boldsymbol{\Delta}^*, \boldsymbol{\epsilon}^*) \right] \\ =&\mathbb{E}_{x,y} \left[\nabla_{\widehat{\boldsymbol{w}}} L(\widehat{\boldsymbol{w}}, x, y) \right]. \end{aligned} \tag{7}$$

For mini-batch mixed-precision SGD, it corresponds to randomly sampling ϵ_A^{bwd} and $\epsilon_{\nabla A}^{\text{bwd}}$ for each sample j. However, the same copy of weight is used across a mini-batch, whether it is stochastically or deterministically quantized. We introduce the following simplified notation:

Definition 5 (Stochastic Rounding Mini-batch Mixed-precision SGD). *With a small abuse of notation, define a simplified notation for Stochastic Rounding Mini-batch Mixed-precision SGD*

$$\begin{aligned} \tilde{G}(\boldsymbol{w}) = \frac{1}{b} \sum_{j=1}^{b} g(\boldsymbol{w}, x^j, y^j, \boldsymbol{\Delta}^*, \\ \{\epsilon_W^{\text{fwd}}, \epsilon_W^{\text{bwd}}, \epsilon_A^{\text{bwd}\boldsymbol{j}}, \epsilon_{\nabla A}^{\text{bwd}\boldsymbol{j}}\}). \end{aligned} \tag{8}$$

To emphasize, ϵ_W^{fwd} and ϵ_W^{bwd} do not have a superscript j, whereas $\epsilon_A^{\text{bwd}\boldsymbol{j}}$ and $\epsilon_{\nabla A}^{\text{bwd}\boldsymbol{j}}$ have superscript j. For each mini-batch, the estimator $\tilde{G}(\boldsymbol{w})$ remains an unbiased estimator of the mini-batch weight only QAT gradient:

$$\begin{aligned} &\mathbb{E}_{(x,y)\in\text{batch}, \epsilon_A^{\text{bwd}}, \epsilon_{\nabla A}^{\text{bwd}}} [\tilde{G}(\boldsymbol{w}_t)] \\ =&\mathbb{E}_{(x,y)} [\nabla_{\widehat{\boldsymbol{w}}_t} L(\widehat{\boldsymbol{w}}_t, x, y)] = \nabla_{\widehat{\boldsymbol{w}}} f(\widehat{\boldsymbol{w}}). \end{aligned} \tag{9}$$

4 Convergence Analysis

This section presents a theoretical analysis of the convergence properties of mini-batch mixed-precision SGD in a non-convex setting. We aim to understand how weight quantization and activation/gradient quantization interact with the mini-batch size and hence affect convergence.

The SGD update rule is given by:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_t \tilde{G}(\boldsymbol{w}_t), \tag{10}$$

where η_t is the learning rate at iteration t, and $\tilde{G}(\boldsymbol{w}_t)$ is the mini-batch gradient estimator computed using low-precision arithmetic.

One aspect of our analysis is the nature of this estimator. We denote $\mathbb{E}_t[\cdot] = \mathbb{E}[\cdot|\boldsymbol{w}_t]$ as the expectation conditional on $\boldsymbol{w}_t$. As stated in Equation 9, $\mathbb{E}_t[\tilde{G}(\boldsymbol{w}_t)] = \nabla_{\widehat{\boldsymbol{w}}_t} f(\widehat{\boldsymbol{w}}_t)$, meaning $\tilde{G}(\boldsymbol{w}_t)$ is an unbiased gradient estimator of the QAT objective $f(\widehat{\boldsymbol{w}})$. However, since $\nabla_{\widehat{\boldsymbol{w}}_t} f(\widehat{\boldsymbol{w}}_t)$ may differ from the true gradient $\nabla f(\boldsymbol{w}_t)$, our SGD algorithm operates with a potentially biased estimate of the gradient of our ultimate objective.

Another crucial aspect of our analysis involves quantifying the total variance of this estimator $\tilde{G}(\boldsymbol{w}_t)$ and understanding how its components behave, particularly in relation to the mini-batch size b. This variance arises from two primary sources: sampling variance as well as variance introduced by stochastic rounding. Our analysis assumes that quantization errors are i.i.d. per-sample. This is a reasonable assumption in our framework, as stochastic rounding is performed independently on each number before multiplication, and accumulation is performed in high precision. A key insight is that SR variance, similar to sampling variance, diminishes as the mini-batch size b increases. Intuitively, because SR is applied independently to each of the b samples' activations/gradients, the errors introduced by these quantization steps tend to average out across the mini-batch. Lemma 3 provides a concrete derivation of this $1/b$ scaling.

The interplay between the bias and the two sources of variance (sampling and per-sample SR, both influenced by batch size) is the core of our convergence analysis. We operate on assumptions:

Assumption 1 (Smoothness and Boundedness). *The true loss function $L(\boldsymbol{w}, x, y)$ is $\mathcal{L}$-smooth, meaning its gradient is $\mathcal{L}$-Lipschitz continuous: $\|\nabla L(\boldsymbol{w}, x, y) - \nabla L(\mathbf{v}, x, y)\| \leq \mathcal{L}\|\boldsymbol{w} - \mathbf{v}\|$ for all $\boldsymbol{w}, \mathbf{v}$. Furthermore, $L(\boldsymbol{w}, x, y)$ is bounded below by $L_{\min}$, i.e., $L(\boldsymbol{w}, x, y) \geq L_{\min}$.*

4.1 Bias from weight quantization

The use of quantized weights $\widehat{\boldsymbol{w}}$ instead of full-precision weights $\boldsymbol{w}$ when defining the target gradient introduces a systematic bias. The following lemma bounds this bias.

Lemma 1 (Bounded Gradient Bias from Weight Quantization). *Let $\widehat{\boldsymbol{w}} = \mathbb{Q}_{\Delta W}(\boldsymbol{w}, \epsilon_W)$ with quantization step ΔW. The difference between the QAT gradient $\nabla_{\widehat{\boldsymbol{w}}} L(\widehat{\boldsymbol{w}}, x, y)$ and the true gradi-*

ent $\nabla L(\boldsymbol{w}, x, y)$ is uniformly bounded:

$$\|\nabla_{\widehat{\boldsymbol{w}}} L(\widehat{\boldsymbol{w}}, x, y) - \nabla L(\boldsymbol{w}, x, y)\| \leq B_W, \quad (11)$$

where $B_W = \frac{1}{2}\mathcal{L}\sqrt{d}\Delta_W$ with RTN and $B_W = \mathcal{L}\sqrt{d}\Delta_W$ with SR.

Proof. The proof relies on the L-smoothness of $f(\boldsymbol{w})$ and the bound on element-wise quantization error, $\|\widehat{\boldsymbol{w}} - \boldsymbol{w}\| \propto \sqrt{d}\Delta_W$. A detailed derivation is provided in Appendix A. □

Lemma 1 establishes that the gradient bias B_W is proportional to the weight quantization precision Δ_W. This bias term B_W will contribute to an error floor in our final convergence bound that is **not reducible by increasing the mini-batch size** b.

