PRE-TRAINING LARGE LANGUAGE MODELS WITH DYNAMIC PRECISION: LOW-COST COMPUTATION WITH HIGH-FIDELITY PERFORMANCE

Boao Kong[1,*], **Weichen Jia**[1,*], **Engao Zhang**[1,*], **Guohong Li**[1,*], **Yonghan Dong**[2], **Yao Wang**[2], **Yaoyuan Wang**[2], **Yunke Peng**[2], **Kun Yuan**[1,†]
[1]Peking University, Beijing, China [2]Huawei Technologies Co., Ltd., Shenzhen, China
[*]Equal contributions [†]Corresponding author: kunyuan@pku.edu.cn

ABSTRACT

Mixed-precision training (MPT) is widely employed in large-scale deep learning tasks to balance computational efficiency and model performance. However, low-precision schemes, including 8-bit and 4-bit, face challenges such as architecture dependence, numerical spikes, and static strategies. To address these issues, this paper proposes the **G**radient **N**or**M** **R**atio (GNMR) metric-driven dynamic precision method. The GNMR index, along with its variant Δ-GNMR index, enables real-time identification of high-risk operators and steps in low-precision training and supports high-precision recovery upon risk detection. We provide a theoretical analysis to validate the proposed dynamic precision strategy. Experimental results on various models demonstrate that the proposed method effectively improves the stability of mixed-precision training and narrows the performance gap between low-precision (4-bit, 8-bit) schemes and popular high-precision training strategies such as BF16.

1 INTRODUCTION

Large-scale Transformer pre-training imposes computational and memory demands, motivating the adoption of mixed-precision training (MPT) (Micikevicius et al., 2017). MPT strategically allocates lower numerical precision to operations that tolerate quantization, while retaining higher precision for sensitive computations, thereby reducing memory usage and improving throughput without sacrificing model accuracy. Early research demonstrated the reliability of FP16 training using techniques such as loss scaling and selective FP32 accumulation (Micikevicius et al., 2017), which were later incorporated into frameworks like NVIDIA AMP and PyTorch AMP. This approach achieved accuracy comparable to FP32 baselines, while delivering a practical 2× speedup.

More recently, BF16 (Wang & Kanwar, 2019) has gained prominence as a 16-bit format with an extended exponent range that mitigates overflow and underflow risks, becoming the default on TPUs and recent GPU architectures. Methodologies for end-to-end FP8 pre-training, including delayed scaling (Peng et al., 2023) and per-tensor or per-block scaling/quantization (Micikevicius et al., 2022), have been optimized for distributed settings. These advances have been complemented by system-level innovations such as model and optimizer sharding (e.g., ZeRO, ZeRO++) (Rajbhandari et al., 2020; Wang et al., 2023), fully sharded data parallelism (FSDP) (Zhao et al., 2023), model and tensor parallelism as implemented in Megatron-LM (Shoeybi et al., 2019), and pipeline parallelism as exemplified by GPipe (Huang et al., 2019). These methods collectively reduce memory footprint and communication overhead at pre-training scale.

Building on BF16 and FP16, several studies have demonstrated FP8 pre-training—including the E4M3 and E5M2 variants (Micikevicius et al., 2022)—for large language models. FP8-LM (Peng et al., 2023) showcased multi-component FP8 training with favorable throughput/peak-memory–accuracy trade-offs, while DeepSeek-V3 (Liu et al., 2024) employed FP8 mixed precision in a 671B mixture-of-experts model, confirming hyperscale feasibility. Recent work streamlined adoption: COAT (Xi et al., 2024) compresses optimizer states and activations to FP8, and INT-FlashAttention integrates INT8 computation into attention mechanisms, enabling hybrid FP8/INT8 pipelines for

long-context training (Chen et al., 2024b). Several studies also explore 4-bit training—including FP4 and NVFP4—on Blackwell-class hardware with selective high-precision fallbacks, suggesting potential throughput and memory gains beyond FP8 (Abecassis et al., 2025; Wang et al., 2025).

1.1 Limitations of Existing Approaches

Despite recent advancements in mixed-precision training, existing methods exhibit several critical limitations that constrain their efficiency and performance in large-scale settings. The principal challenges are outlined below.

Static precision assignment across dynamic training process. Most mixed-precision methods use static precision policies—uniform BF16 or partial FP8 assignments—that remain fixed throughout training. Yet training dynamics are inherently non-stationary: weight, gradient, and activation distributions evolve continuously, creating a fundamental misalignment with fixed precision assignments. This disconnect highlights the need for adaptive, risk-aware controllers that adjust precision in real time. An effective approach would dynamically allocate higher precision during unstable phases (e.g., during unstable gradient phases) and lower precision during stable training phases, guided by stability indicators like gradient norms or activation statistics. This ensures precision is used only when necessary, optimizing both efficiency and stability. It is worth noting that prior work has exploited training dynamics indicators to enhance stability through gradient clipping, adaptive norm constraints, or dynamic batch sizing (Pascanu et al., 2013; McCandlish et al., 2018; Brock et al., 2021; Takase et al., 2023), but these techniques remain isolated from precision control mechanisms.

Inflexible precision strategies restricted to specific architectures. Current mixed-precision methods are architecture-bound and hand-tuned: precision assignments are fixed per operator or layer and must be re-engineered for each new model or task. In DeepSeek-V3 (Liu et al., 2024), for example, high precision is deliberately retained for components such as the embedding/output head, attention/normalization, and MoE gating, while compute-dense GEMMs run at lower precision—choices that must be revisited whenever the architecture changes (e.g., switching to or reconfiguring a Mixture-of-Experts, or altering the number of activated experts). This rigidity undermines scalability and generality. A better path is a model-agnostic, dynamic controller that selects layer- and operator-level precision in real time based on stability signals (e.g., gradient norms, activation ranges/noise scale, loss spikes), eliminating manual retuning across architectures and improving the flexibility and training-cost efficiency (throughput/peak memory) of mixed-precision training.

Numerical spikes and late-stage instability. Loss and gradient spikes persist in reduced-precision pre-training, driven by outlier amplification and desynchronized scaling across reductions and accumulations. Recent studies propose partial mitigations—spectral-norm-based analyses with architectural adjustments to blunt spikes (Takase et al., 2023), activation smoothing such as Smooth-SwiGLU to suppress outliers (Fishman et al., 2024), and FP8-specific scaling policies and safeguards (Lee et al., 2024). However, these remedies are operator- or architecture-specific, rely on static thresholds or offline calibration, and frequently fall back to higher precision in difficult phases. Thus, they attenuate but do not eliminate late-stage instabilities: rare spikes still surface at scale, hyperparameter robustness remains below full precision, and efficiency gains erode when conservative guardbands are required (Lee et al., 2024). This gap motivates a training-aware, dynamically responsive precision policy rather than a fixed recipe.

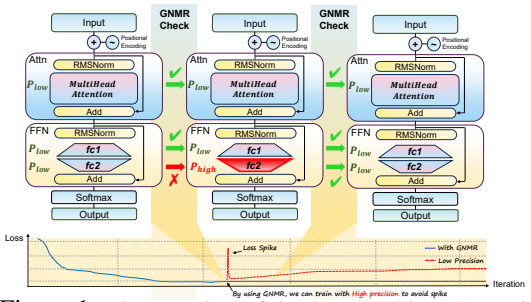

Figure 1: The overview of the GNMR-driven dynamic precision strategy in pre-training LLM tasks.

Related works. Beyond the works discussed above, Appendix A summarizes additional results on mixed-precision training and numerical stability in LLM pre-training.

1.2 Contributions

To directly address the limitations mentioned above, we present a dynamic mixed-precision pre-training framework for LLMs based on the Gradient Norm Ratio (GNMR), which can be illustrated as Figure 1. GNMR enables real-time, stability-aware precision adjustment. Our contributions are:

(C1). A general, architecture-agnostic metric for precision determination. GNMR is a per-module indicator that quantifies stability risk with minimal overhead. Unlike static operator whitelists, it adapts to evolving training dynamics and makes no assumptions about architecture-specific layouts, while remaining aligned with established gradient-norm stability principles.

(C2). A dynamic precision controller for LLM pre-training. We propose an architecture-native, operator-local controller that automatically adjusts numerical formats based on GNMR risk, elevating unstable modules to higher precision and retaining low precision for stable paths, thereby providing runtime adaptivity for non-stationary training and online intervention against incipient spikes without altering the computation graph or training loop, thus avoiding repeated architecture-specific retuning.

(C3). Theoretical guarantees. We rigorously characterize approximation error under mixed precision and show how dynamic precision selection affects convergence. We further derive a principled scheme for choosing the GNMR threshold, providing formal justification for our controller. These guarantees support safe, non-conservative intervention—mitigating the need for broad guardbands or global fallbacks—while remaining agnostic to architecture.

(C4). Scalability and robustness. Experiments show that GNMR substantially narrows the gap between low- and high-precision training and improves robustness precisely in regimes where static policies fail or diverge. We validate stable convergence across model scales and learning-rate configurations and provide ablations on thresholding and precision routing, demonstrating portability beyond hand-tuned, architecture-specific recipes.

The adaptive precision preserves model quality under efficient low-precision computation, while mitigating late-stage fragility that undermines static policies, thereby supporting deployment at scale.

2 Operator-Level Modeling of Transformers

We formalize transformers at the *operator* level, separating compositional structure (nesting) from within-layer parallelism (sibling operators and aggregation). This abstraction will be used uniformly across attention, feed-forward networks, normalization, and residual paths; concrete instantiations and backpropagation rules are deferred to Appendix B.

Objective. Let $F(\cdot;\Theta)$ be the model mapping and $r(\cdot;\theta^{(r)})$ a readout to logits. For data $(x,y)\sim\mathcal{D}$,

$$L(\Theta,\theta^{(r)}) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\Big[\,\ell\big(r(F(x;\Theta);\theta^{(r)}),\,y\big)\,\Big], \tag{1}$$

where Θ collects the parameters of all operator blocks.

Nest–parallel operator modeling. The network is a composition of U *nested* operators:

$$y^{[0]} = x, \quad y^{[u]} = \mathcal{N}^{[u]}\big(y^{[u-1]};\,\Xi^{[u]}\big) \quad (u = 1,\dots,U). \tag{2}$$

Inside a block $\mathcal{N}^{[u]}$, K_u sibling operators act on the *same* input and their outputs are combined by a parameterized aggregator:

$$z^{[u,k]} := f^{[u,k]}\big(y^{[u-1]};\,\theta^{[u,k]}\big), \quad k = 1,\dots,K_u. \tag{3}$$

Let $z^{[u,0]}$ denote the aggregator that merges $\{z^{[u,k]}\}_{k=1}^{K_u}$ and optionally the skip input $y^{[u-1]}$. Then

$$\mathcal{N}^{[u]} = z^{[u,0]}\big(z^{[u,1]},\dots,z^{[u,K_u]};\,\theta^{[u,0]},\,y^{[u-1]}\big), \quad \text{with} \quad \Xi^{[u]} = \big(\theta^{[u,k]}\big)_{k=0}^{K_u}. \tag{4}$$

This modeling captures core transformer operations: multi-head attention (parallel heads and a combining aggregator), feed-forward layers (one or more parallel MLP branches), residual/skip connections (through the aggregator's dependence on $y^{[u-1]}$), and (pre/post) normalization (as either a sibling operator or part of the aggregator). Further instantiations and backpropagation formulas are provided in Appendix B.

3 Method

3.1 GNMR and Δ-GNMR metric

GNMR metric. Low-precision pre-training of large language models (LLMs) imposes strict requirements on the stability of operators. Consequently, it is essential to evaluate the stability of the

gradient for each operator during pre-training. To this end, we introduce the **Gradient Norm Ratio** (GNMR) metric to obtain a stability reflection for each operator and to examine those with precision risks. Specifically, for $u = 1, 2, \cdots, U$ and $k = 0, 1, 2, \cdots, K_u$, the GNMR of the parameter $\theta^{[u,k]}$ in the t-th iteration is defined as follows:

$$\text{GNMR}_t(\theta^{[u,k]}) := \frac{\left\| g_t^{[u,k]} \right\|}{\sum_{\tau=1}^{t-1} \left\| g_\tau^{[u,k]} \right\| / (t-1)}, \tag{5}$$

where $g_t^{[u,k]}$ denotes the evaluated gradient at t-th step with respect to $\theta^{[u,k]}$. we can also set $\text{GNMR}_1(\theta^{[u,k]}) = 1$.

Comparison to existing gradient stability metrics. Compared to existing gradient stability metrics, such as the gradient spike score (GSS) (Huang et al., 2025b), gradient signal to noise ratio (GSNR) (Jiang et al., 2023), and Jacobian spectral norm (Takase et al., 2023), these approaches face significant limitations. Specifically, GSS relies on full-process gradient norms and cannot be acquired in real-time, GSNR requires large-batch gradients and fails to incorporate historical information effectively, while the Jacobian norm introduces substantial computational overhead. In contrast, GNMR can be directly obtained from the current gradient norm and historical GNMR values, as outlined in Algorithm 1, implying minimal computational cost and real-time capability. Furthermore, GNMR is a dimensionless quantity, facilitating unified horizontal comparison across different operators. This ensures the generalization and scalability of our GNMR-based method for various models.

Δ-GNMR metric. To detect emerging instability that may be missed by a long-horizon average, we define the Δ-GNMR as the ratio between the current GNMR and a short-window average of recent GNMR values. For operator parameter $\theta^{[u,k]}$ at step t, we can define

$$\Delta\text{-GNMR}_t(\theta^{[u,k]}) := \frac{\text{GNMR}_t(\theta^{[u,k]})}{\sum_{\tau=1}^{\delta} \text{GNMR}_{t-\tau}(\theta^{[u,k]})/\delta}, \tag{6}$$

This short-window normalization complements GNMR (which uses the full history) by highlighting abrupt changes in operator-wise stability.

Such a formulation specifically captures short-term, instantaneous instability, enabling more responsive high-precision compensation and thereby strengthening overall training stability. Similar to the GNMR metric, Δ-GNMR can also be computed directly from the current and a limited history of gradient norms (See Algorithm 1), allowing it to be obtained in real-time with minimal computational overhead.