4.2 Variance Reduction via Mini-Batching for Per-Sample Quantization

In addition to the bias, the gradient estimator $\tilde{G}(\mathbf{w}_t)$ is subject to variance. This variance stems from both the stochastic sampling of data and the stochastic rounding. In this section, we focus on the computation $\nabla W = A^T A^{\text{out}}$ since this would provide insight about why the variance of gradient estimator would decay by $\frac{1}{b}$, whereas the variance of $\nabla A = A^{\text{out}} W^T$ is independent from batch size.

Consider the computation of a gradient component ∇W_{ij} for a weight matrix. Let $A \in \mathbb{R}^{D\times h_1}$ and $A^{\text{out}} \in \mathbb{R}^{D\times h_2}$ be the full-dataset activations and upstream gradients, respectively. The (i, j)-th component of the true full-batch gradient is:

$$\nabla W_{ij}^{\text{(full)}} = \frac{1}{D}\sum_{k=1}^{D} A_{ki} A_{kj}^{\text{out}}. \quad (12)$$

For a mini-batch of b samples, the corresponding component of the quantized gradient estimate is:

$$\widehat{\nabla} W_{ij}^{\text{(quant-mini)}} = \frac{1}{b}\sum_{k\in batch} \mathbb{Q}_{\Delta_A}(A_{ki}, \epsilon_{A_{ki}})\, \mathbb{Q}_{\Delta_{A^{\text{out}}}}(A_{kj}^{\text{out}}, \epsilon_{A_{kj}^{\text{out}}}), \quad (13)$$

where $\epsilon_{A_{ki}}$ and $\epsilon_{A_{kj}^{\text{out}}}$ are per-sample random thresholds for stochastic rounding.

The following lemma characterizes the Mean Squared Error (MSE) of this quantized mini-batch estimate.

Lemma 2 (**Error Decomposition for Fully Quantized Gradient Component**). *Under assumption of stochastic rounding, the MSE of $\widehat{\nabla} W_{ij}^{\textit{(quant-mini)}}$ with respect to $\nabla W_{ij}^{\textit{(full)}}$ is:*

$$\mathbb{E}\left[\left(\nabla W_{ij}^{\textit{(full)}} - \widehat{\nabla} W_{ij}^{\textit{(quant-mini)}}\right)^2\right] = T_{ij}^{s} + T_{ij}^{\mathbb{Q}}, \quad (14)$$

where

$$T_{ij}^{\text{s}} = \mathbb{E}\Big[(\nabla W_{ij}^{\textit{(full)}} - \tfrac{1}{b}\sum_{k\in batch} A_{ki} A_{kj}^{\text{out}})^2\Big]$$
$$T_{ij}^{\mathbb{Q}} = \mathbb{E}\Big[\tfrac{1}{b}\sum_{k\in batch} (A_{ki}\nabla A_{kj}^{\text{out}} - \mathbb{Q}(A_{ki})\,\mathbb{Q}(\nabla A_{kj}^{\text{out}}))^2\Big]. \quad (15)$$

The decomposition holds because the cross-term is zero due to the unbiased nature of stochastic rounding when conditioned on the mini-batch data. (See Appendix B for details). Standard SGD tells us that the sampling error T_{ij}^{s} can be reduced by increasing the batch size b. We now show that the quantization error $T_{ij}^{\mathbb{Q}}$ also benefits from larger batch sizes.

Lemma 3 (**Quantization Error under Per-Sample SR Scaling and Batch Size**). *Let $T_{ij}^{\mathbb{Q}}$ be the quantization error term defined in Lemma 2. Assuming i.i.d. samples within each mini-batch and independent stochastic rounding for every element,*

$$T_{ij}^{\mathbb{Q}} \leq \frac{1}{b}\left(\mathbb{E}[A_{ki}^2]\sigma_{A^{\text{out}}}^2 + \mathbb{E}[(A_{kj}^{\text{out}})^2]\sigma_A^2 + \sigma_A^2\sigma_{A^{\text{out}}}^2\right) \propto \frac{C}{b}\cdot 2^{-2B}. \quad (16)$$

Here σ_A^2 and $\sigma_{A^{\text{out}}}^2$ denote the quantization error variances of A and A^{out}. For step size Δ_X, we have $\sigma_X^2 \leq \Delta_X^2$. With B mantissa bits, $\Delta \propto 2^{-B}$, leading to $\sigma_X^2 \propto 2^{-2B}$. The constant C depends on the second moments of A and A^{out}. A detailed proof is provided in Appendix C.

Lemma 3 demonstrates that the SR variance component $T_{ij}^{\mathbb{Q}}$ **decays inversely with the mini-batch size** b. This insight is crucial for understanding the trade-off depicted in Figure 4. Specifically, if reducing precision by 1 bit causes the 2^{-2B} factor to increase by 4 times, this increase can be counteracted by increasing the batch size b by a factor of 4 to maintain the same level of $T_{ij}^{\mathbb{Q}}$. This provides a theoretical basis for the empirical observation that larger batch sizes can compensate for reduced precision when using SR for activations and gradients.

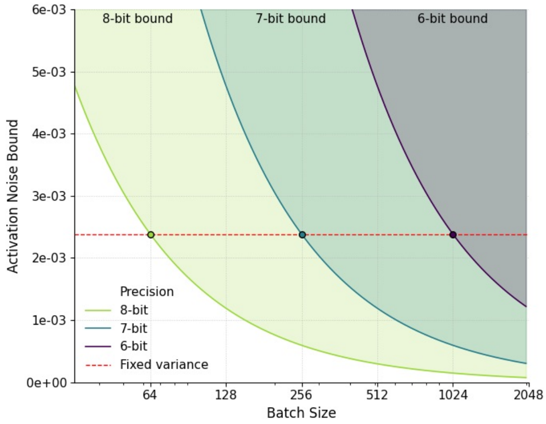


Figure 4: To guarantee the same variance, increase batch size by at most $4\times$ when reducing 1 bit of precision.

4.3 Convergence Theorem

Assumption 2 (Bounded Gradient Variance Components).

Let $\widehat{G}(\boldsymbol{w}_t) = \mathbb{E}_{(x,y)}[\nabla_{\widehat{\boldsymbol{w}}_t} L(\widehat{\boldsymbol{w}}_t, x, y)]$ be the expected QAT gradient. We assume the following for the stochastic gradient estimator $\tilde{G}(\boldsymbol{w}_t)$:

Sampling Variance: *The variance of the true QAT gradients is bounded by σ_S^2:*

$$\mathbb{E}_{(x,y)} \|\nabla_{\widehat{\boldsymbol{w}}_t} L(\widehat{\boldsymbol{w}}_t, x, y) - \widehat{G}(\boldsymbol{w}_t)\|^2 \le \sigma_S^2. \quad (17)$$

Quantization Noise Variance: *The expected variance from stochastic rounding for a single sample, when estimating $\nabla_{\widehat{\boldsymbol{w}}_t} L(\widehat{\boldsymbol{w}}_t, x, y)$ with $g(\boldsymbol{w}_t, x, y, \boldsymbol{\Delta}^*, \boldsymbol{\epsilon}^*)$, is bounded by σ_Q^2:*

$$\mathbb{E}_{x,y,\boldsymbol{\epsilon}^*}\left[\|g(\boldsymbol{w}_t, x, y, \boldsymbol{\Delta}^*, \boldsymbol{\epsilon}^*) - \nabla_{\widehat{\boldsymbol{w}}_t} L(\widehat{\boldsymbol{w}}_t, x, y)\|^2\right] \le \sigma_Q^2. \quad (18)$$

The variance of the mini-batch gradient estimator $\tilde{G}(\boldsymbol{w}_t)$ around $\widehat{G}(\boldsymbol{w}_t)$ is bounded by:

$$\mathbb{E}_t[\|\tilde{G}(\boldsymbol{w}_t) - \widehat{G}(\boldsymbol{w}_t)\|^2] \le \frac{\sigma_S^2 + \sigma_Q^2}{b},$$

where b is the mini-batch size.

With the above setup and assumptions, we can state the convergence properties of the mixed-precision SGD algorithm.

Theorem 1 (Convergence of SGD with Low-Precision Gradients). *Under Assumptions 1 and 2, running low-precision SGD with a constant learning rate $\eta \le 1/4\mathcal{L}$ for T iterations, the average squared norm of the true gradient is bounded by:*

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}[\|\nabla L(\boldsymbol{w}_t)\|^2] \le \frac{4(L(\boldsymbol{w}_0) - L_{\min})}{\eta T} + C_B B_W^2 + C_V \eta \mathcal{L}\left(\frac{\sigma_S^2 + \sigma_Q^2}{b}\right), \quad (19)$$

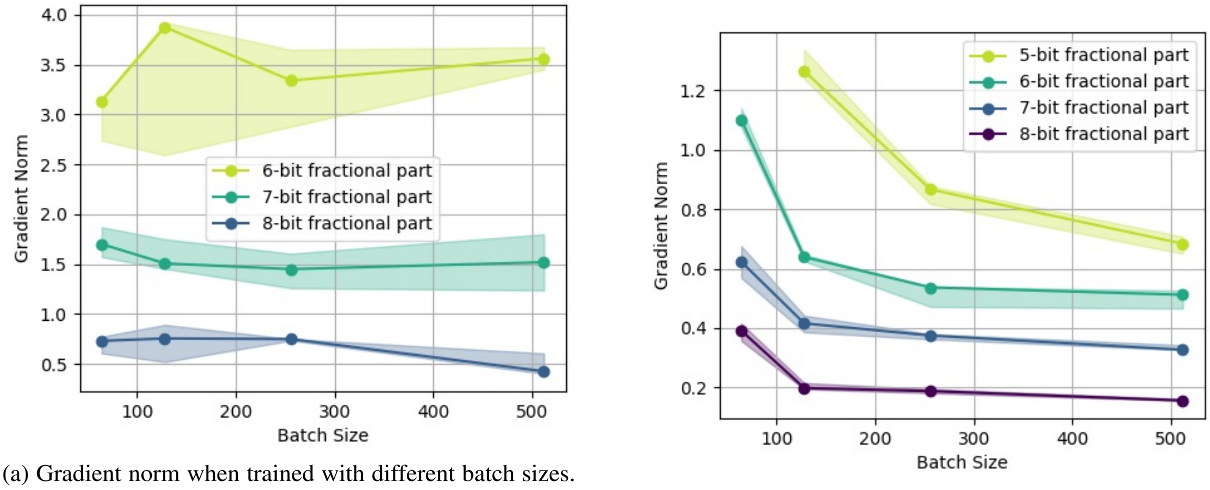
where $C_B = (2 + 4\eta\mathcal{L})$ and $C_V = 2$ are constants.

The term $\frac{4(L(\boldsymbol{w}_0) - L_{\min})}{\eta T}$ diminishes as $T \to \infty$, indicating that the algorithm converges to a region where the expected squared gradient norm $\|\nabla L(\boldsymbol{w}_t)\|^2$ is bounded by an error floor. The error from the difference between the QAT gradient $\nabla_{\widehat{\boldsymbol{w}}} L(\widehat{\boldsymbol{w}})$ and the true gradient $\nabla L(\boldsymbol{w})$ is not reduced by increasing the mini-batch size b. The magnitude of B_W is primarily determined by the precision of weight quantization Δ_W. The variance term $C_V \eta L(\sigma_S^2 + \sigma_Q^2)/b$ captures the noise from two sources: data sampling variance (σ_S^2/b) and stochastic rounding variance. This entire variance contribution is inversely proportional to the mini-batch size b. If activation/gradient precision is reduced, σ_Q^2 increases (e.g., quadrupling if precision is reduced by 1 bit, as $\sigma_Q^2 \propto 2^{-2B}$). Theorem 1 shows that increasing the batch size b can directly compensate for this increase in σ_Q^2, keeping the term $(\sigma_S^2 + \sigma_Q^2)/b$ constant or even reducing it. This is a key mechanism for enabling aggressive quantization of activations/gradients.

5 Experiment

Our experiments evaluate both gradient norms and downstream task performance, allowing us to verify the theoretical findings and examine their practical implications. Figure 5 reports results on CIFAR-10 and figure 6 reports results on LMSYS-Chat (Zheng et al., 2023), showing that the degradation caused by SR in activation and gradient quantization can be substantially mitigated by increasing the batch size, whereas RTN does not benefit from such scaling.

For downstream evaluation, we fine-tuned Llama-3.2-3B (Dubey et al., 2024) on GSM8K (Cobbe et al., 2021) using both Adam and SGD, and BERT-base-uncased (Devlin et al., 2019) on the GLUE benchmark. The GSM8K results (Tables 1 and 2) are consistent with our theoretical predictions, showing clear accuracy improvements as the batch size increases across



(a) Gradient norm when trained with different batch sizes. The backward activations/gradients are RTN quantized. CIFAR-10, WideResNet-16(Zagoruyko and Komodakis, 2016).

(b) Gradient norm when trained with different batch sizes. The backward activations/gradients are SR quantized. CIFAR-10, WideResNet-16.

Figure 5: Practical experiments with image models. The error bars in (a) and (b) represent the 25th-75th percentiles across independent runs.

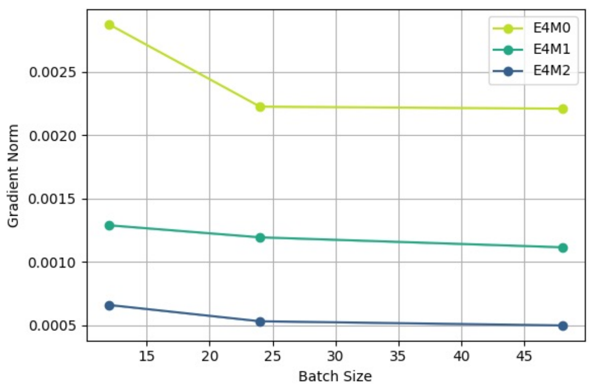


Figure 6: Gradient norm when trained with different batch sizes. The backward activations/gradients are SR quantized. LMSYS-chat, Llama-3.2-3B.

quantization formats. The GLUE (Wang et al., 2018a) benchmark results (Table 3) further confirm this trend, with consistent gains across tasks. Taken together, our theoretical findings—validated by empirical evidence—provide actionable guidelines for training under tight resource constraints.