Here we need to clarify that GNMR/Δ-GNMR only determines a per-operator precision *schedule*; it can be implemented by switching between already-available low-/high-precision execution paths, without introducing new custom kernels or changing the computation graph.

3.2 Pre-training models with dynamic precision based on GNMR and Δ-GNMR

Based on the GNMR and Δ-GNMR metric presented in Algorithm 1, operators whose values exceed a predefined threshold during pre-training can be identified as risky. For an operator $z^{[u,k]}$ and its corresponding parameter $\theta^{[u,k]}$, if its GNMR reaches the threshold α_t **or** Δ-GNMR reaches the threshold β_t, it will be switched to high precision and maintain this state for a number of subsequent steps, controlled by the hyperparameter T_{lock}. The operator may be switched back to low precision if its GNMR **and** Δ-GNMR subsequently falls below a sufficiently low value. This dynamic precision strategy is summarized in Algorithm 2, which effectively identifies unstable blocks and equips them with more precise forward and backward computation. Furthermore, we have included Figure 1 to intuitively illustrate our Algorithm 2. Specifically, as training proceeds, we utilize the GNMR and Δ-GNMR metrics to locate risky operators, and restore them to high precision to ensure stable training. The entire process is both iteration-adaptive and operator-adaptive. We initialize the precision state for operators $\{P^{[u,k]} = P_{\text{low}}\}_{u=1,2,\cdots,U}^{k=0,1,2,\cdots,K_u}$, the lock state $\{\text{Lock}^{[u,k]} = 0\}_{u=1,2,\cdots,U}^{k=0,1,2,\cdots,K_u}$ and the historical average gradient norm $\{\text{AGNM}_0(\theta^{[u,k]}) = 0\}_{u=1,2,\cdots,U}^{k=0,1,2,\cdots,K_u}$.

Compatibility with existing mixed-precision strategies for LLM pre-training. Our proposed dynamic precision strategy functions as a plug-and-play module that can be readily integrated into existing mixed-precision training frameworks, such as Deepseek-V3 (Liu et al., 2024) and NVFP4 (Abecassis et al., 2025). Since these frameworks often enforce high precision for certain high-risk

Algorithm 1 $\text{GNMR}_t(\theta^{[u,k]})$ and $\Delta\text{-GNMR}_t(\theta^{[u,k]})$

Input: Stochastic gradient $g_t^{[u,k]}$, historical average gradient norm $\text{AGNM}_{t-1}(\theta^{[u,k]})$, iteration step t, window length δ, ring buffer $\mathcal{B}^{[u,k]}[1{:}\delta]$ that stores the most recent δ values of GNMR (initialized as 0), rolling sum $\text{SGNMR}_{t-1}(\theta^{[u,k]}) = \sum_{\tau=1}^{\delta} \text{GNMR}_{t-\tau}(\theta^{[u,k]})$ (initialized as 0).
$n_t \leftarrow \left\| g_t^{[u,k]} \right\|$
if $\text{AGNM}_{t-1}(\theta^{[u,k]}) = 0$ **then**
  $\text{GNMR}_t \leftarrow 1, \quad \text{AGNM}_t \leftarrow n_t$
else
  $\text{GNMR}_t \leftarrow \frac{n_t}{\text{AGNM}_{t-1}(\theta^{[u,k]})}, \quad \text{AGNM}_t \leftarrow \frac{t-1}{t}\text{AGNM}_{t-1}(\theta^{[u,k]}) + \frac{1}{t} n_t$
end if
if $t > \delta$ **then**
  $\overline{\text{GNMR}}_{t-1} \leftarrow \frac{1}{\delta}\text{SGNMR}_{t-1}(\theta^{[u,k]}), \quad \Delta\text{-GNMR}_t \leftarrow \frac{\text{GNMR}_t}{\overline{\text{GNMR}}_{t-1}}$
else
  $\Delta\text{-GNMR}_t \leftarrow 1$
end if
Update rolling window (for next step):
$j \leftarrow ((t-1) \bmod \delta) + 1$
if $t \le \delta$ **then**
  $\mathcal{B}^{[u,k]}[j] \leftarrow \text{GNMR}_t, \quad \text{SGNMR}_t \leftarrow \text{SGNMR}_{t-1}(\theta^{[u,k]}) + \text{GNMR}_t$
else
  $old \leftarrow \mathcal{B}^{[u,k]}[j], \quad \mathcal{B}^{[u,k]}[j] \leftarrow \text{GNMR}_t, \quad \text{SGNMR}_t \leftarrow \text{SGNMR}_{t-1}(\theta^{[u,k]}) + \text{GNMR}_t - old$
end if
Output: GNMR_t, $\Delta\text{-GNMR}_t$.

Algorithm 2 Pre-training with Dynamic Precision under GNMR and Δ-GNMR

Input: High precision P_{high}, low precision P_{low}, lock steps T_{lock}, GNMR threshold schedule $\{\alpha_t\}$, Δ-GNMR threshold schedule $\{\beta_t\}$ (and Alg. 1 running states).
Init: $P^{[u,k]} \leftarrow P_{\text{low}}$, $\text{Lock}^{[u,k]} \leftarrow 0$ for all $u = 1, \ldots, U,\ k = 0, \ldots, K_u$.
for $t = 1, \cdots, T$, $u = 1, \cdots, U$, and $k = 0, \cdots, K_u$ **do**
  Obtain stochastic gradient $g_t^{[u,k]}$ under $P^{[u,k]}$, get $\text{GNMR}_t(\theta^{[u,k]}),\ \Delta\text{-GNMR}_t(\theta^{[u,k]})$
  $\text{Lock}^{[u,k]} \leftarrow \max\{\text{Lock}^{[u,k]} - 1, 0\}$
  if $\text{GNMR}_t(\theta^{[u,k]}) > \alpha_t \ \vee\ \Delta\text{-GNMR}_t(\theta^{[u,k]}) > \beta_t$ **then**
    $(P^{[u,k]}, \text{Lock}^{[u,k]}) \leftarrow (P_{\text{high}}, T_{\text{lock}})$
  else if $\text{Lock}^{[u,k]} = 0 \ \wedge\ \text{GNMR}_t(\theta^{[u,k]}) \le \alpha_t \ \wedge\ \Delta\text{-GNMR}_t(\theta^{[u,k]}) \le \beta_t$ **then**
    $P^{[u,k]} \leftarrow P_{\text{low}}$
  end if
end for

operators by design, our approach selectively applies dynamic precision only to the operators that remain eligible for low-precision computation. To further address potential inconsistencies arising from mixed-precision operations, we can alternatively treat each transformer block as a holistic unit, applying the dynamic precision strategy uniformly to all low-precision operators within the block. This ensures operational consistency while maintaining flexibility.

Peak memory control in dynamic precision strategy. Compared to pure low-precision training, switching high-risk operators that exceed the GNMR and Δ-GNMR threshold to high precision introduces additional memory usage. Notably, during iterations where multiple operators simultaneously surpass the threshold—even if transient—this can result in substantial peak memory demands.

To effectively reduce such peak memory overhead, we set an upper limit, denoted as $maxO$, on the number of operators allowed to switch to high precision. In implementation, we first identify risky operators via the GNMR and Δ-GNMR metric, then switch the top $maxO$ operators to high precision, while maintaining the rest in low precision. This strategy enables a balance between computational efficiency and model performance, mitigating performance degradation from low-

precision training while controlling peak memory usage. We employ this peak memory control approach in the experiments detailed in Section 5. A simple upper-bound model for the additional peak activation memory under the $maxO$ budget is provided in Appendix D.

Decaying threshold for evolving training dynamics. The GNMR threshold α_t (and β_t for Δ-GNMR) controls how often we trigger high-precision recovery, thereby trading off stability/quality against compute–memory overhead. Empirically, GNMR can be higher at early iterations and gradually decreases as the historical average stabilizes (Figure 2 in Appendix). Our analysis in Section 4.3 further shows that the exceedance probability of a *fixed* threshold decays with training steps. Consequently, keeping α_t fixed can become overly conservative later in training, suppressing necessary high-precision fallbacks. To address this, we use a *decaying* threshold schedule: α_t is set larger in the early stage (to avoid over-triggering) and then decreased to maintain an effective trigger frequency in later stages. A simple instantiation is a two-stage schedule with $\alpha_t = \alpha^{\text{init}}$ for the initial period and $\alpha_t = \alpha^{\text{main}} < \alpha^{\text{init}}$ afterwards (Section 5).

4 Theoretical analysis

In this section, we provide theoretical insights into (i) why GNMR can identify high-risk operators, (ii) why high-risk states can persist across adjacent steps (supporting T_{lock}), and (iii) how a GNMR-triggered dynamic precision policy improves one-step expected descent by reducing mixed-precision error on critical updates. We further analyze the exceedance probability under a fixed threshold and motivate a decaying threshold schedule.

Notation and standing conditions. Let $\mathcal{F}_t$ denote the filtration representing the training history up to step t. Let $\mathcal{B} = \{[u,k] \mid u = 1,\dots,U;\ k = 0,1,\dots,K_u\}$ be the set of operators. For $b = [u,k] \in \mathcal{B}$, write $\rho_t^b := \mathrm{GNMR}_t(\theta^b)$. Given a threshold $\alpha > 1$, define the high-risk set at step t as $\mathcal{H}_t(\alpha) := \{\, b \in \mathcal{B} : \rho_t^b \geq \alpha \,\}$.

We work under standard smoothness/noise conditions for stochastic optimization together with an operator-wise mixed-precision perturbation model; the precise statements are collected in Appendix C.1. Our convergence bound also uses a one-step persistence parameter π_b that lower-bounds the probability of remaining in a high-risk regime. Importantly, this is not an ad-hoc requirement: Appendix C.2 derives an explicit lower bound under benign noise, which provides a sufficient condition for (and interpretation of) π_b.

4.1 Persistence of high-risk operators

Dynamic precision can suffer from *thrashing* if operators frequently toggle between low and high precision. To capture the empirically observed short-term persistence of operator-wise instability, we introduce the following one-step persistence quantity: for each $b \in \mathcal{B}$, there exists $\pi_b \in (0,1]$ and a representative level $\tau_b \geq \alpha$ such that

$$\mathbb{P}\big(\rho_{t+1}^b \geq \tau_b \mid \rho_t^b \geq \alpha\big) \ \geq\ \pi_b. \tag{7}$$

This quantity directly motivates locking: once an operator is promoted, keeping it in high precision for a short window T_{lock} increases the chance of covering consecutive high-risk steps with fewer policy switches. Appendix C.2 provides a sufficient condition and an explicit lower bound (under benign mixed-precision/SGD noise) that makes Eq. (7) interpretable and practically plausible.

4.2 Convergence analysis with dynamic mixed-precision policy

This subsection integrates the dynamic mixed-precision policy—gating operators by GNMR values with lock time $T_{\text{lock}} = 1$—into convergence analysis. If an operator is high-risk at step t, it switches to high precision at $t+1$, curbing mixed-precision errors when critical. The following theorem shows the policy reduces quantization variance at $t+1$ only for operators high-risk at t.

Theorem 4.1 (One-step expected-descent gain of dynamic policy)**.** *Under the standing conditions in Appendix C.1, and for $\eta \leq 1/L$, the expected decrease from θ_{t+1} to θ_{t+2} satisfies*

$$\Delta_{\text{desc}}^{(\text{Low}-\text{Dynamic})} := \mathbb{E}[F(\theta_{t+2})]_{\text{Low}} - \mathbb{E}[F(\theta_{t+2})]_{\text{Dynamic}} \geq \frac{L\eta^2}{2} \sum_{b \in \mathfrak{B}} \mathbb{E}\big[\Delta v^b(\rho_{t+1}^b)\, \mathbf{1}_{H_t(\alpha)}\big], \tag{8}$$

where $\mathbb{E}[F(\theta_{t+2})]_{\text{Dynamic}}$ and $\mathbb{E}[F(\theta_{t+2})]_{\text{Low}}$ denote the expected value of $F(\theta_{t+2})$ conditioned on $\mathcal{F}_t$, obtained with dynamic and low precision in the subsequent steps, respectively. In addition, it holds that

Table 1: The validation perplexity (↓) for the pre-training LLaMA-2 models for variance model size with activation quantization under different precision strategy. α_t and β_t represent the threshold of GNMR and Δ-GNMR, respectively. ✗ means the model fails to converge. N.A. means the training task has not been taken.

precision	α_t	β_t	60M 1.1B tokens	130M 2.2B tokens	350M 6.4B tokens	1.3B 13.1B tokens
4-bit	N.A.	N.A.	67.22	169.21	26.56	✗
4-bit/8-bit	1.5	0.3	**30.59**	**24.66**	18.84	15.71
	2.0	0.5	30.66	24.86	19.07	N.A.
	3.0	1.0	30.68	25.33	18.77	N.A.
8-bit	N.A.	N.A.	30.88	25.13	**18.59**	15.68
16-bit	N.A.	N.A.	30.79	25.06	18.80	**15.56**

Table 2: The validation perplexity (↓) for the pre-training LLaMA-2 models for variance model size with DeepSeek-style mixed-precision strategy under different precision strategy. α_t and β_t represent the threshold of GNMR and Δ-GNMR, respectively.

precision	α_t	β_t	Training tokens (B)								
			1.3	2.6	3.9	5.2	6.6	7.9	9.2	10.5	13.1
6-bit/8-bit	1.5/1.1$^{\diamond}$	0.5	28.52	21.17	19.07	17.88	17.06	16.43	15.93	15.57	15.20
8-bit	N.A.	N.A.	29.54	21.57	19.25	17.96	17.11	16.45	15.94	15.56	15.18
8-bit/16-bit	1.5/1.1$^{\diamond}$	N.A.	29.50	21.48	19.14	17.89	17.04	16.37	15.86	15.48	15.10
16-bit	N.A.	N.A.	29.10	21.35	19.08	17.84	16.98	16.33	15.82	15.44	15.06

$^{\diamond}$ We use the two-stage threshold in this training task, see Appendix E.2 for more details.

$$\Delta_{\text{desc}}^{(\text{Low-Dynamic})} \geq \frac{L\eta^2}{2} \sum_{b\in\mathfrak{B}} \mathbb{P}(H_t(\alpha))\, \pi^b\, \Delta v^b(\tau^b). \tag{9}$$

(See the proof in Appendix C.3.)