Table 1: Llama-3.2B fine-tuning on GSM8K with Adam. Reported numbers are end-of-training accuracy.

Quant / Batch Size	8	16	32
E4M0	0.210	0.265	0.290
E4M1	0.245	0.275	0.345
E4M2	0.260	0.305	0.340

Table 2: Llama-3.2B fine-tuning on GSM8K with SGD. Reported numbers are end-of-training accuracy.

Quant / Batch Size	8	32
E4M0	0.180	0.310
E4M1	0.290	0.390
E4M2	0.315	0.365

6 Practical Implications

6.1 Trading Precision for Batch Size on Edge Devices

A key takeaway from our analysis is the explicit trade-off between quantization precision and mini-batch size. On edge hardware, there is often a significant imbalance between the availability of low-precision and high-precision compute units. For instance, Qualcomm's Snapdragon X Elite offers up to 45 TOPS of `INT8` throughput via its Hexagon NPU, while its Adreno GPU provides 4.6 `FP32` TFLOPS (Qualcomm Technologies, Inc., 2024). This means low-precision compute exceeds high-precision compute by roughly an order of magnitude.

Practitioners can leverage this imbalance by using our framework. By accumulating gradients over several micro-batches (e.g., 4–8), one can emulate a larger effective batch size. This allows the use of abundant `INT8`/`INT4` MACs for the bulk of the computation. Our theorem guarantees that the extra stochastic variance introduced by the lower precision is compensated for, as it decays at a rate of $1/b$, ensuring that convergence remains stable.

Table 3: Fine-tuning BERT-base-uncased on GLUE benchmark tasks. We observe a general trend of improved scores as the batch size increases.
†Matthews correlation coefficient. ‡Pearson/Spearman correlation coefficients.

Task	Quant	16	32	64	128	256
CoLA†	E4M0	0.332	0.442	0.447	0.485	0.528
	E4M1	0.341	0.501	0.404	0.560	0.564
	E4M2	0.408	0.443	0.539	0.580	0.618
MRPC	E4M0	0.815	0.820	0.840	0.853	0.862
	E4M1	0.825	0.826	0.847	0.853	0.861
	E4M2	0.846	0.856	0.879	0.883	0.908
MNLI	E4M0	0.619	0.647	0.683	0.717	0.745
	E4M1	0.652	0.689	0.712	0.755	0.782
	E4M2	0.674	0.731	0.755	0.777	0.793
QNLI	E4M0	0.766	0.780	0.820	0.827	0.826
	E4M1	0.797	0.837	0.833	0.846	0.869
	E4M2	0.818	0.840	0.862	0.868	0.898
QQP	E4M0	0.732	0.743	0.768	0.748	0.806
	E4M1	0.755	0.768	0.793	0.808	0.820
	E4M2	0.773	0.856	0.799	0.822	0.835
RTE	E4M0	0.553	0.446	0.511	0.525	0.554
	E4M1	0.504	0.489	0.504	0.482	0.468
	E4M2	0.540	0.504	0.518	0.489	0.554
STS-B‡	E4M0	0.757	0.816	0.824	0.824	0.842
	E4M1	0.808	0.845	0.833	0.843	0.845
	E4M2	0.789	0.826	0.839	0.848	0.847
SST-2	E4M0	0.869	0.862	0.862	0.897	0.897
	E4M1	0.876	0.873	0.878	0.901	0.908
	E4M2	0.860	0.895	0.897	0.906	0.906

6.2 Hardware Overhead of Stochastic Rounding

A natural concern is the hardware cost of implementing SR compared to the simpler RTN. While a fully precise SR unit can be expensive, practical approximations can be implemented with minimal overhead. To provide a concrete estimate, we analyzed the resource usage on an FPGA for converting from `FP32` to `FP8` (`E4M3`).

The mantissa rounding logic requires 9 look-up tables (LUTs) for RTN. In contrast, an SR implementation using a 6-bit linear feedback shift register (LFSR) as a pseudo-random number generator requires only 6 extra registers and 9 LUTs. When compared to the overall cost of a fused multiply-add (FMA) unit (e.g., an `E4M3` FMA requires 223 LUTs and 140 registers), the difference between SR and RTN is negligible. For a 16×16 systolic array (256 MACs), only 32 rounding units are needed for the conversion, making the additional cost of SR insignificant in the context of the entire accelerator.

Furthermore, the feasibility of efficient SR implementation in hardware has been demonstrated in prior work. For example, (Zhang et al., 2022) reports that their BFP converter, which includes the LFSR for SR, accounts for just 4.56% of the chip area and $1.77W$ of power, compared to 47.79% and $15.61W$ for the systolic array itself. This confirms that the benefits of SR can be realized without a significant hardware penalty.

7 Conclusion

This paper studied SR in mixed-precision training and provided a theoretical analysis of mini-batch SGD under quantization. We showed that the variance introduced by SR in activation and gradient quantization decays inversely with batch size, while bias from weight quantization remains unaffected. Empirical results confirmed that SR consistently outperforms deterministic RTN, particularly under aggressive quantization.

These results offer practical guidance for mixed-precision training: larger batch sizes can offset SR-induced noise, enabling more aggressive quantization without loss of convergence. This insight has direct implications for fine-tuning and training LLMs under resource constraints. Future work may examine adaptive batch sizing and broader quantization schemes.

Acknowledgments

We thank the anonymous reviewers for their insightful comments and suggestions that helped improve this paper. We are also grateful to Ebby Samson for his help during the rebuttal on hardware cost modeling. We acknowledge the resources provided by AMD that made this research possible.

Limitations

While this work provides valuable insights into SR-based mixed-precision training, several limitations should be acknowledged:

- **Quantization Schemes and Complementary Techniques:** Our analysis primarily centered on uniform quantization with SR. The interaction of SR and batch size with other advanced quantization techniques, such as non-uniform quantization, block-wise quantization (which we noted as complementary), or learned quantization, remains an area for future exploration. Similarly, a detailed investigation of the interplay with other mixed-precision optimization techniques like dynamic loss scaling or adaptive gradient clipping was beyond the scope of this paper.

- **Hardware Considerations and Performance Metrics:** Our experimental validation focused on convergence behavior (gradient norms, accuracy) rather than direct measurements of training speedup, memory footprint reduction, or energy consumption on target edge hardware. Such practical performance metrics are crucial for assessing the full benefits for edge deployment.

These limitations offer avenues for future research to build upon the foundational understanding of SR in low-precision training established in this work.

References

Ron Banner, Itay Hubara, Elad Hoffer, and Daniel Soudry. 2018. Scalable methods for 8-bit training of neural networks. *Advances in neural information processing systems*, 31.