Eq. (9) decomposes the gain into three factors: (i) the trigger frequency $\mathbb{P}(\rho_t^b \geq \alpha)$, (ii) the persistence term π_b, and (iii) the error-gap $\Delta v_b(\tau_b)$ (how much high precision reduces mixed-precision error in risky regimes). This directly explains our controller knobs: T_{lock} targets persistence to reduce thrashing, while a hard budget $maxO$ (TopK promotion) focuses high precision on the dominant-risk operators that contribute most to the sum.

4.3 Probabilistic Analysis of Gradient Norm to Mean Ratio

We now analyze the probabilistic behavior of the Gradient Norm to Mean Ratio (GNMR) for a fixed operator $[u, k]$.

Theorem 4.2 (Fixed-threshold GNMR: exceedance probability decays with t)**.** *For any $\alpha > 1$ and $[u, k] \in \mathfrak{B}$, let $\varepsilon = \alpha - 1$. Then For any $\alpha > 1$ and $b \in \mathcal{B}$, let $\epsilon = \alpha - 1$. Under the sub-exponential tail condition stated in Appendix C.4, there exists $c > 0$ such that*

$$\mathbb{P}\left(GNMR_t(\theta^{[u,k]}) \geq \alpha\right) \leq \exp\left(-c \min\left\{\frac{\varepsilon^2}{\kappa_b^2}, \frac{\varepsilon}{\kappa_b}\right\}\right) + 2\exp\left(-c(t-1)\min\left\{\frac{\varepsilon^2}{\kappa_b^2}, \frac{\varepsilon}{\kappa_b}\right\}\right). \tag{10}$$

(See the proof in Appendix C.4.)

Theorem 4.2 shows that, due to the concentration of the historical mean, the probability of exceeding a *fixed* GNMR threshold can diminish as training progresses. Consequently, a static threshold may become overly conservative in later stages and miss emerging instabilities. This motivates *threshold scheduling* that gradually relaxes the trigger condition (i.e., letting α_t decay toward 1), so that the controller maintains a non-vanishing sensitivity to operator-wise risk throughout training. In practice, we adopt a simple two-stage schedule with a higher threshold in the initial warm-up phase and a lower threshold afterward (see Appendix E.2).

5 EXPERIMENTS

In this section, we present a series of experiments to validate the proposed dynamic-precision strategy in pre-training tasks of LLMs.

5.1 PRE-TRAINING LLAMA-2 WITH ACTIVATION QUANTIZATION

Experiment setup. We pre-train LLaMA-2 models (Touvron et al., 2023) with parameter sizes ranging from 60M to 1.3B. Following existing experimental settings (Zhao et al., 2024; Kong et al., 2025b), we use the C4-en dataset (Raffel et al., 2020), which is designed for large-scale language model pre-training. The models are trained with bfloat16 precision for both forward and backward propagation. In line with the method of (Jiang et al., 2022), we introduce a token-wise quantization (Ramasinghe et al., 2025a) for the activations of each linear operator to low precision for backpropagation. We then apply our proposed dynamic-precision method, controlled by both the GNMR and Δ-GNMR metrics, and compare it against fixed-precision quantization. The detailed experimental set up is provided in Appendix E.1.

Experiment results. Table 1 reports the validation perplexity for our dynamic-precision strategies using different GNMR/Δ-GNMR thresholds. A key finding is that dynamic 4-bit and 8-bit activation quantization delivers performance on par with fixed 8-bit quantization. This stands in stark contrast to fixed 4-bit quantization, which fails to converge, thereby highlighting the critical role of adaptivity. The efficacy of our method is further solidified for the 1.3B model, where the GNMR-driven dynamic policy 4-bit and 8-bit with results in less than a 1% increase in validation perplexity compared to standard BF16 training, showcasing its promise in large-scale applications.

5.2 PRE-TRAINING LLAMA-2 WITH DEEPSEEK-STYLE MIXED-PRECISION STRATEGY

Experiment setup. We employ the pre-training tasks of the LLaMA-2 model, adopting the mixed-precision strategy from DeepSeek V3 (Liu et al., 2024) on the C4-en dataset (Raffel et al., 2020). To maintain alignment with our dynamic-precision approach guided by GNMR and Δ-GNMR, we preserve identical precision settings for operators that utilize 16-bit and 32-bit precision in DeepSeek, while implementing distinct low-precision strategies for the remaining operators. To mitigate potential inconsistencies arising from mixed-precision operators, we treat each transformer block as an integrated unit, thereby ensuring a consistent precision strategy across the architecture. During training with our dynamic precision framework, we set the maximum number of high-precision layers to $maxO = 6$, corresponding to at most 25% of the total layers operating in high precision. We evaluate our proposed precision strategies—specifically the 8-bit/6-bit and 16-bit/8-bit configurations—against their respective fixed-precision counterparts: fixed 8-bit training and fixed 16-bit training. Complete experimental configurations and implementation details are documented in Appendix E.2.

Experiment results. Table 2 presents the validation perplexity during training across different precision strategies. The results demonstrate that both dynamic 6-bit/8-bit and 8-bit/16-bit configurations achieve performance comparable to their fixed-precision counterparts—fixed 8-bit and fixed 16-bit training, respectively.

6 CONCLUSION AND FUTURE WORK

This paper presents a dynamic precision strategy for the pre-training of Large Language Models (LLMs). We introduce the GNMR and Δ-GNMR metrics to characterize the stability of operators or transformer layers and adjust the precision in real-time to mitigate instability. Experimental results validate that the proposed strategy achieves computational efficiency comparable to low-precision training while maintaining performance on par with high-precision setups, thereby enabling the use of low-bit strategies in LLM pre-training. The proposed strategy may be further improved by engineering optimizations that reduce casting and routing overheads, which we leave as future work.

ACKNOWLEDGMENTS

The research is supported by Huawei Technologies Co., Ltd.. This work is supported by the National Natural Science Foundation of China (No. 12288101, 92370121, 12301392, W2441021) and National Key Research and Development Program of China (No. 2024YFA1012902). This work is also supported by AI for Science Institute, Beijing, China.

REFERENCES

Felix Abecassis, Anjulie Agrusa, Dong Ahn, Jonah Alben, Stefania Alborghetti, Michael Andersch, Sivakumar Arayandi, Alexis Bjorlin, Aaron Blakeman, Evan Briones, et al. Pretraining large language models with nvfp4. *arXiv preprint arXiv:2509.25149*, 2025.

Saleh Ashkboos, Bram Verhoef, Torsten Hoefler, Evangelos Eleftheriou, and Martino Dazzi. Efqat: An efficient framework for quantization-aware training. *arXiv preprint arXiv:2411.11038*, 2024.

Yelysei Bondarenko, Riccardo Del Chiaro, and Markus Nagel. Low-rank quantization-aware training for llms. *arXiv preprint arXiv:2406.06385*, 2024.

Andy Brock, Soham De, Samuel L Smith, and Karen Simonyan. High-performance large-scale image recognition without normalization. In *International conference on machine learning*, pp. 1059–1071. PMLR, 2021.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Mengzhao Chen, Wenqi Shao, Peng Xu, Jiahao Wang, Peng Gao, Kaipeng Zhang, and Ping Luo. Efficientqat: Efficient quantization-aware training for large language models. *arXiv preprint arXiv:2407.11062*, 2024a.

Shimao Chen, Zirui Liu, Zhiying Wu, Ce Zheng, Peizhuang Cong, Zihan Jiang, Yuhan Wu, Lei Su, and Tong Yang. Int-flashattention: Enabling flash attention for int8 quantization. *arXiv preprint arXiv:2409.16997*, 2024b.

Herman Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *The Annals of Mathematical Statistics*, pp. 493–507, 1952.

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Yu Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*, 2024.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in neural information processing systems*, 35: 30318–30332, 2022.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, 36:10088–10115, 2023.

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, et al. Glam: Efficient scaling of language models with mixture-of-experts. In *International conference on machine learning*, pp. 5547–5569. PMLR, 2022.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

Maxim Fishman, Brian Chmiel, Ron Banner, and Daniel Soudry. Scaling fp8 training to trillion-token llms. *arXiv preprint arXiv:2409.12517*, 2024.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

Tianjin Huang, Haotian Hu, Zhenyu Zhang, Gaojie Jin, Xiang Li, Li Shen, Tianlong Chen, Lu Liu, Qingsong Wen, Zhangyang Wang, et al. Stable-spam: How to train in 4-bit more stably than 16-bit adam. *arXiv preprint arXiv:2502.17055*, 2025a.

Tianjin Huang, Ziquan Zhu, Gaojie Jin, Lu Liu, Zhangyang Wang, and Shiwei Liu. Spam: Spike-aware adam with momentum reset for stable llm training. *arXiv preprint arXiv:2501.06842*, 2025b.

Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Dehao Chen, Mia Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V Le, Yonghui Wu, et al. Gpipe: Efficient training of giant neural networks using pipeline parallelism. *Advances in neural information processing systems*, 32, 2019.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

Jinda Jia, Cong Xie, Hanlin Lu, Daoce Wang, Hao Feng, Chengming Zhang, Baixi Sun, Haibin Lin, Zhi Zhang, Xin Liu, et al. Sdp4bit: Toward 4-bit communication quantization in sharded data parallelism for llm training. *Advances in Neural Information Processing Systems*, 37:8734–8759, 2024.

Guoqing Jiang, Jinlong Liu, Zixiang Ding, Lin Guo, and Wei Lin. Accelerating large batch training via gradient signal to noise ratio (gsnr). *arXiv preprint arXiv:2309.13681*, 2023.

Ziyu Jiang, Xuxi Chen, Xueqin Huang, Xianzhi Du, Denny Zhou, and Zhangyang Wang. Back razor: Memory-efficient transfer learning by self-sparsified backpropagation. *Advances in neural information processing systems*, 35:29248–29261, 2022.

Boao Kong, Xu Huang, Yuqi Xu, Yixuan Liang, Bin Wang, and Kun Yuan. Clapping: Removing per-sample storage for pipeline parallel distributed optimization with communication compression. *arXiv preprint arXiv:2509.19029*, 2025a.

Boao Kong, Junzhu Liang, Yuxi Liu, Renjia Deng, and Kun Yuan. Cr-net: Scaling parameter-efficient training with cross-layer low-rank structure. *arXiv preprint arXiv:2509.18993*, 2025b.

Lorenz Kummer, Kevin Sidak, Tabea Reichmann, and Wilfried Gansterer. Adaptive precision training (adapt): A dynamic quantized training approach for dnns. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pp. 559–567. SIAM, 2023.

Joonhyung Lee, Jeongin Bae, Byeongwook Kim, Se Jung Kwon, and Dongsoo Lee. To fp8 and back again: Quantifying reduced precision effects on llm training stability. *arXiv preprint arXiv:2405.18710*, 2024.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of machine learning and systems*, 6: 87–100, 2024.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. Llm-qat: Data-free quantization aware training for large language models. *arXiv preprint arXiv:2305.17888*, 2023.

Sam McCandlish, Jared Kaplan, Dario Amodei, and OpenAI Dota Team. An empirical model of large-batch training. *arXiv preprint arXiv:1812.06162*, 2018.

Harshitha Menon, Michael O Lam, Daniel Osei-Kuffuor, Markus Schordan, Scott Lloyd, Kathryn Mohror, and Jeffrey Hittinger. Adapt: Algorithmic differentiation applied to floating-point precision tuning. In *SC18: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 614–626. IEEE, 2018.

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

Paulius Micikevicius, Dusan Stosic, Neil Burgess, Marius Cornea, Pradeep Dubey, Richard Grisenthwaite, Sangwon Ha, Alexander Heinecke, Patrick Judd, John Kamalu, et al. Fp8 formats for deep learning. *arXiv preprint arXiv:2209.05433*, 2022.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pp. 1310–1318. Pmlr, 2013.

Houwen Peng, Kan Wu, Yixuan Wei, Guoshuai Zhao, Yuxiang Yang, Ze Liu, Yifan Xiong, Ziyue Yang, Bolin Ni, Jingcheng Hu, et al. Fp8-lm: Training fp8 large language models. *arXiv preprint arXiv:2310.18313*, 2023.

Sergio P Perez, Yan Zhang, James Briggs, Charlie Blake, Josh Levy-Kramer, Paul Balanca, Carlo Luschi, Stephen Barlow, and Andrew William Fitzgibbon. Training and inference of large language models using 8-bit floating point. *arXiv preprint arXiv:2309.17224*, 2023.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–16. IEEE, 2020.

Sameera Ramasinghe, Thalaiyasingam Ajanthan, Gil Avraham, Yan Zuo, and Alexander Long. Beyond top-k: Structured sparsification for compression in pipeline parallel. In *ICLR 2025 Workshop on Modularity for Collaborative, Decentralized, and Continual Deep Learning*, 2025a.

Sameera Ramasinghe, Thalaiyasingam Ajanthan, Gil Avraham, Yan Zuo, and Alexander Long. Protocol models: Scaling decentralized training with communication-efficient model parallelism. *arXiv preprint arXiv:2506.01260*, 2025b.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

Sho Takase, Shun Kiyono, Sosuke Kobayashi, and Jun Suzuki. Spike no more: Stabilizing the pre-training of large language models. *arXiv preprint arXiv:2312.16903*, 2023.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Xinyi Wan, Penghui Qi, Guangxing Huang, Min Lin, and Jialin Li. Pipeoffload: Improving scalability of pipeline parallelism with memory optimization. *arXiv preprint arXiv:2503.01328*, 2025.

Guanhua Wang, Heyang Qin, Sam Ade Jacobs, Connor Holmes, Samyam Rajbhandari, Olatunji Ruwase, Feng Yan, Lei Yang, and Yuxiong He. Zero++: Extremely efficient collective communication for giant model training. *arXiv preprint arXiv:2306.10209*, 2023.

Ruizhe Wang, Yeyun Gong, Xiao Liu, Guoshuai Zhao, Ziyue Yang, Baining Guo, Zhengjun Zha, and Peng Cheng. Optimizing large language model training using fp4 quantization. *arXiv preprint arXiv:2501.17116*, 2025.

Shibo Wang and Pankaj Kanwar. Bfloat16: The secret to high performance on cloud tpus. *URL https://cloud. google. com/blog/products/ai-machine-learning/bfloat16-the-secret-to-high-performance-on-cloud-tpus*, 2019.

Xiuying Wei, Yunchen Zhang, Yuhang Li, Xiangguo Zhang, Ruihao Gong, Jinyang Guo, and Xianglong Liu. Outlier suppression+: Accurate quantization of large language models by equivalent and optimal shifting and scaling. *arXiv preprint arXiv:2304.09145*, 2023.

Haocheng Xi, Han Cai, Ligeng Zhu, Yao Lu, Kurt Keutzer, Jianfei Chen, and Song Han. Coat: Compressing optimizer states and activation for memory-efficient fp8 training. *arXiv preprint arXiv:2410.19313*, 2024.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International conference on machine learning*, pp. 38087–38099. PMLR, 2023.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International conference on machine learning*, pp. 10524–10533. PMLR, 2020.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in neural information processing systems*, 35:27168–27183, 2022.

Zhongzhi Yu, Yonggan Fu, Shang Wu, Mengquan Li, Haoran You, and Yingyan Lin. Ldp: Learnable dynamic precision for efficient deep neural network training and inference. *arXiv preprint arXiv:2203.07713*, 2022.

Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. Galore: Memory-efficient llm training by gradient low-rank projection. *arXiv preprint arXiv:2403.03507*, 2024.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023.

Appendix

A Related works

In this section, we review the literature most relevant to our proposed method.

Mixed-precision training and numerical formats for LLM pre-training. Mixed-precision training (MPT) has evolved as a practical strategy to trade arithmetic precision for enhanced throughput and memory efficiency, beginning with FP16 approaches that combine loss scaling and selective FP32 accumulations to preserve accuracy while optimizing compute on modern accelerators like GPUs/TPUs (Micikevicius et al., 2017). The adoption of bfloat16 (BF16) improved dynamic-range robustness over IEEE FP16, becoming a default for large-scale language model pre-training (Wang & Kanwar, 2019). In recent years, the field has shifted toward sub-16-bit formats, particularly FP8 (including E4M3 and E5M2), which demonstrate competitive convergence in CNN/RNN/Transformer workloads, including LLMs up to hundreds of billions parameters, under careful recipes (Micikevicius et al., 2022). NVIDIA's Transformer Engine (TE) productized FP8 training with mixed-precision primitives like delayed scaling and per-tensor format selection, making it practical on Hopper/Blackwell GPUs (Perez et al., 2023), with system-level reports showing substantial throughput and memory gains over BF16 baselines (Peng et al., 2023). Beyond FP8, vendors now expose FP4/NVFP4 formats for portions of the training graph on Blackwell-class hardware, requiring tighter scaling and outlier handling for stability (Abecassis et al., 2025). Additionally, alternative numerical formats like COAT, which redefine resolution and distribution, have also achieved promising results in training scenarios (Xi et al., 2024).

Low-precision optimizers, systems implementation, and quantization for LLM training and inference. Low-precision methods are central to LLM pre-training and inference: they improve memory efficiency and throughput while handling outliers and operational fragility. In pre-training, optimizer states dominate memory; 8-bit optimizers use block-wise quantization of first/second moments with on-the-fly (de)quantization in registers, typically matching FP32 accuracy while reducing memory and wall-clock via bitsandbytes (Dettmers et al., 2023). At the distributed layer, mixed-precision is paired with data/tensor/pipeline parallelism (e.g., Megatron-LM), ZeRO/FSDP-style sharding, and communication-aware variants (e.g., ZeRO++) (Shoeybi et al., 2019); communication-time quantization—such as 4-bit gradient/weight exchange for ShardedDP with accuracy-preserving corrections—further lowers cost (Jia et al., 2024). For inference and fine-tuning, advances include outlier-aware GPT3.int8 (Dettmers et al., 2022); SmoothQuant, which shifts activation outliers to weights to enable W8A8 kernels (Xiao et al., 2023); fast one-shot 3–4-bit GPTQ (Frantar et al., 2022); salient-weight-preserving AWQ for robust 4-bit inference (Lin et al., 2024); and layer-wise PTQ with distillation in ZeroQuant (Yao et al., 2022). In fine-tuning, QLoRA trains adapters over frozen 4-bit bases (NF4 + paged optimizers), enabling 65B model adaptation on a single 48 GB GPU (Dettmers et al., 2023); training-time quantization continues to progress—LLM-QAT (Liu et al., 2023), LR-QAT (Bondarenko et al., 2024), EfficientQAT (Chen et al., 2024a; Ashkboos et al., 2024), and PyTorch QAT toolchains often exceed PTQ at the same bitwidths (Liu et al., 2023). Although aimed primarily at inference/fine-tuning, these works foreground activation outliers and fragile layers/operations; mitigations (outlier smoothing/suppression, per-channel scaling, salient-weight protection) inform operator-/layer-aware precision control during training (Wei et al., 2023). Finally, cross-device communication compression and quantization are increasingly used to curb training communication overheads (Huang et al., 2019; Wan et al., 2025; Ramasinghe et al., 2025b; Kong et al., 2025a).

Stability challenges and precision considerations in LLM pre-training Large-scale pre-training can suffer rare yet catastrophic loss spikes, linked to surges in gradient norms and sublayer spectral norms; recent work bounds such growth via architectural and initialization changes (Takase et al., 2023). Core stabilization includes gradient-norm clipping—originating in RNNs and now standard for Transformers—with Adaptive Gradient Clipping (AGC) clipping by the gradient-to-parameter-norm ratio (Pascanu et al., 2013). The shift from post-LN to Pre-LN, together with RMSNorm and DeepNorm, further regularizes update magnitudes in deep stacks (Xiong et al., 2020). These issues intensify under low precision: FP8 pipelines must address reduced dynamic range (e.g., Transformer Engine's delayed scaling with running amax histories for per-tensor scales), and emerging best practices favor per-block scaling to blunt outliers. New FP8 recipes (e.g., unit-scaling, COAT) stress that

quantization granularity and optimizer/activation precision are pivotal to avoid underflow/overflow and keep optimization stable (Xi et al., 2024). Parallel work on gradient-noise-scale modeling illuminates training regimes and critical batch sizes; step-wise gradient statistics, though not designed for precision control, can guide adaptive decisions (McCandlish et al., 2018). Overall, monitoring gradient statistics is crucial both for stability and as signals for precision-adjustment strategies.

Dynamic / operator-wise precision selection. Existing research has investigated dynamic precision policies that adaptively adjust numerical bitwidths during model training. AdaPT employs information-theoretic criteria to determine per-layer fixed-point precision across training epochs, while other approaches dynamically adjust floating-point precision based on empirical requirements (Kummer et al., 2023). More recently, CoMP introduces a convergence-aware operator-wise mixed precision scheme with two-stage adjustments at both epoch and batch levels (Xi et al., 2024). Learnable Dynamic Precision (LDP) further advances this direction by formulating both temporal and spatial precision configurations as trainable parameters to optimize the compute-accuracy tradeoff (Yu et al., 2022). In contrast to scaling-factor heuristics or epoch-level schedules, we propose the Gradient Norm Ratio (GNMR)—defined as the ratio between the current step's gradient norm and the running average of historical norms—computed separately for each matrix operator. This approach elevates precision only when GNMR exceeds a predetermined threshold (indicating high-risk optimization steps) and reduces it otherwise, thereby aligning precision allocation with optimization risk rather than relying solely on magnitude distributions. The GNMR strategy complements existing stabilization techniques including AGC, normalization methods, and delayed scaling protocols (Huang et al., 2025b;a). Preliminary implementations demonstrate that precision gating based on gradient excursions effectively mitigates mixed-precision training instability while preserving the memory efficiency and computational throughput advantages of FP8/FP4 regimes.

B Additional details for the operator-level modeling.

This section complements the operator-level paradigm introduced in Eqs. (1)–(4). We first clarify how the *nest–parallel* abstraction instantiates canonical Transformer components, then give a compact backward pass that matches this abstraction, and finally note several implementation details (residual/normalization/parameter sharing). Throughout, we avoid any architectural commitment beyond the operator view, so the descriptions apply to both encoder-only and decoder-style models (Vaswani et al., 2017; Brown et al., 2020).

Compatibility with Transformer layers. A depth-U Transformer stack corresponds to the composition of nested operators $\mathcal{N}^{[1]}, \ldots, \mathcal{N}^{[U]}$ acting on the same input state within each layer (Eq. (2)). Inside one layer u, sibling operators $f^{[u,k]}$ run in parallel on $y^{[u-1]}$ (Eq. (3)), and an aggregator $z^{[u,0]}$ merges their outputs into $y^{[u]}$ (Eq. (4)). This pattern captures multi-head attention (parallel heads), feed-forward modules (one or multiple experts), and residual connections (the aggregator includes the skip path).

Multi-head self-attention (MHA). Let H be the layer input. The head-wise projections are parallel operators

$$Q = HW_Q^{[u]}, \qquad K = HW_K^{[u]}, \qquad V = HW_V^{[u]}. \tag{11}$$

The attention aggregator then maps (Q, K, V) and the residual input H to the layer output:

$$z_{\text{att}}^{[u]}\Big(Q, K, V;\, W_O^{[u]},\, H\Big) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V\, W_O^{[u]} + H. \tag{12}$$

Stacking multiple heads fits by letting the sibling set contain per-head projections and attention kernels, with the aggregator concatenating head outputs and applying $W_O^{[u]}$.

Feed-forward networks (FFN) and MoE. A standard two-layer FFN with residual can be represented with $K_u = 1$ as

$$z^{[u,0]}\big(z^{[u,1]};\, \theta^{[u,0]},\, y\big) = y + W_2\, \varphi(W_1\, \text{LN}(y))\,. \tag{13}$$

Mixture-of-experts (MoE) variants use $K_u > 1$ sibling experts $f^{[u,k]}$ and an aggregator that gates or mixes expert outputs, e.g.,

$$z^{[u,0]}\big(z^{[u,1]},\dots,z^{[u,K_u]};\,\theta^{[u,0]},\,y\big) = \sum_{k=1}^{K_u} \alpha_k(y)\, z^{[u,k]}, \quad \sum_{k=1}^{K_u} \alpha_k(y) = 1, \tag{14}$$

which covers router-based and top-k mixing used in modern MoE systems (Du et al., 2022; Fedus et al., 2022; Dai et al., 2024). Variants with concatenation followed by projection are obtained by letting $z^{[u,0]}$ first concatenate $\big(z^{[u,1]},\dots,z^{[u,K_u]}\big)$ and then apply a linear map.

Backpropagation in the operator view. Denote the upstream gradient at layer u by $\bar{y}^{[u]} := \partial L/\partial y^{[u]}$, and define $\bar{z}^{[u,k]} := \partial L/\partial z^{[u,k]}$. The aggregator routes sensitivity to each sibling output via

$$\bar{z}^{[u,k]} = \left(\frac{\partial z^{[u,0]}}{\partial z^{[u,k]}}\right)^{\top} \bar{y}^{[u]}, \quad k = 1,\dots,K_u. \tag{15}$$

Parameter gradients follow directly from the local Jacobians:

$$\nabla_{\theta^{[u,k]}} L = \left(\frac{\partial f^{[u,k]}}{\partial \theta^{[u,k]}}\right)^{\top} \bar{z}^{[u,k]}, \quad \nabla_{\theta^{[u,0]}} L = \left(\frac{\partial z^{[u,0]}}{\partial \theta^{[u,0]}}\right)^{\top} \bar{y}^{[u]}. \tag{16}$$

The gradient w.r.t. the layer input $y^{[u-1]}$ sums the sibling and aggregator contributions:

$$\bar{y}^{[u-1]} = \sum_{k=1}^{K_u} \left(\frac{\partial f^{[u,k]}}{\partial y}\right)^{\top} \bar{z}^{[u,k]} + \left(\frac{\partial z^{[u,0]}}{\partial y}\right)^{\top} \bar{y}^{[u]}. \tag{17}$$

These expressions are shape-consistent with Eq. (4) and specialize to standard Transformer derivatives: for attention, the aggregator derivatives backpropagate through the $\mathrm{softmax}$ and output projection; for FFN/MoE, they reduce to linear maps and elementwise nonlinearity Jacobians.

Practical notes for the two-column setting. (i) Residual paths are naturally included by passing $y^{[u-1]}$ to the aggregator, as in Eqs. (12)–(13); layer normalization can be modeled either as a sibling $f^{[u,k]}$ or inside the aggregator, depending on implementation needs. (ii) When documenting models in two-column format, prefer short, split equations (as above) and avoid long in-line tensor lists; if necessary, use `aligned` with line breaks. (iii) All parentheses have been typeset with $(\cdot)$ and $\{\cdot\}$ to ensure robust sizing without relying on `\big`-class delimiters. (iv) The operator view remains agnostic to parameter sharing and masking (e.g., causal masks): both appear as linear operators inside $f^{[u,k]}$ or the aggregator.

In summary, the nest–parallel operator modeling *exactly* recovers MHA, FFN, and MoE while providing a concise interface for composing new blocks and for analyzing training dynamics at the operator level. This makes it a convenient front end for the GNMR-based, operator-level mixed-precision strategies considered in the main text.

C Assumptions and detailed proofs

In Section 4, we analyze the proposed dynamic-precision strategy. This appendix first states the standing assumptions used by Theorems 4.1-4.2 (Appendix C.1). We then provide: (i) a sufficient condition that justifies the one-step persistence parameter (Appendix C.2), (ii) the proof of the one-step descent gain result (Appendix C.3), and (iii) the proof of the exceedance-probability bound motivating threshold decay (Appendix C.4).

C.1 Assumptions

Here we present the assumption in the theoretical analysis. The first two assumptions are from the standard analysis of stochastic gradient descend.