Charlie Blake, Douglas Orr, and Carlo Luschi. 2023. Unit scaling: Out-of-the-box low-precision training. In *International Conference on Machine Learning*, pages 2548–2576. PMLR.

Jianfei Chen, Yu Gai, Zhewei Yao, Michael W. Mahoney, and Joseph E. Gonzalez. 2020. A statistical framework for low-bitwidth training of deep neural networks. In *Advances in Neural Information Processing Systems*, volume 33, pages 883–894.

Brian Chmiel, Ron Banner, Elad Hoffer, Hilla Ben Yaacov, and Daniel Soudry. 2021. Logarithmic unbiased quantization: Practical 4-bit training in deep learning.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Matteo Croci, Massimiliano Fasi, Nicholas J. Higham, Theo Mary, and Mantas Mikaitis. 2022. Stochastic rounding: Implementation, error analysis and applications. *Royal Society Open Science*, 9(3):211631.

Dipankar Das, Naveen Mellempudi, Dheevatsa Mudigere, Dhiraj Kalamkar, Sasikanth Avancha, Kunal Banerjee, Srinivas Sridharan, Karthik Vaidyanathan, Bharat Kaul, Evangelos Georganas, Alexander Heinecke, Pradeep Dubey, Jesus Corbal, Nikita Shustrov, Roma Dubtsov, Evarist Fomenko, and Vadim Pirogov. 2018. Mixed precision training of convolutional neural networks using integer operations.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

Zhen Dong, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. 2019. Hawq: Hessian aware quantization of neural networks with mixed-precision. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 293–302.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Steven K Esser, Jeffrey L McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S Modha. 2019. Learned step size quantization. *arXiv preprint arXiv:1902.08153*.

Suyog Gupta, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan. 2015. Deep learning with limited numerical precision. In *International Conference on Machine Learning*, pages 1737–1746. PMLR.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. 2018. Quantized neural networks: Training neural networks with low precision weights and activations. *journal of machine learning research*, 18(187):1–30.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. 2018. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2704–2713.

Vinay Joshi, Geethan Karunaratne, Manuel Le Gallo, Irem Boybat, Christophe Piveteau, Abu Sebastian, Bipin Rajendran, and Evangelos Eleftheriou. 2020. *ESSOP: Efficient and scalable stochastic outer product architecture for deep learning*, pages 1–5.

Savitha Viswanadh Kandala and Ambuj Varshney. 2024. Your data, your model: A framework for training and deploying foundational language models for embedded devices. In *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*, pages 1704–1706.

Simon Knowles. 2021. Graphcore. In *2021 IEEE Hot Chips 33 Symposium (HCS)*, pages 1–25.

Wonyeol Lee, Rahul Sharma, and Alex Aiken. Training with mixed-precision floating-point assignments. *Transactions on Machine Learning Research*.

Hao Li, Soham De, Zheng Xu, Christoph Studer, Hanan Samet, and Tom Goldstein. 2017. Training quantized nets: A deeper understanding. In *Advances in Neural Information Processing Systems*, volume 30.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. 2017. Mixed precision training. *arXiv preprint arXiv:1710.03740*.

Houwen Peng, Kan Wu, Yixuan Wei, Guoshuai Zhao, Yuxiang Yang, Ze Liu, Yifan Xiong, Ziyue Yang, Bolin Ni, Jingcheng Hu, Ruihang Li, Miaosen Zhang, Chen Li, Jia Ning, Ruizhe Wang, Zheng Zhang, Shuguang Liu, Joe Chau, Han Hu, and Peng Cheng. 2023. Fp8-lm: Training fp8 large language models. *arXiv preprint arXiv:2310.18313*.

Qualcomm Technologies, Inc. 2024. Snapdragon X Elite Laptop Platform. `https://www.qualcomm.com/products/mobile/snapdragon/laptops-and-tablets/snapdragon-x-elite`. Accessed: 2024-09-11.

Xiao Sun, Naigang Wang, Chia-Yu Chen, Jiamin Ni, Ankur Agrawal, Xiaodong Cui, Swagath Venkataramani, Kaoutar El Maghraoui, Vijayalakshmi Viji Srinivasan, and Kailash Gopalakrishnan. 2020. Ultra-low precision 4-bit training of deep neural networks. *Advances in Neural Information Processing Systems*, 33:1796–1807.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018a. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Naigang Wang, Jungwook Choi, Daniel Brand, Chia-Yu Chen, and Kailash Gopalakrishnan. 2018b. Training deep neural networks with 8-bit floating point numbers. *Advances in neural information processing systems*, 31.

Shuang Wu, Guoqi Li, Feng Chen, and Luping Shi. 2018. Training and inference with integers in deep neural networks. In *International Conference on Learning Representations*.

Lu Xia, Martijn Anthonissen, Michiel Hochstenbach, and Barry Koren. 2021. A simple and efficient stochastic rounding method for training neural networks in low precision. *arXiv preprint arXiv:2103.13445*.

Lu Xia, Stefano Massei, and Michiel E Hochstenbach. 2025. On the convergence of the gradient descent method with stochastic fixed-point rounding errors under the polyak–łojasiewicz inequality. *Computational Optimization and Applications*, pages 1–47.

Lu Xia, Stefano Massei, Michiel E Hochstenbach, and Barry Koren. 2022. On the influence of stochastic roundoff errors and their bias on the convergence of the gradient descent method with low-precision floating-point computation. *arXiv preprint arXiv:2202.12276*.

Lu Xia, Stefano Massei, Michiel E Hochstenbach, and Barry Koren. 2024. On stochastic roundoff errors in gradient descent with low-precision computation. *Journal of Optimization Theory and Applications*, 200(2):634–668.

Yukuan Yang, Lei Deng, Shuang Wu, Tianyi Yan, Yuan Xie, and Guoqi Li. 2020. Training high-performance and large-scale deep neural networks with full 8-bit integers. *Neural Networks*, 125:70–82.

Sergey Zagoruyko and Nikos Komodakis. 2016. Wide residual networks. In *British Machine Vision Conference 2016*. British Machine Vision Association.

Yiren Zhang, Pu Li, Xuefei Wang, Zizhang Liu, and Weisheng Zhao. 2022. Bfx: A bit-flexible format for training and inference of neural networks. In *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 63–78. IEEE.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric P Xing, and 1 others. 2023. Lmsys-chat-1m: A large-scale real-world llm conversation dataset. *arXiv preprint arXiv:2309.11998*.

Aojun Zhou, Anbang Yao, Yiwen Guo, Lin Xu, and Yurong Chen. 2017. Incremental network quantization: Towards lossless cnns with low-precision weights. *arXiv preprint arXiv:1702.03044*.