Assumption C.1 (Stochastic gradient noise)**.** We assume that the stochastic gradient g_t at step t is an unbiased estimator of the full gradient $\nabla F(\theta_t)$ with bounded variance. Specifically, there exists a constant $\sigma_{\text{sgd}} > 0$ such that

$$g_t = \nabla F(\theta_t) + \xi_t, \quad \xi_t \sim \mathcal{N}(0, \sigma_{\text{sgd}}^2 I),$$

where ξ_t is independent of $\mathcal{F}_t$.

Moreover, for each operator $[u,k]$, the stochastic gradient $g_t^{[u,k]}$ is an unbiased estimator of the partial gradient $\nabla_{\theta^{[u,k]}} F(\theta_t)$ with bounded variance. That is,

$$g_t^{[u,k]} = \nabla_{\theta^{[u,k]}} F(\theta_t) + \xi_t^{[u,k]}, \quad \xi_t^{[u,k]} \sim \mathcal{N}(0, \sigma_{u,k}^2 I),$$

and the noise terms $\xi_t^{[u,k]}$ are independent across time steps t and independent of $\mathcal{F}_t$.

Assumption C.2 (L-smooth)**.** There exists $L_\nabla > 0$ such that for any two parameter vectors θ, θ',

$$\left\|\nabla F(\theta') - \nabla F(\theta)\right\| \le L_\nabla \|\theta' - \theta\|.$$

With Assumption C.2, it holds for any $[u,k]$ and any parameters θ, θ' that

$$\left\|\nabla_{\theta^{[u,k]}} F(\theta') - \nabla_{\theta^{[u,k]}} F(\theta)\right\| \le L_\nabla \|\theta' - \theta\|.$$

The following assumption characterize the mixed-precision gradient evaluation:

Assumption C.3 (Mixed-precision gradient error)**.** In mixed-precision training, the gradient update is computed as $\widehat{g}_t = g_t + \varepsilon_t$, where ε_t represents the error induced by precision reduction. This error term is zero-mean, independent of $\mathcal{F}_t$, and satisfies the following high-probability bound:

$$\mathbb{P}(\|\varepsilon_t\| \le \iota_q) \ge 1 - \delta_q,$$

for constants $\iota_q > 0$ and $\delta_q > 0$.

Moreover for $[u,k] \in \mathfrak{B}$, denote $\Delta v^b(\rho_t^{[u,k]})$ as the difference of the expectation $\mathbb{E}[\|\varepsilon_t\|^2]$ under dynamic-precision training and low-precision training with $\text{GNMR}_t(\theta^{[u,k]}) = \rho_t^{[u,k]}$. See Assumption C.9 in Appendix for more details.

The analysis is also guided by the assumption that an operator identified as high-risk at a given step has a significant probability of remaining high-risk in the subsequent step.

Assumption C.4 (One-step persistence)**.** For each operator $[u,k] \in \mathfrak{B}$, there exists $\pi^b \in (0,1]$ such that

$$\mathbb{P}\big(\text{GNMR}_{t+1}(\theta^{[u,k]}) \ge \tau^b \mid H_t(\alpha)\big) \ge \pi^b.$$

Such a persistence condition can be justified under benign mixed-precision/SGD noise. Appendix C.2 derives an explicit lower bound that provides a sufficient condition for this assumption and clarifies the meaning of π_b.

C.2 Justifying Assumption C.4: one-step predictability under benign noise

This subsection provides a sufficient condition for Assumption C.4 by lower-bounding $\mathbb{P}(\text{GNMR}_{t+1}^b \ge \alpha \mid \mathcal{F}_t)$ on the event $\{\text{GNMR}_t^b \ge \alpha\}$. Theorem 4.1 only requires Assumption C.4; the result below is included to make the persistence parameter π_b interpretable and to support the practical plausibility of short-horizon risk persistence in mixed-precision LLM pre-training.

To begin with, we introduce the following notations to represent the metric AGMN and GNMR of step $t \neq 1$ as follows:

$$m_t^{[u,k]} = \frac{1}{t-1} \sum_{\tau=1}^{t-1} \left\|g_\tau^{[u,k]}\right\|, \quad \rho_t^{[u,k]} = \frac{\|g_t^{[u,k]}\|}{m_t^{[u,k]}}. \tag{18}$$

And we also denote $u_t^{[u,k]} := g_t^{[u,k]} / \|g_t^{[u,k]}\|$ as the normalized gradient of the operator $z^{[u,k]}$. Then, we define the set of high risk operators for step t with GNMR threshold $\alpha > 1$ as follows:

$$H_t(\alpha) := \left\{(t,u,k) \middle| \rho_t^{[u,k]} \ge \alpha\right\}. \tag{19}$$

The following lemma is useful for the subsequent analysis.

Lemma C.5 (Arithmetic-mean dynamics at a operator). *For any $u = 1, 2, \cdots, U$ and $k = 0, 1, 2, \cdots, K_u$ and a given $\alpha > 1$, if $(t, u, k) \in H_t(\alpha)$, it holds that:*

$$\alpha m_{t+1}^{[u,k]} \;\le\; \|g_t^{[u,k]}\|\left(1 + \frac{\alpha - 1}{t}\right). \tag{20}$$

Proof. By the definition of the GNMR metric, it holds that:

$$m_{t+1}^{[u,k]} = m_t^{[u,k]} + \frac{\|g_t^{[u,k]}\| - m_t^{[u,k]}}{t} = m_t^{[u,k]}\left(1 + \frac{\rho_t^{[u,k]} - 1}{t}\right), \tag{21}$$

Multiply both sides of (21) by α:

$$\alpha m_{t+1}^{[u,k]} = \alpha m_t^{[u,k]} + \frac{\alpha}{t}\left(\|g_t^{[u,k]}\| - m_t^{[u,k]}\right) = \|g_t^{[u,k]}\|\left(\frac{\alpha}{\rho_t^{[u,k]}} + \frac{\alpha}{t}\left(1 - \frac{1}{\rho_t^{[u,k]}}\right)\right).$$

Since $(t, u, k) \in H_t(\alpha)$ implies $\rho_t^{[u,k]} \ge \alpha$, we have $\frac{\alpha}{\rho_t^{[u,k]}} \le 1$ and $1 - \frac{1}{\rho_t^{[u,k]}} \le 1 - \frac{1}{\alpha} = \frac{\alpha-1}{\alpha}$. Therefore,

$$\alpha m_{t+1}^{[u,k]} \;\le\; \|g_t^{[u,k]}\|\left(1 + \frac{\alpha - 1}{t}\right). \tag{22}$$

Thus we finish the proof of this lemma. □

Lemma C.6 (One-dimensional sufficient condition). *Assume that $(t, u, k) \in H_t(\alpha)$,* i.e. *, $\rho_t^{[u,k]} \ge \alpha$. Then it holds that $\rho_{t+1}^{[u,k]} \ge \alpha$ if*

$$g_{t+1}^{[u,k]} \cdot u_t^{[u,k]} \;\ge\; \left(1 + \frac{\alpha - 1}{t}\right)\|g_t^{[u,k]}\|. \tag{23}$$

Proof. As $\rho_t^{[u,k]} \ge \alpha$, then $u_t^{[u,k]} := g_t^{[u,k]}/\|g_t^{[u,k]}\|$ is well-defined. Then from Cauchy-Schwarz inequality and the fact that $u_t^{[u,k]}$ is normalized, it holds that

$$\|g_{t+1}^{[u,k]}\| \;\ge\; g_{t+1}^{[u,k]} \cdot u_t^{[u,k]}. \tag{24}$$

If (23) holds, then using (24) gives

$$\|g_{t+1}^{[u,k]}\| \;\ge\; g_{t+1}^{[u,k]} \cdot u_t^{[u,k]} \;\ge\; \left(1 + \frac{\alpha - 1}{t}\right)\|g_t^{[u,k]}\|.$$

Comparing with (20), we obtain

$$\|g_{t+1}^{[u,k]}\| \;\ge\; \alpha m_{t+1}^{[u,k]}.$$

Divide both sides by $m_{t+1}^{[u,k]} > 0$ to conclude $\rho_{t+1}^{[u,k]} = \|g_{t+1}^{[u,k]}\|/m_{t+1}^{[u,k]} \;\ge\; \alpha$. □

The following lemma characterize the directional drift along the operator direction.

Lemma C.7 (Directional drift along the operator direction). *Under Assumption C.2, for any $[u, k]$,*

$$g_{t+1}^{[u,k]} \cdot u_t^{[u,k]} \;\ge\; \left\|g_t^{[u,k]}\right\| \;-\; L_\nabla\,\eta\,\|\widehat{g}_t\| \;+\; Z_t^{[u,k]}, \tag{25}$$

where $Z_t^{[u,k]} := \left(\xi_{t+1}^{[u,k]} - \xi_t^{[u,k]}\right) \cdot u_t^{[u,k]}$. Conditionally on $\mathcal{F}_t$ and $u_t^{[u,k]}$, $\xi_{t+1}^{[u,k]} \cdot u_t^{[u,k]} \sim \mathcal{N}(0, \sigma_{u,k}^2)$ and is independent of $\xi_t^{[u,k]}$.

Proof. By the L-smooth of F from Assumption C.2, it holds that:

$$\nabla_{\theta^{[u,k]}} F(\theta_{t+1}) \cdot u_t^{[u,k]} \ge \nabla_{\theta^{[u,k]}} F(\theta_t) \cdot u_t^{[u,k]} \;-\; L_\nabla\,\|\theta_{t+1} - \theta_t\| \ge \nabla_{\theta^{[u,k]}} F(\theta_t) \cdot u_t^{[u,k]} \;-\; L_\nabla\,\eta\,\|\widehat{g}_t\|. \tag{26}$$

Then it can be obtained that:

$$\begin{aligned} g_{t+1}^{[u,k]} \cdot u_t^{[u,k]} =& \nabla_{\theta^{[u,k]}} F(\theta_{t+1}) \cdot u_t^{[u,k]} + \xi_{t+1}^{[u,k]} \cdot u_t^{[u,k]} \\ \geq& \left(\nabla_{\theta^{[u,k]}} F(\theta_t) \cdot u_t^{[u,k]} \right) - L_\nabla \eta \|\widehat{g}_t\| + \xi_{t+1}^{[u,k]} \cdot u_t^{[u,k]} \\ =& \left(g_t^{[u,k]} - \xi_t^{[u,k]} \right) \cdot u_t^{[u,k]} - L_\nabla \eta \|\widehat{g}_t\| + \xi_{t+1}^{[u,k]} \cdot u_t^{[u,k]} \\ =& \underbrace{g_t^{[u,k]} \cdot u_t^{[u,k]}}_{= \|g_t^{[u,k]}\|} - L_\nabla \eta \|\widehat{g}_t\| + \underbrace{\left(\xi_{t+1}^{[u,k]} - \xi_t^{[u,k]} \right) \cdot u_t^{[u,k]}}_{= Z_t^{[u,k]}}. \end{aligned} \tag{27}$$

Then Eq. (25) is satisfied.

Moreover, as $u_t^{[u,k]}$ is $\mathcal{F}_t$-measurable and $\xi_{t+1}^{[u,k]}$ is independent of $\mathcal{F}_t$. With the fact that $\|u_t^{[u,k]}\| = 1$, it can be obtained that

$$\xi_{t+1}^{[u,k]} \cdot u_t^{[u,k]} \,\big|\, \mathcal{F}_t \;\sim\; \mathcal{N}(0, \sigma_{u,k}^2). \tag{28}$$

As $\xi_{t+1}^{[u,k]}$ is independent of $(\xi_s^{[u,k]})_{s \leq t}$ and of $\mathcal{F}_t$, the projection $\xi_{t+1}^{[u,k]} \cdot u_t^{[u,k]}$ is independent of both $\xi_t^{[u,k]}$ and $\mathcal{F}_t$. Thus, we obtained all the conclusions of this lemma. □

Then, we present the following proposition, which illustrates the high probability for the high-risk operator maintain the high GNMR in subsequent steps.

Proposition C.8 (Sufficient condition for one-step persistence)**.** *Let the high-probability events*

$$\mathcal{E}_q := \{\|\varepsilon_t\| \leq \iota_q\}, \qquad \mathcal{E}_s := \{\|\xi_t\| \leq \iota_s\}$$

satisfy $\mathbb{P}(\mathcal{E}_q) \geq 1 - \delta_q$ *and* $\mathbb{P}(\mathcal{E}_s) \geq 1 - \delta_s$*. We also assumpe that* $(t, u, k) \in H_t(\alpha)$*. Then conditioning on* $\mathcal{F}_t$*,* $H_t^{[u,k]}$*,* $\mathcal{E}_q$ *and* $\mathcal{E}_s$*, it holds that*

$$\mathbb{P}\left(\rho_{t+1}^{[u,k]} \geq \alpha \middle| \mathcal{F}_t, \mathcal{E}_q, \mathcal{E}_s \right) \geq \Phi\left(- \frac{\left(\frac{\alpha-1}{t}\right) \|g_t^{[u,k]}\| + L_\nabla \eta \left(\|g_t\| + \iota_q \right) + \iota_s}{\sigma_{u,k}} \right), \tag{29}$$

where Φ *is the standard normal distribution cumulative distribution function (CDF) of standard normal distribution and* $\|g_t\|$ *is the full-model gradient norm. Consequently, without conditioning on* $\mathcal{E}_q, \mathcal{E}_s$*, it holds that*

$$\mathbb{P}\left(\rho_{t+1}^{[u,k]} \geq \alpha \middle| \mathcal{F}_t \right) \geq (1 - \delta_q)(1 - \delta_s) \cdot \Phi\left(- \frac{\left(\frac{\alpha-1}{t}\right) \|g_t^{[u,k]}\| + L_\nabla \eta \left(\|g_t\| + \iota_q \right) + \iota_s}{\sigma_{u,k}} \right). \tag{30}$$

Proof. From Lemma C.5, $\rho_{t+1}^{[u,k]} \geq \alpha$ yields that

$$\alpha \, m_{t+1}^{[u,k]} \;\leq\; \|g_t^{[u,k]}\| \left(1 + \frac{\alpha - 1}{t} \right). \tag{31}$$

Hence, a *sufficient* condition for $\rho_{t+1}^{[u,k]} \geq \alpha$ is

$$\|g_{t+1}^{[u,k]}\| \;\geq\; \alpha \, m_{t+1}^{[u,k]} \overset{(31)}{\leq} \left(1 + \frac{\alpha - 1}{t} \right) \|g_t^{[u,k]}\|.$$

By Cauchy-Schwarz inequality and the fact that $\|u_t^{[u,k]}\| = 1$, $\|g_{t+1}^{[u,k]}\| \geq g_{t+1}^{[u,k]} \cdot u_t^{[u,k]}$, so it is enough to satisfy $\rho_{t+1}^{[u,k]} \geq \alpha$ that

$$g_{t+1}^{[u,k]} \cdot u_t^{[u,k]} \;\geq\; \left(1 + \frac{\alpha - 1}{t} \right) \|g_t^{[u,k]}\|. \tag{32}$$

Eq. (32) also implies that:

$$\left\{ \rho_{t+1}^{[u,k]} \geq \alpha \right\} \;\supseteq\; \left\{ g_{t+1}^{[u,k]} \cdot u_t^{[u,k]} \geq (1 + \tfrac{\alpha-1}{t}) \|g_t^{[u,k]}\| \right\}.$$

Moreover, from (25), (32) can be implied by

$$\|g_t^{[u,k]}\| - L_\nabla \eta \|\widehat{g}_t\| + Z_t^{[u,k]} \;\geq\; \left(1 + \tfrac{\alpha-1}{t}\right)\|g_t^{[u,k]}\|.$$

Then it holds that

$$Z_t^{[u,k]} \;\geq\; \left(\tfrac{\alpha-1}{t}\right)\|g_t^{[u,k]}\| + L_\nabla\, \eta\, \|\widehat{g}_t\|. \tag{33}$$

On the event $\mathcal{E}_q = \{\|\varepsilon_t\| \leq \iota_q\}$, the triangle inequality gives

$$\|\widehat{g}_t\| = \|g_t + \varepsilon_t\| \;\leq\; \|g_t\| + \|\varepsilon_t\| \;\leq\; \|g_t\| + \iota_q. \tag{34}$$

On the event $\mathcal{E}_s = \{\|\xi_t\| \leq \iota_s\}$ and for the vector $u_t^{[u,k]}$,

$$|\xi_t^{[u,k]} \cdot u_t^{[u,k]}| \;\leq\; \|\xi_t^{[u,k]}\|\, \|u_t^{[u,k]}\| \;\leq\; \|\xi_t\| \;\leq\; \iota_s. \tag{35}$$

Recall from Lemma C.7 that $Z_t^{[u,k]} = \xi_{t+1}^{[u,k]} \cdot u_t^{[u,k]} - \xi_t^{[u,k]} \cdot u_t^{[u,k]}$. Combining (33), (34), and (35), we obtain the *sufficient* condition for $\rho_{t+1}^{[u,k]} \geq \alpha$ that

$$\xi_{t+1}^{[u,k]} \cdot u_t^{[u,k]} \;\geq\; \left(\tfrac{\alpha-1}{t}\right)\|g_t^{[u,k]}\| + L_\nabla\, \eta\left(\|g_t\| + \iota_q\right) + \iota_s. \tag{36}$$

Furthermore, condition on $\mathcal{F}_t$ and $u_t^{[u,k]}$. With the conclusion in Lemma C.7 that $\xi_{t+1}^{[u,k]} \cdot u_t^{[u,k]} \sim \mathcal{N}(0, \sigma_{u,k}^2)$ and is independent of $(\xi_s)_{s\leq t}$ and ε_t, we can get

$$\mathbb{P}\left(\xi_{t+1}^{[u,k]} \cdot u_t^{[u,k]} \geq a \middle| \mathcal{F}_t, \mathcal{E}_q, \mathcal{E}_s\right) = 1 - \Phi\left(\frac{a}{\sigma_{u,k}}\right) = \Phi\left(-\frac{a}{\sigma_{u,k}}\right), \tag{37}$$

for any deterministic threshold a. Taking a to be the right-hand side of (36) yields exactly the conditional bound (29).

By construction, the mixed-precision error ε_t is independent of the SGD noise ξ_t and of $\mathcal{F}_t$ (Assumption C.1); hence $\mathcal{E}_q$ and $\mathcal{E}_s$ are independent events. Therefore

$$\mathbb{P}(\mathcal{E}_q \cap \mathcal{E}_s) \;=\; \mathbb{P}(\mathcal{E}_q)\,\mathbb{P}(\mathcal{E}_s) \;\geq\; (1-\delta_q)(1-\delta_s).$$

For any event $\mathcal{A}$, $\mathbb{P}(\mathcal{A}) \geq \mathbb{P}(\mathcal{A} \cap \mathcal{E}_q \cap \mathcal{E}_s)$, hence

$$\begin{aligned}
\mathbb{P}\left(\rho_{t+1}^{[u,k]} \geq \alpha \middle| \mathcal{F}_t\right) \geq& \mathbb{E}\left[\mathbf{1}_{\mathcal{E}_q \cap \mathcal{E}_s}\, \mathbb{P}\left(\rho_{t+1}^{[u,k]} \geq \alpha \middle| \mathcal{F}_t, \mathcal{E}_q, \mathcal{E}_s\right) \;\middle|\; \mathcal{F}_t\right] \\
\geq& (1-\delta_q)(1-\delta_s) \cdot \Phi\left(-\frac{\left(\tfrac{\alpha-1}{t}\right)\|g_t^{[u,k]}\| \;+\; L_\nabla\, \eta\,(\|g_t\| + \iota_q) \;+\; \iota_s}{\sigma_{u,k}}\right),
\end{aligned}$$

which is (30). □

C.3 Proof for Theorem 4.1

Then we consider the convergence rate with the proposed dynamic and the error recursion for the mixed-precision error, we present the following assumption to characterize the mixed-precision error. Here, we denote $\mathfrak{B} := \{[u,k] | u = 1, \cdots, U;\, k = 0, 1, \cdots, K_u\}$ as the set of operators. Here we assumption that $T_{\text{lock}} = 1$ for convince, which means that the operator that has switched to high precision will maintain the high-precision state for 1 step.

Assumption C.9. The mixed-precision error is mean-zero and conditionally independent of $\mathcal{F}_t$. It decomposes operator-wise: $\varepsilon_t = \left(\varepsilon_t^b\right)_{b\in\mathfrak{B}}$ with

$$\mathbb{E}\left[\|\varepsilon_t\|^2 \middle| \{\rho_t^b\}_{b\in\mathfrak{B}}\right] = \sum_{b\in\mathfrak{B}} v_{P_t^b}^b\left(\rho_t^b\right),$$

where $P_t^b \in \{\text{low}, \text{high}\}$ is the precision used at operator b and time t.

Moreover, for each operator $b \in \mathfrak{B}$, the low/high-precision error profiles $v_{\text{low}}^b(\rho), v_{\text{high}}^b(\rho) \geq 0$ define a nonnegative gap $\Delta v^b(\rho) := v_{\text{low}}^b(\rho) - v_{\text{high}}^b(\rho) \geq 0$ that is nondecreasing in ρ. And we also assume that v is monotonous with respect to ρ.

Lemma C.10. *Under Assumption C.2, for any policy of $\{P_t^b\}$ and $\eta \le 1/L$,*

$$\mathbb{E}\left[F(\theta_{t+1})|\mathcal{F}_t\right] \le F(\theta_t) - \eta\Big(1 - \tfrac{L\eta}{2}\Big)\|\nabla F(\theta_t)\|^2 + \tfrac{L\eta^2}{2}\left(\sigma_{\mathrm{sgd}}^2 + \sum_{b\in\mathfrak{B}} \mathbb{E}\big[\, v^b_{p_t^b}(\rho_t^b) \mid \mathcal{F}_t\big]\right). \tag{38}$$

Proof. By L_∇-smoothness of F as Assumption C.2, for any θ_t and $y = \theta_{t+1} = \theta_t - \eta(\nabla F(\theta_t) + \xi_t + \varepsilon_t)$:

$$\begin{aligned} F(\theta_{t+1}) \le& F(\theta_t) - \eta\big\langle \nabla F(\theta_t), \nabla F(\theta_t) + \xi_t + \varepsilon_t\big\rangle + \tfrac{L\eta^2}{2}\big\|\nabla F(\theta_t) + \xi_t + \varepsilon_t\big\|^2 \\ =& F(\theta_t) - \eta\|\nabla F(\theta_t)\|^2 - \eta\,\langle \nabla F(\theta_t), \xi_t\rangle - \eta\,\langle \nabla F(\theta_t), \varepsilon_t\rangle \\ &+ \tfrac{L\eta^2}{2}\Big(\|\nabla F(\theta_t)\|^2 + 2\langle \nabla F(\theta_t), \xi_t + \varepsilon_t\rangle + \|\xi_t + \varepsilon_t\|^2\Big). \end{aligned} \tag{39}$$

Take conditional expectation with respect to $\mathcal{F}_t$. It holds that:

$$\begin{aligned} \mathbb{E}\left[F(\theta_{t+1})|\mathcal{F}_t\right] \le& F(\theta_t) - \eta\|\nabla F(\theta_t)\|^2 + \tfrac{L\eta^2}{2}\|\nabla F(\theta_t)\|^2 + \tfrac{L\eta^2}{2}\,\mathbb{E}\|\xi_t\|^2 + \tfrac{L\eta^2}{2}\,\mathbb{E}\|\varepsilon_t\|^2 \\ =& F(\theta_t) - \eta\Big(1 - \tfrac{L\eta}{2}\Big)\|\nabla F(\theta_t)\|^2 + \tfrac{L\eta^2}{2}\mathbb{E}\|\xi_t\|^2 + \tfrac{L\eta^2}{2}\mathbb{E}\|\varepsilon_t\|^2. \end{aligned} \tag{40}$$

Finally, invoke $\mathbb{E}[\|\xi_t\|^2 \mid \mathcal{F}_t] \le \sigma_{\mathrm{sgd}}^2$ and

$$\mathbb{E}\Big[\|\varepsilon_t\|^2 \mid \mathcal{F}_t\Big] = \sum_{b\in\mathfrak{B}} \mathbb{E}\Big[v^b_{p_t^b}(\rho_t^b) \mid \mathcal{F}_t\Big],$$

which gives the stated inequality. □

The following proposition characterize the reduction of mixed-precision error with dynamic precision compared to low-precision training.

Proposition C.11. *Suppose Assumption C.9 holds, then the mixed-precision error with dynamic precision strategy can be given by*

$$\mathbb{E}\Big[\|\varepsilon_{t+1}\|^2\Big]_{\mathrm{Dynamic}} = \sum_{b\in\mathfrak{B}} \mathbb{E}\Big[v^b_{\mathrm{high}}(\rho_{t+1}^b)\,\mathbf{1}_{H_t(\alpha)} + v^b_{\mathrm{low}}(\rho_{t+1}^b)\,\mathbf{1}_{\neg H_t(\alpha)}\Big]. \tag{41}$$

Therefore

$$\mathbb{E}\Big[\|\varepsilon_{t+1}\|^2\Big]_{\mathrm{Low}} - \mathbb{E}\Big[\|\varepsilon_{t+1}\|^2\Big]_{\mathrm{Dynamic}} = \mathbb{E}\Big[\Delta v^b(\rho_{t+1}^b)\,\mathbf{1}_{H_t(\alpha)}\Big]. \tag{42}$$

Proof. Fix t and condition on $\mathcal{F}_t$. Under the GNMR-based Dynamic policy, it holds for any $[u,k] \in \mathfrak{B}$ that:

$$p_{t+1}^{[u,k]} = \begin{cases} \text{high}, & \text{if } (t,u,k) \in H_t(\alpha), \\ \text{low}, & \text{otherwise.} \end{cases}$$

Thus by Assumption C.9,

$$\begin{aligned} \mathbb{E}\left[\|\varepsilon_{t+1}\|^2\middle|\mathcal{F}_t\right]_{\mathrm{Dynamic}} &= \sum_{b\in\mathfrak{B}} \mathbb{E}\Big[\, v^b_{p^b_{t+1}}(\rho_{t+1}^b)\Big|\mathcal{F}_t\Big] \\ &= \sum_{b\in\mathfrak{B}} \mathbb{E}\Big[\, v^b_{\mathrm{high}}(\rho_{t+1}^b)\mathbf{1}_{H_t(\alpha)} + v^b_{\mathrm{low}}(\rho_{t+1}^b)\mathbf{1}_{\neg H_t(\alpha)} \;\Big|\; \mathcal{F}_t\Big]. \end{aligned} \tag{43}$$

Under the low-precision, $p_{t+1}^b \equiv$ low hence

$$\mathbb{E}\Big[\|\varepsilon_{t+1}\|^2 \mid \mathcal{F}_t\Big]_{\mathrm{Low}} = \sum_{b\in\mathfrak{B}} \mathbb{E}\Big[\, v^b_{\mathrm{low}}(\rho_{t+1}^b) \;\Big|\; \mathcal{F}_t\Big].$$

Subtract the two conditional expressions, use linearity of expectation, and then uncondition:

$$\begin{aligned} \mathbb{E}\Big[\|\varepsilon_{t+1}\|^2\Big]_{\mathrm{Low}} - \mathbb{E}\Big[\|\varepsilon_{t+1}\|^2\Big]_{\mathrm{Dynamic}} &= \sum_{b\in\mathfrak{B}} \mathbb{E}\Big[\,\Big(v^b_{\mathrm{low}}(\rho_{t+1}^b) - v^b_{\mathrm{high}}(\rho_{t+1}^b)\Big)\mathbf{1}_{H_t(\alpha)}\Big] \\ &= \sum_{b\in\mathfrak{B}} \mathbb{E}\Big[\Delta v^b(\rho_{t+1}^b)\,\mathbf{1}_{H_t(\alpha)}\Big], \end{aligned} \tag{44}$$

where $\Delta v^b(\rho) := v^b_{\mathrm{low}}(\rho) - v^b_{\mathrm{high}}(\rho) \ge 0$. No independence between ρ_{t+1}^b and $H_t(\alpha)$ is required, since the indicator remains inside the expectation. □

Then the following theorem illustrates the improvement of the dynamic precision strategy in the objective function, which shows the benefit of the proposed strategy in convergence and training performance.