A Proof of Lemma 1

Proof. Assume $L(\boldsymbol{w})$ is $\mathcal{L}$-smooth, i.e. $\|\nabla L(\boldsymbol{w}') - \nabla L(\boldsymbol{w})\| \leq \mathcal{L}\|\boldsymbol{w}' - \boldsymbol{w}\|$ for all $\boldsymbol{w}', \boldsymbol{w}$. Let $\hat{\boldsymbol{w}}$ be obtained by elementwise uniform quantization with step Δ_W. For RTN

$$|w_i - \hat{w}_i| \leq \tfrac{\Delta_W}{2} \quad \forall i \quad \Rightarrow \quad \|\hat{\boldsymbol{w}} - \boldsymbol{w}\|^2 = \sum_{i=1}^{d}(w_i - \hat{w}_i)^2 \leq d\Big(\tfrac{\Delta_W}{2}\Big)^2, \tag{20}$$

hence $\|\hat{\boldsymbol{w}} - \boldsymbol{w}\| \leq \frac{\sqrt{d}}{2}\Delta_W$. By $\mathcal{L}$-smoothness,

$$\|\nabla L(\hat{\boldsymbol{w}}) - \nabla L(\boldsymbol{w})\| \leq \mathcal{L}\,\|\hat{\boldsymbol{w}} - \boldsymbol{w}\| \leq \mathcal{L}\,\tfrac{\sqrt{d}}{2}\,\Delta_W. \tag{21}$$

Therefore the bias bound is

$$B_W := \|\nabla L(\hat{\boldsymbol{w}}) - \nabla L(\boldsymbol{w})\| \;\leq\; \mathcal{L}\,\tfrac{\sqrt{d}}{2}\,\Delta_W \tag{22}$$

And for SR, $|w_i - \hat{w}_i| \leq \Delta_W$, the bias bound is

$$B_W \leq \mathcal{L}\sqrt{d}\Delta_W \tag{23}$$

□

B Proof of MSE Decomposition in Lemma 2

Proof. Let $\nabla W_{ij}^{(\text{true-mini})} = \frac{1}{b}\sum_{k\in\text{batch}} A_{ki}A_{kj}^{\text{out}}$ and define

$$E \;=\; \nabla W_{ij}^{(\text{full})} - \widehat{\nabla W}_{ij}^{(\text{quant-mini})} \;=\; S + Q', \tag{24}$$

where $S = \nabla W_{ij}^{(\text{full})} - \nabla W_{ij}^{(\text{true-mini})}$ and

$$Q' \;=\; \nabla W_{ij}^{(\text{true-mini})} - \widehat{\nabla W}_{ij}^{(\text{quant-mini})} = \frac{1}{b}\sum_{k\in\text{batch}}\Big(A_{ki}A_{kj}^{\text{out}} - \widehat{A}_{ki}\,\widehat{A^{\text{out}}}_{kj}\Big). \tag{25}$$

Elementwise SR is unbiased and independent across elements/samples: $\mathbb{E}[\widehat{A} \mid A] = A$, $\mathbb{E}[\widehat{A^{\text{out}}} \mid A^{\text{out}}] = A^{\text{out}}$, and the errors are independent of (A, A^{out}) given the values.

Conditioning on the realized mini-batch, S is deterministic and

$$\mathbb{E}_\epsilon[Q' \mid \text{mini-batch}] = \frac{1}{b}\sum_k\big(A_{ki}A_{kj}^{\text{out}} - \mathbb{E}_\epsilon[\widehat{A}_{ki}\widehat{A^{\text{out}}}_{kj} \mid \text{samples}]\big) = 0, \tag{26}$$

because $\mathbb{E}[\widehat{A}_{ki}\widehat{A^{\text{out}}}_{kj} \mid \text{samples}] = A_{ki}A_{kj}^{\text{out}}$ by independence and zero-mean error. Hence

$$\mathbb{E}[E^2] = \mathbb{E}[(S+Q')^2] = \mathbb{E}[S^2] + \mathbb{E}[(Q')^2] + 2\,\mathbb{E}[SQ'] = \underbrace{\mathbb{E}[S^2]}_{T_{ij}^{\text{s}}} + \underbrace{\mathbb{E}[(Q')^2]}_{T_{ij}^{\mathbb{Q}}}, \tag{27}$$

since $\mathbb{E}[SQ'] = \mathbb{E}_{\text{samples}}[S\,\mathbb{E}_\epsilon[Q'|\text{samples}]] = 0$. □

C Proof of Lemma 3

Proof. From Lemma 2,

$$T_{ij}^{\mathbb{Q}} = \mathbb{E}\left[\left(\frac{1}{b}\sum_k E_{k,ij}\right)^2\right], \qquad E_{k,ij} = A_{ki}A_{kj}^{\text{out}} - \widehat{A}_{ki}\,\widehat{A^{\text{out}}}_{kj}. \tag{28}$$

Let $X = A_{ki}, Y = A^{\text{out}}_{kj}$ and $e_X = X - \widehat{X}$, $e_Y = Y - \widehat{Y}$. Then

$$E_{k,ij} = XY - \widehat{X}\widehat{Y} = Xe_Y + Ye_X - e_Xe_Y. \tag{29}$$

Under the SR assumptions, $\mathbb{E}[e_X \mid X] = \mathbb{E}[e_Y \mid Y] = 0$, (e_X, e_Y) are independent across elements/samples and independent of (X, Y); thus $E_{k,ij}$ are i.i.d., zero-mean across k, and

$$T^{\mathbb{Q}}_{ij} = \frac{1}{b}\,\sigma^2_{E_{ij}}, \qquad \sigma^2_{E_{ij}} = \mathbb{E}\big[(Xe_Y + Ye_X - e_Xe_Y)^2\big]. \tag{30}$$

Expanding and using the independence/zero-mean properties,

$$\sigma^2_{E_{ij}} = \mathbb{E}[X^2]\,\underbrace{\mathbb{E}[e_Y^2]}_{\sigma^2_{A^{\text{out}}}} + \mathbb{E}[Y^2]\,\underbrace{\mathbb{E}[e_X^2]}_{\sigma^2_A} + \sigma^2_A\,\sigma^2_{A^{\text{out}}}. \tag{31}$$

For uniform quantization with step sizes $\Delta_A, \Delta_{A^{\text{out}}}$, $|e_X| \le \Delta_A$ and $|e_Y| \le \Delta_{A^{\text{out}}}$, hence $\sigma^2_A \le \Delta^2_A$ and $\sigma^2_{A^{\text{out}}} \le \Delta^2_{A^{\text{out}}}$. Therefore

$$T^{\mathbb{Q}}_{ij} \;\le\; \frac{1}{b}\left(\mathbb{E}[A^2_{ki}]\Delta^2_{A^{\text{out}}} + \mathbb{E}[(A^{\text{out}}_{kj})^2]\Delta^2_A + \Delta^2_A\,\Delta^2_{A^{\text{out}}}\right). \tag{32}$$

If $\Delta \propto 2^{-B}$ (bitwidth B), then $T^{\mathbb{Q}}_{ij} \propto \frac{1}{b}\,2^{-2B}$ with constants depending on $\mathbb{E}[A^2_{ki}]$ and $\mathbb{E}[(A^{\text{out}}_{kj})^2]$. □

D Proof of Theorem 1

Proof. Let $L(\boldsymbol{w}) = \mathbb{E}_{x,y}[L(\boldsymbol{w}; x, y)]$ denote the expected loss. Assume: (i) $L(\boldsymbol{w})$ is $\mathcal{L}$-smooth, i.e., $\|\nabla L(\boldsymbol{w}') - \nabla L(\boldsymbol{w})\| \le \mathcal{L}\|\boldsymbol{w}' - \boldsymbol{w}\|$ for all $\boldsymbol{w}', \boldsymbol{w}$; (ii) $L(\boldsymbol{w})$ is bounded below by $L_{\min}$. The SGD update is $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta\,\tilde{G}(\boldsymbol{w}_t)$ with constant step size $\eta > 0$. Define

$$\widehat{G}(\boldsymbol{w}_t) := \mathbb{E}_t[\tilde{G}(\boldsymbol{w}_t)], \qquad V^2 := \frac{\sigma^2_S + \sigma^2_Q}{b}, \tag{33}$$

so that $\mathbb{E}_t[\|\tilde{G}(\boldsymbol{w}_t)\|^2] \le V^2 + \|\widehat{G}(\boldsymbol{w}_t)\|^2$.

Step 1: $\mathcal{L}$-smoothness. By $\mathcal{L}$-smoothness and $\boldsymbol{w}_{t+1} - \boldsymbol{w}_t = -\eta\tilde{G}(\boldsymbol{w}_t)$,

$$\begin{aligned}\mathbb{E}_t[L(\boldsymbol{w}_{t+1})] &\le L(\boldsymbol{w}_t) + \langle \nabla L(\boldsymbol{w}_t), \mathbb{E}_t[\boldsymbol{w}_{t+1} - \boldsymbol{w}_t]\rangle + \frac{\mathcal{L}}{2}\,\mathbb{E}_t[\|\boldsymbol{w}_{t+1} - \boldsymbol{w}_t\|^2] \\ &= L(\boldsymbol{w}_t) - \eta\langle \nabla L(\boldsymbol{w}_t), \widehat{G}(\boldsymbol{w}_t)\rangle + \frac{\eta^2\mathcal{L}}{2}\,\mathbb{E}_t[\|\tilde{G}(\boldsymbol{w}_t)\|^2] \\ &\le L(\boldsymbol{w}_t) - \eta\langle \nabla L(\boldsymbol{w}_t), \widehat{G}(\boldsymbol{w}_t)\rangle + \frac{\eta^2\mathcal{L}}{2}\Big(V^2 + \|\widehat{G}(\boldsymbol{w}_t)\|^2\Big).\end{aligned} \tag{34}$$

Step 2: Bounds using the bias lemma. By Lemma 1, $\|\widehat{G}(\boldsymbol{w}_t) - \nabla L(\boldsymbol{w}_t)\| \le B_W$.

$$\begin{aligned}\langle \nabla L(\boldsymbol{w}_t), \widehat{G}(\boldsymbol{w}_t)\rangle &= \|\nabla L(\boldsymbol{w}_t)\|^2 + \langle \nabla L(\boldsymbol{w}_t), \widehat{G}(\boldsymbol{w}_t) - \nabla L(\boldsymbol{w}_t)\rangle \\ &\ge \|\nabla L(\boldsymbol{w}_t)\|^2 - \|\nabla L(\boldsymbol{w}_t)\|\,\|\widehat{G}(\boldsymbol{w}_t) - \nabla L(\boldsymbol{w}_t)\| \\ &\ge \|\nabla L(\boldsymbol{w}_t)\|^2 - B_W\,\|\nabla L(\boldsymbol{w}_t)\|,\end{aligned} \tag{35}$$

and

$$\begin{aligned}\|\widehat{G}(\boldsymbol{w}_t)\|^2 &= \|\nabla L(\boldsymbol{w}_t) + (\widehat{G}(\boldsymbol{w}_t) - \nabla L(\boldsymbol{w}_t))\|^2 \\ &\le \big(\|\nabla L(\boldsymbol{w}_t)\| + \|\widehat{G}(\boldsymbol{w}_t) - \nabla L(\boldsymbol{w}_t)\|\big)^2 \\ &\le \big(\|\nabla L(\boldsymbol{w}_t)\| + B_W\big)^2 \;\le\; 2\|\nabla L(\boldsymbol{w}_t)\|^2 + 2B_W^2.\end{aligned} \tag{36}$$

Step 3: Substitute (35) and (36) into (34). We get

$$\begin{aligned}\mathbb{E}_t[L(\boldsymbol{w}_{t+1})] &\le L(\boldsymbol{w}_t) - \eta\Big(\|\nabla L(\boldsymbol{w}_t)\|^2 - B_W\|\nabla L(\boldsymbol{w}_t)\|\Big) + \frac{\eta^2\mathcal{L}}{2}\Big(V^2 + 2\|\nabla L(\boldsymbol{w}_t)\|^2 + 2B_W^2\Big)\\ &= L(\boldsymbol{w}_t) - \eta\|\nabla L(\boldsymbol{w}_t)\|^2 + \eta B_W\|\nabla L(\boldsymbol{w}_t)\| + \eta^2\mathcal{L}\|\nabla L(\boldsymbol{w}_t)\|^2 + \eta^2\mathcal{L}B_W^2 + \frac{\eta^2\mathcal{L}}{2}V^2.\end{aligned} \tag{37}$$

Step 4: Rearrangement and Young's inequality. Move the gradient-norm terms to the left:

$$\eta\|\nabla L(\boldsymbol{w}_t)\|^2 - \eta^2\mathcal{L}\|\nabla L(\boldsymbol{w}_t)\|^2 \le L(\boldsymbol{w}_t) - \mathbb{E}_t[L(\boldsymbol{w}_{t+1})] + \eta B_W\|\nabla L(\boldsymbol{w}_t)\| + \eta^2\mathcal{L}B_W^2 + \frac{\eta^2\mathcal{L}}{2}V^2.$$

Apply Young's inequality to the linear term with $\epsilon = 1$:

$$\eta B_W\|\nabla L(\boldsymbol{w}_t)\| \ \le\ \frac{\eta}{2}\|\nabla L(\boldsymbol{w}_t)\|^2 + \frac{\eta}{2}B_W^2. \tag{38}$$

Thus,

$$\big(\eta - \eta^2\mathcal{L} - \tfrac{\eta}{2}\big)\|\nabla L(\boldsymbol{w}_t)\|^2 \le L(\boldsymbol{w}_t) - \mathbb{E}_t[L(\boldsymbol{w}_{t+1})] + \Big(\tfrac{\eta}{2} + \eta^2\mathcal{L}\Big)B_W^2 + \frac{\eta^2\mathcal{L}}{2}V^2.$$

Equivalently,

$$\eta\Big(\tfrac{1}{2} - \eta\mathcal{L}\Big)\,\|\nabla L(\boldsymbol{w}_t)\|^2 \ \le\ L(\boldsymbol{w}_t) - \mathbb{E}_t[L(\boldsymbol{w}_{t+1})] + \eta\Big(\tfrac{1}{2} + \eta\mathcal{L}\Big)B_W^2 + \frac{\eta^2\mathcal{L}}{2}V^2. \tag{39}$$