Theorem C.12. *Under Assumptions C.1, C.2, C.4, and C.9 holds. If $\eta \le 1/L$, the expected decrease from θ_{t+1} to θ_{t+2} satisfies*

$$\Delta_{\text{desc}}^{(\text{Low-Dynamic})} := \mathbb{E}[F(\theta_{t+2})]_{\text{Low}} - \mathbb{E}[F(\theta_{t+2})]_{\text{Dynamic}} \ge \frac{L\eta^2}{2} \sum_{b\in\mathfrak{B}} \mathbb{E}\big[\Delta v^b(\rho_{t+1}^b)\,\mathbf{1}_{H_t(\alpha)}\big]. \tag{45}$$

In addition, it holds that

$$\Delta_{\text{desc}}^{(\text{Low-Dynamic})} \;\ge\; \frac{L\eta^2}{2} \sum_{b\in\mathfrak{B}} \mathbb{P}(H_t(\alpha))\;\pi^b\;\Delta v^b(\tau^b). \tag{46}$$

Proof. For policy "Pol" $\in \{\text{Low}, \text{Dynamic}\}$, it holds from Lemma C.10 that:

$$\mathbb{E}[F(\theta_{t+2})]_{\text{Pol}} \le \mathbb{E}[F(\theta_{t+1})]_{\text{Pol}} - \eta\Big(1 - \tfrac{L\eta}{2}\Big)\mathbb{E}\|\nabla F(\theta_{t+1})\|^2 + \tfrac{L\eta^2}{2}\left(\sigma_{\text{sgd}}^2 + \mathbb{E}\sum_b v^b_{p^b_{t+1}(\text{Pol})}(\rho^b_{t+1})\right). \tag{47}$$

Note that the $t+2$-th step uses low precision or high precision, the function value $\mathbb{E}[F(\theta_{t+1})]$ is the same for both cases. Then we can obtain that:

$$\Delta_{\text{desc}}^{(\text{Low-Dynamic})} \ge \tfrac{L\eta^2}{2}\mathbb{E}\sum_b v^b_{\text{low}}(\rho^b_{t+1}) - \tfrac{L\eta^2}{2}\mathbb{E}\sum_b \big[v^b_{\text{high}}(\rho^b_{t+1})\,\mathbf{1}_{H_t(\alpha)} + v^b_{\text{low}}(\rho^b_{t+1})\,\mathbf{1}_{\neg H_t(\alpha)}\big]. \tag{48}$$

Hence

$$\Delta_{\text{desc}}^{(\text{Low-Dynamic})} \;\ge\; \frac{L\eta^2}{2}\sum_b \mathbb{E}\big[\Delta v^b(\rho^b_{t+1})\,\mathbf{1}_{H_t(\alpha)}\big], \tag{49}$$

which proves the first claim.

For the second claim, since Δv^b satisfies that:

$$\Delta v^b(\rho^b_{t+1}) \;\ge\; \Delta v^b(\tau^b)\,\mathbf{1}_{\{\rho^b_{t+1}\ge\tau^b\}}.$$

Therefore, with Assumption C.4, it holds that

$$\begin{aligned}
&\mathbb{E}\big[\Delta v^b(\rho^b_{t+1})\,\mathbf{1}_{H_t(\alpha)}\big] \ge \Delta v^b(\tau^b)\,\mathbb{E}\big[\mathbf{1}_{\{\rho^b_{t+1}\ge\tau^b\}}\mathbf{1}_{H_t(\alpha)}\big]\\
=&\Delta v^b(\tau^b)\,\mathbb{P}\big(\rho^b_{t+1}\ge\tau^b,\,H_t(\alpha)\big) = \Delta v^b(\tau^b)\,\mathbb{P}(H_t(\alpha))\,\mathbb{P}\big(\rho^b_{t+1}\ge\tau^b \mid H_t(\alpha)\big)\\
\ge&\Delta v^b(\tau^b)\,\mathbb{P}(H_t(\alpha))\,\pi^b.
\end{aligned} \tag{50}$$

Summing over b and multiplying by $L\eta^2/2$ yields the stated lower bound. □

C.4 Proof for Theorem 4.2

Here we present the estimation for the probability that the GNMR value reach the threshold. To begin with, we need to present some essential notations.

Notations. For a fixed operator $[u,k] \in \mathfrak{B}$, we regard the evaluated gradient norm at step s as a stochastic variable that can be denoted as $X_t^{[u,k]} := \big\|g_t^{[u,k]}\big\|$. We also write $\mu^{[u,k]} := \mathbb{E}[X_s^{[u,k]}]$ for its expectation. Define the normalized (dimensionless) sequence

$$Y_t^{[u,k]} := \frac{X_t^{[u,k]}}{\mu^{[u,k]}}, \quad \mathbb{E}[Y_t^{[u,k]}] = 1,$$

and the arithmetic mean of past normalized magnitudes

$$S_{t-1}^{[u,k]} := \frac{1}{t-1}\sum_{\tau=1}^{t-1} Y_\tau^{[u,k]} = \frac{1}{\mu^{[u,k]}}\Big(\frac{1}{t-1}\sum_{\tau=1}^{t-1} X_\tau^{[u,k]}\Big) = \frac{m_t^{[u,k]}}{\mu^{[u,k]}}. \tag{51}$$

Thus, the operator-wise GNMR can then be written equivalently as

$$R_t^{[u,k]} := \frac{X_t^{[u,k]}}{m_t^{[u,k]}} = \frac{Y_t^{[u,k]}}{S_{t-1}^{[u,k]}}.$$

The following assumptions will be used in the subsequent proofs.

Assumption C.13. For a fixed operator $b \in \mathfrak{B}$, the normalized sequence $\{Y_t^b\}_{t\geq 1}$ satisfies a sub-exponential (Orlicz ψ_1) tail: $\|Y_s^b - 1\|_{\psi_1} \leq \kappa_b$ for some $\kappa_b > 0$ that is independent to s, where the ψ_1-Orlicz norm can be defined by

$$\|Z\|_{\psi_1} := \inf\{c > 0 : \mathbb{E}\exp(|Z|/c) \leq 2\}.$$

Remark C.14. With Assumption C.13, the stochastic variable Y_t^b for all $b \in \mathfrak{B}$ satisfy that

$$\mathbb{E}e^{\lambda Y_t^b} \leq \exp\big(C_2\lambda^2\kappa_b^2\big) \quad \text{for all } |\lambda| \leq C_1/Y_t^b, \tag{52}$$

where $C_1, C_2 > 0$ are constants.

Moreover, Y_t^b obeys the standard tail:

$$\mathbb{P}(Y_t^b \geq x) \leq \exp\Big(-c\min\big\{x^2/\kappa_b^2,\ x/\kappa_b\big\}\Big), \quad x \geq 0, \tag{53}$$

for a constant $c > 0$. Inequality (53) is obtained from (52) by Chernoff's method (Chernoff, 1952) with the choice $\lambda = \min\{x/(2C_2\kappa^2),\ C_1/(2\kappa)\}$.

Lemma C.15. *Under Assumption C.13, for any $\nu \in (0,1)$ and $t \geq 2$,*

$$\mathbb{P}\left(\left|S_{t-1}^b - 1\right| \geq \nu\right) \leq 2\exp\left(-c(t-1)\min\left\{\frac{\nu^2}{\kappa_b^2}, \frac{\nu}{\kappa_b}\right\}\right), \tag{54}$$

where $c > 0$ is an absolute constant.

Proof. For $\tau = 1, \ldots, t-1$ and $b \in \mathfrak{B}$, denote $Z_\tau := Y_s^b - 1$ Let $Z_\tau := Y_s^b - 1$ for $s = 1, \ldots, t-1$. By Assumption C.13 and the definition, $\mathbb{E}[Z_\tau] = 0$, $\|Z_s\|_{\psi_1} \leq \kappa_b$, and Z_τ are independent.

We firstly bound the upper tail $\mathbb{P}\big(S_{t-1}^b - 1 \geq \nu\big)$. Observe that

$$S_{t-1}^b - 1 = \frac{1}{t-1}\sum_{\tau=1}^{t-1} Z_\tau.$$

For any $\lambda > 0$, by Chernoff's method it holds for $0 < \lambda \leq C_1/\kappa_b$ that,

$$\begin{aligned}\mathbb{P}\left(S_{t-1}^b - 1 \geq \nu\right) &= \mathbb{P}\left(\sum_{s=1}^{t-1} Z_s \geq (t-1)\nu\right) \leq \exp\left(-\lambda(t-1)\nu\right)\mathbb{E}\left[\exp\left(\lambda\sum_{s=1}^{t-1} Z_s\right)\right] \\ &= \exp\big(-\lambda(t-1)\nu\big)\prod_{s=1}^{t-1}\mathbb{E}\left[\exp(\lambda Z_s)\right] \leq \exp\Big(-\lambda(t-1)\nu + (t-1)C_2\lambda^2\kappa_b^2\Big).\end{aligned} \tag{55}$$

Optimize the quadratic upper bound over λ under this constraint. The unconstrained minimizer is $\lambda^\star = \nu/(2C_2\kappa_b^2)$. Then we consider the probability obtained in Eq. (55) in the following two cases:

(i): if $\lambda^\star \leq C_1/\kappa_b$ (i.e., $\nu \leq 2C_1C_2\,\kappa_b$), then

$$\mathbb{P}\big(S_{t-1}^b - 1 \geq \nu\big) \leq \exp\left(-\tfrac{(t-1)\nu^2}{4C_2\kappa_b^2}\right). \tag{56}$$

(ii): if $\lambda^\star > C_1/\kappa_b$ (i.e., $\nu > 2C_1C_2\,\kappa_b$), plugging $\lambda = C_1/(2\kappa_b)$ into (55) and we can get:

$$\mathbb{P}\big(S_{t-1}^b - 1 \geq \nu\big) \leq \exp\Big(-\tfrac{C_1}{2}(t-1)\tfrac{\nu}{\kappa_b} + (t-1)\tfrac{C_1^2C_2}{4}\Big) \leq \exp\Big(-c'(t-1)\tfrac{\nu}{\kappa_b}\Big), \tag{57}$$

where $c' > 0$ is an constant.

Combining the two cases with a single expression yields

$$\mathbb{P}\big(S_{t-1}^b - 1 \geq \nu\big) \leq \exp\left(-c(t-1)\min\left\{\tfrac{\nu^2}{\kappa_b^2}, \tfrac{\nu}{\kappa_b}\right\}\right).$$

Applying the same bound to $-(S_{t-1}^b - 1)$ and then we completes the proof. □

Lemma C.16. *For any $\varepsilon, \nu \in (0,1)$, it holds for all $b \in \mathfrak{B}$ that*

$$\left\{R_t^b \geq 1+\varepsilon\right\} \subseteq \left\{S_{t-1}^b \leq 1-\nu\right\} \cup \left\{Y_t^b \geq (1+\varepsilon)(1-\nu)\right\}.$$

Proof. Recall $R_t^b = Y_t^b / S_{t-1}^b$. Consider the complement of the union on the right:

$$\{S_{t-1}^b > 1-\nu\} \cap \{Y_t^b < (1+\varepsilon)(1-\nu)\}.$$

On this event we have

$$R_t^b = \frac{Y_t^b}{S_{t-1}^b} < \frac{(1+\varepsilon)(1-\nu)}{1-\nu} = 1+\varepsilon,$$

hence $\{R_t^b \geq 1+\varepsilon\}$ cannot occur. Taking complements proves the inclusion. □

Theorem C.17. *Under Assumption C.13, for any $\varepsilon \in (0,1)$ and any $\nu \in (0,1)$,*

$$\mathbb{P}\left(R_t^b \geq 1+\varepsilon\right) \leq 2\exp\left(-c\,(t-1)\min\left\{\frac{\nu^2}{\kappa_b^2}, \frac{\nu}{\kappa_b}\right\}\right) + \exp\left(-c\,\min\left\{\frac{\zeta^2}{\kappa_b^2}, \frac{\zeta}{\kappa_b}\right\}\right), \tag{58}$$

where $\zeta := (1+\varepsilon)(1-\nu) - 1 = \varepsilon - \nu - \varepsilon\nu$ and $c > 0$ is a constant. In particular, choosing $\nu = \varepsilon/2$ yields

$$\mathbb{P}\left(R_t^b \geq 1+\varepsilon\right) \leq 2\exp\left(-c\,(t-1)\min\left\{\frac{\varepsilon^2}{\kappa_b^2}, \frac{\varepsilon}{\kappa_b}\right\}\right) + \exp\left(-c\min\left\{\frac{\varepsilon^2}{\kappa_b^2}, \frac{\varepsilon}{\kappa_b}\right\}\right). \tag{59}$$

Proof. By Lemma C.16 it can be obtained that,

$$\mathbb{P}\left(R_t^b \geq 1+\varepsilon\right) \leq \mathbb{P}\left(S_{t-1}^b \leq 1-\nu\right) + \mathbb{P}\left(Y_t^b \geq (1+\varepsilon)(1-\nu)\right). \tag{60}$$

We firstly consider the first term on the right hand side of (60). Applying Lemma C.15 to the deviation $S_{t-1}^b - 1 \leq -\nu$ and using the two-sided bound yields

$$\mathbb{P}\big(S_{t-1}^b \leq 1-\nu\big) \leq 2\exp\Big(-c\,(t-1)\,\min\left\{\tfrac{\nu^2}{\kappa_b^2}, \tfrac{\nu}{\kappa_b}\right\}\Big).$$

Then we consider the second term. Let $\zeta := (1+\varepsilon)(1-\nu) - 1$. Then

$$\mathbb{P}\big(Y_t^b \geq (1+\varepsilon)(1-\nu)\big) = \mathbb{P}\big(Y_t^b - 1 \geq \zeta\big).$$

By Assumption C.13, $Y_t^b - 1$ is sub-exponential with ψ_1 norm less than κ_b, so the tail bound (53) gives

$$\mathbb{P}\big(Y_t^b - 1 \geq \zeta\big) \leq \exp\Big(-c\,\min\left\{\tfrac{\zeta^2}{\kappa_b^2}, \tfrac{\zeta}{\kappa_b}\right\}\Big),$$

with the understanding that if $\zeta \leq 0$ this upper bound is ≥ 1, in which case the inequality remains valid. Combining the two estimates with (60) proves the stated bound. Finally, take $\nu = \varepsilon/2$; then $\zeta = (1+\varepsilon)(1-\varepsilon/2) - 1 = \frac{\varepsilon}{2}(1-\varepsilon) \in (0, \varepsilon/2]$, and both exponents share the same functional form up to constants, giving the simplified display. □

D PEAK-MEMORY MODELING UNDER THE $maxO$ BUDGET

This section provides a simple upper-bound model for the additional peak *activation* memory introduced when the controller promotes a subset of operators to higher precision under a hard $maxO$ budget.