Step 5: Telescoping If $\eta \le \frac{1}{4\mathcal{L}}$, then $\frac{1}{2} - \eta\mathcal{L} \ge \frac{1}{4}$, so the left side of (39) is at least $\frac{\eta}{4}\|\nabla L(\boldsymbol{w}_t)\|^2$. Taking full expectation,

$$\frac{\eta}{4}\,\mathbb{E}[\|\nabla L(\boldsymbol{w}_t)\|^2] \le \mathbb{E}[L(\boldsymbol{w}_t)] - \mathbb{E}[L(\boldsymbol{w}_{t+1})] + \eta\Big(\tfrac{1}{2} + \eta\mathcal{L}\Big)B_W^2 + \frac{\eta^2\mathcal{L}}{2}V^2. \tag{40}$$

Summing $t = 0$ to $T-1$ and using telescoping plus $L(\boldsymbol{w}_T) \ge L_{\min}$,

$$\frac{\eta}{4}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla L(\boldsymbol{w}_t)\|^2] \le L(\boldsymbol{w}_0) - L_{\min} + T\,\eta\Big(\tfrac{1}{2} + \eta\mathcal{L}\Big)B_W^2 + T\,\frac{\eta^2\mathcal{L}}{2}V^2. \tag{41}$$

Divide by T and multiply by $4/\eta$:

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla L(\boldsymbol{w}_t)\|^2] \le \frac{4\,(L(\boldsymbol{w}_0) - L_{\min})}{\eta T} + (2 + 4\eta\mathcal{L})\,B_W^2 + 2\eta\mathcal{L}\,V^2. \tag{42}$$

Finally substitute $V^2 = (\sigma_S^2 + \sigma_Q^2)/b$ to obtain

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\|\nabla L(\boldsymbol{w}_t)\|^2] \ \le\ \frac{4\,(L(\boldsymbol{w}_0) - L_{\min})}{\eta T} + (2 + 4\eta\mathcal{L})\,B_W^2 + 2\eta\mathcal{L}\,\frac{\sigma_S^2 + \sigma_Q^2}{b}, \tag{43}$$

which is the claimed bound with $C_B = 2 + 4\eta\mathcal{L}$ and $C_V = 2$. □

E Experiment setup

For CIFAR-10 experiments, we train a Wide ResNet-16-4 model on the CIFAR-10 dataset. A constant learning rate of 1×10^{-4} is used, and standard data augmentation is disabled to enhance stability and isolate the quantization effects under investigation. Training proceeds for 20,000 steps, which is sufficient for the training loss to stabilize near zero across most configurations. As a measure of convergence quality near a stationary point, we evaluate the squared ℓ_2 norm of the full-precision gradient, $|\mathbb{E}[\nabla_\theta L(\theta)]|^2$, computed at the end of each epoch after step 15,000.

For LMSYS-Chat experiments, we fine-tune a Llama-3.2-3B model on the first 1,000 conversations from the LMSYS-Chat dataset, using a next-token prediction objective in SFT format (assistant prompts masked out). We train for 800 steps using the Adam optimizer with a learning rate of 5×10^{-5}. The training loss reliably stabilizes near zero across configurations. Convergence is assessed via the squared ℓ_2 norm of the full-precision gradient, computed at the end of epochs after step 400.

For downstream evaluation, we fine-tune Llama-3.2-3B on GSM8K for 100 steps with Adam (learning rate 5×10^{-7}, context length 512), and for 300 steps with vanilla SGD (learning rate 10^{-3}). End-of-training accuracy is reported across different batch sizes. Additionally, we fine-tune BERT-base-uncased on the GLUE benchmark, training each task for 1,500 steps with Adam (learning rate 2×10^{-5}) and batch sizes in $16, 32, 64, 128, 256$. Task-specific metrics are used for evaluation: accuracy for most tasks, Matthews correlation for CoLA, and Pearson/Spearman average for STS-B.

Across all experiments, we evaluate multiple quantization formats (`E4M0`, `E4M1`, `E4M2`) applied to activations and gradients under both stochastic rounding (SR) and round-to-nearest (RTN). Training and fine-tuning are implemented in PyTorch 2.7 and executed on 2 NVIDIA H100 GPUs for around 1 week.

F Licenses, Models, and Datasets

This research makes use of publicly available models and datasets. Their sources and licenses are summarized below to ensure transparency and proper attribution.

- **Llama Model Family:** Llama 3 models are released under the **Meta Llama 3 Community License Agreement**.
- **CIFAR-10 Dataset:** Created by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton; hosted by the University of Toronto (`https://www.cs.toronto.edu/~kriz/cifar.html`).
- **LMSYS-Chat Dataset:** The `lmsys-chat-1m` dataset is distributed under the **LMSYS-Chat-1M Dataset License Agreement**.
- **GSM8K (Grade School Math 8K):** Released by OpenAI and licensed under the **Apache2 License** (`https://huggingface.co/datasets/openai/gsm8k`).
- **BERT (Bidirectional Encoder Representations from Transformers):** Released by Google AI Language and licensed under the **Apache License 2.0** (`https://github.com/google-research/bert`).
- **GLUE Benchmark:** A collection of multiple NLP datasets, distributed under the **Apache 2.0 License** license (`https://www.tensorflow.org/datasets/catalog/glue`).

All models and datasets were used in accordance with their respective licensing terms. All the models and datasets are used according to the intended usage. We thank the creators and maintainers of these resources for their contributions to the research community.

We did not collect or use any data for this work. Therefore, no personally identifying information (PII) or offensive content is present, and no additional anonymization or protection steps were required. No datasets or artifacts were collected or produced as part of this work, so documentation of domains, languages, linguistic phenomena, or demographic groups is not applicable.

G Potential Risk

The methods presented in this paper aim to enhance the efficiency of training Large Language Models (LLMs), particularly through improved understanding and application of Stochastic Rounding (SR) in mixed-precision settings. By potentially lowering computational and memory barriers, our work could contribute to the democratization of LLM development and fine-tuning, enabling smaller research groups, startups, or even on-device applications. This increased accessibility can foster innovation and allow for more diverse applications tailored to specific needs, potentially benefiting areas like personalized education or assistive technologies.

However, as with any technology that makes powerful AI more accessible, there are dual-use considerations. Lowering the technical threshold for training capable LLMs could inadvertently facilitate their misuse for generating disinformation, spam, or other harmful content if not accompanied by responsible development practices and robust safeguards. Furthermore, while our work focuses on training efficiency, it does not inherently address pre-existing biases within datasets or models. Therefore, the efficient training of LLMs must be paired with ongoing efforts in bias detection, mitigation, and the ethical deployment of these increasingly capable systems. We believe continued research into efficient and responsible AI development is crucial.

H Use Of AI Assistants

In this work, we used AI coding and polishing for writing.