Setup. Let $\mathcal{B}$ be the set of eligible operators. For each operator $b \in \mathcal{B}$, let A_b denote the number of activation elements that must be saved for backward under a fixed batch and sequence configuration. At step t, let $\mathcal{S}_t \subseteq \mathcal{B}$ be the set of operators executed in high precision. By design, the budget enforces $|\mathcal{S}_t| \leq maxO$ for all t.

From bitwidth to bytes. Let p_{low} and p_{high} denote the storage precisions in bits per element. Memory is typically measured in bytes, and one byte equals eight bits. Therefore, the bytes-per-element conversion is

$$\text{bytes per element} = \frac{\text{bits per element}}{8}.$$

Table 3: Model configurations for different LLaMA scales used in GNMR-guided activation quantization.

Params	Hidden	Intermediate	Heads	Layers	Training Tokens	Learning Rate
60M	512	1376	8	8	1.3B	2.5E-3
130M	768	2048	12	12	2.6B	2.5E-3
350M	1024	2736	16	24	7.8B	1E-3
1B	2048	5461	24	32	13.1B	6E-4

Table 4: Proportion of linear blocks exceeding GNMR and Δ-GNMR thresholds with different model size and thresholds.

Model size	α_t	β_t	GNMR	Δ-GNMR
60M	2.0	0.5	0.070%	0.081%
	3.0	1.0	0.019%	0.033%
350M	2.0	0.5	0.141%	0.067%
	3.0	1.0	0.019%	0.016%

Additional peak activation memory. Relative to a baseline that runs all eligible operators in low precision, the additional activation memory at step t is

$$\Delta M_t = \frac{p_{\text{high}} - p_{\text{low}}}{8} \sum_{b \in \mathcal{S}_t} A_b, \tag{61}$$

measured in bytes. Using $|\mathcal{S}_t| \le maxO$, we obtain the upper bound

$$\Delta M_t \le \frac{p_{\text{high}} - p_{\text{low}}}{8} \sum_{b \in \text{TopK}(\mathcal{B}, maxO; A_b)} A_b \le \frac{p_{\text{high}} - p_{\text{low}}}{8} \cdot maxO \cdot A_{\max}, \tag{62}$$

where $A_{\max} := \max_{b \in \mathcal{B}} A_b$.

Interpretation. Eq. (62) shows that the additional peak activation memory grows linearly with the budget $maxO$ and with the activation footprint of the promoted operators. This explains why a hard $maxO$ constraint is an effective and model-agnostic knob for controlling peak memory when dynamic precision promotion is triggered by rare but bursty instability events.

E Experimental setup and Additional Results

E.1 Pre-training LLaMA-2 with activation quantization

During pre-training across all LLaMA model scales, we implement the standardized configuration framework from (Zhao et al., 2024), with key technical specifications comprising a 256-token maximum sequence length and a global batch size of 512 samples, translating to 13.1K tokens per batch. The learning rate scheduling integrates two-phase optimization: initial linear warm-up during the first 10% of training iterations, succeeded by cosine decay gradually reducing the learning rate to 10% of its initial magnitude. Complete architectural configurations and training protocol details are systematically documented in Table 3.

To understand this effectiveness, Table 4 provides the percentage of linear blocks where the GNMR and Δ-GNMR value surpasses the respective thresholds. The fact that fewer than 0.2% of blocks exceed these thresholds is pivotal. It demonstrates that the performance improvement is achieved by precisely identifying and applying higher precision to only a minimal yet critical set of unstable blocks, thereby stabilizing the training process without incurring substantial overhead.

Table 5: Model configurations for different LLaMA scales used in GNMR-guided DeepSeek-style training.

Params	Hidden	Intermediate	Heads	Layers	Training Tokens	Learning Rate
60M	512	1376	8	8	1.3B	2E-3
130M	768	2048	12	12	2.6B	1E-3
350M	1024	2736	16	24	7.8B	5E-4
1B	2048	5472	24	32	13.1B	5E-4

Table 6: The per-step time (↓) and the throughput (↑) of the LLaMA-2 1.3B training task with different precision strategies on the same device.

precision	per-step time (s)	throughput (token/s)
8-bit	1.266	103532
8-bit/16-bit	**1.264**	**103696**
16-bit	1.294	101292

E.2 Pre-training LLaMA-2 with DeepSeek-style Mixed-Precision Strategy

E.2.1 Experimental setup

In this experiment, we maintain the same baseline configuration as the activation quantization study, including sequence length, batch size, and learning rate scheduler. The detailed model architecture specifications are provided in Table 5. For the training task, we use the two-stage GNMR threshold that we use a high threshold in the first 2.5% steps and use a low threshold in the other steps.

We investigate two complementary mixed-precision configurations, each establishing a distinct precision hierarchy:

- BF16 for high-precision operators with Transformer Engine's hybrid FP8 for low-precision operators;
- Hybrid FP8 for high-precision operators with quantization-simulated 4-bit (E2M1) and 6-bit (E3M2) linear operators as low-precision alternatives.

This experimental design enables systematic analysis of GNMR-driven dynamic precision allocation across layers, while examining how different precision hierarchies influence convergence stability and training efficiency.

BF16 High-Precision with Hybrid FP8 Low-Precision Operators. In the first configuration, bfloat16 (BF16) serves as the high-precision format, establishing the baseline for parameter storage and computationally intensive operations. For low-precision computations, we implement a hybrid FP8 strategy to balance numerical stability with computational efficiency.

Specifically, the FP8-E4M3 format (4 exponent bits, 3 mantissa bits) is employed during forward propagation, while FP8-E5M2 (5 exponent bits, 2 mantissa bits) is utilized for backward propagation. This mixed-precision framework is implemented through Transformer Engine, which manages both GEMM operations and activation tensor storage in FP8 formats.

Within the FP8 autocast context, BF16 weights and inputs undergo dynamic scaling and conversion to FP8-E4M3 for forward computations. During backward propagation, incoming gradients are cast to FP8-E5M2 to prevent numerical underflow or overflow. All gradients resulting from FP8 GEMM operations are subsequently dequantized back to BF16 for parameter updates.

Quantization-Simulated Low-Precision Operators. In the second configuration, hybrid FP8 operators constitute the high-precision baseline, while low-precision behavior is emulated through quantization using E2M1 (4-bit) and E3M2 (6-bit) formats. For each token, activation values are scaled into the representable range of the target precision, rounded to the nearest representable value, and rescaled to their original dynamic range.

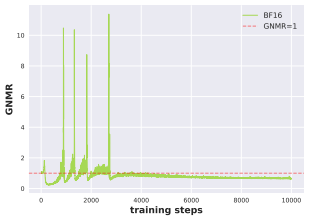

Figure 2: The GNMR curve during the pre-training task of LLaMA-2 model with BF16 precision.

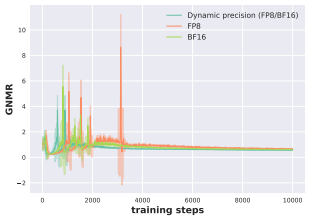

Figure 3: Evolution of average GNMR values across operators under different precision strategies.

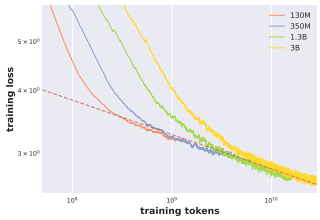

Figure 4: Scaling performance for LLaMA-2 pre-training with our dynamic-precision strategy.

Table 7: The validation perplexity (↓) for the pre-training LLaMA-2 130M model with different two-stage GNMR threshold, α_t^{init} represents the threshold in the initial 2.5% steps and α_t^{main} represents the threshold in the other steps.

α_t^{init}	α_t^{main}	Training tokens (B)		
		1.3	2.1	2.6
1.1	1.1	26.53	23.95	23.57
1.5	1.1	**26.49**	**23.92**	**23.54**
1.5	1.5	26.56	23.98	23.60

During forward propagation, both weights (w) and activations (x) undergo quantization prior to computation to simulate precision constraints, with the resulting activations (y) being quantized again. Backward propagation follows a similar quantization procedure: incoming gradients ($\partial L/\partial y$) are quantized, and the computed weight and input gradients ($\partial L/\partial w$ and $\partial L/\partial x$) are quantized before further propagation.

Dynamic Assignment of High-Precision Operators. Under GNMR guidance, the assignment of high- and low-precision operators is determined dynamically rather than through manual specification. GNMR adaptively selects layers for high-precision operation to maintain convergence stability, while allocating remaining layers to low-precision formats to enhance computational efficiency.

However, given the severely constrained representational range of 4-bit quantization, pretraining may still diverge despite GNMR regulation. To ensure stable and equitable comparisons, we impose additional manual constraints on precision assignment. Specifically, in selected experiments, we restrict low-precision operators to MLP submodules while maintaining attention submodules in high precision. Furthermore, the initial transformer layers are consistently preserved in high precision, independent of GNMR's dynamic allocation. These statically assigned high-precision layers are included in the total high-precision budget to maintain consistency across experimental configurations.

E.2.2 Wall-clock time and token throughput

Table 6 reports the wall-clock per-step time and token throughput under the same hardware/software stack. The results show that our GNMR-driven dynamic policy preserves essentially the same training wall-clock cost as fixed low-precision training (per-step time / throughput), while achieving the stability and quality gains reported in Table 2.

E.2.3 Average GNMR during the pre-training period

Figure 3 illustrates the evolution of average GNMR values across operators under different precision strategies. The results reveal that the dynamic precision strategy effectively reduces operator-level GNMR values, thereby enhancing training stability throughout the pre-training process.

E.2.4 Scaling performance

We also train LLaMA-2 models of various sizes—including 130M, 350M, 1.3B, and 3B parameters—under the proposed dynamic-precision strategy and record the corresponding training loss.

Table 8: The validation perplexity (↓) for the pre-training LLaMA-2 130M model with different $maxO$.

$maxO$	Training tokens (B)		
	1.3	2.1	2.6
2	26.56	23.97	23.59
4	**26.49**	**23.92**	**23.54**
6	26.50	23.94	23.56
8	26.56	24.01	23.63

Table 9: The validation perplexity (↓) for the pre-training LLaMA-2 models for variance model size with activation quantization under different precision strategy. α_t and β_t represent the threshold of GNMR and Δ-GNMR, respectively.

method	α_t	β_t	130M	350M
			2.2B tokens	6.4B tokens
ADAPT	N.A.	N.A.	215.10	88.97
GNMR + Δ-GNMR	1.5	0.3	**24.66**	18.84
	2.0	0.5	24.86	19.07
	3.0	1.0	25.33	18.77

As shown in Figure 4, the results demonstrate consistent scaling behavior across model sizes, with larger models achieving lower training loss under the same computational budget. Notably, the observed scaling trends align well with power-law predictions commonly reported in the literature, suggesting that our dynamic-precision approach preserves model quality while reducing bit-width. This consistent behavior across scales highlights the robustness of the proposed strategy and its potential for efficient large-scale model training.

E.2.5 Impact of threshold and maxO

To validate the benefit of the proposed two-stage decaying GNMR threshold, we pre-trained a LLaMA-2 130M model under different threshold configurations. With the maximum number of high-precision operators set to $maxO = 4$, we report validation perplexity in Table 7. We observe that the two-stage schedule outperforms using a consistently high threshold throughout training. Moreover, using a consistently low threshold also underperforms the two-stage schedule, supporting the benefit of adapting the threshold over the course of training.

We further trained the model using the two-stage threshold $(\alpha_t^{\text{init}}, \alpha_t^{\text{main}}) = (1.5, 1.1)$ with varying $maxO$ values. As summarized in Table 8, the model does not achieve optimal performance when $maxO$ is set too large or too small, providing clear guidance for hyperparameter selection.

E.3 Comparison of Static Baselines

To further analyze the effect of GNMR-based dynamic precision, we run additional experiments on LLaMA-2 130M and 350M with simulated low-bit activation quantization under a standard BF16 training stack. All runs use the same data, optimizer, learning-rate schedule, and token budget as in our main activation-quantization experiments presented in Appendix E.1; only the precision policy changes. Activations of all linear operators in Transformer blocks are quantized using a simulated 4/8-bit floating-point format on top of a BF16 implementation, while weights and optimizer states remain in BF16.

Static and stage-wise baselines. We implement a proxy ADAPT-style (Menon et al., 2018) baseline while sharing the same infrastructure as GNMR. Specifically, after a short warm-up phase (500 steps) in BF16, we enable a GNMR/GSS monitor that records per-block gradient-norm-to-history ratios. At the end of warm-up, we aggregate these statistics into a single scalar per block by averaging GNMR over the warm-up steps, sort blocks by this score, and select the top-25% "high-risk" blocks. For the remainder of training, these blocks are always executed with 8-bit activations, while all remaining blocks use 4-bit. The set of high-precision blocks is fixed and no further updates are made. This

Table 10: Model configurations for GPT-based MoE model used in GNMR-guided DeepSeek-style training.

Params	Attn Hidden	Per Expert Hidden	MoE Heads	Attn Heads	Topk	Layers	Learning Rate
3.7B	7168	2048	64	128	8	3	5E-5

Table 11: Training loss (↓) of the pre-training task for GPT models with MoE architecture with different precision strategies. ✗ means the model fails to converge.

Precision	Training steps (K)											
	1	2	3	4	6	8	10	12	14	17	20	29
8-bit	0.223	0.205	0.189	0.312	0.176	0.170	0.167	0.163	0.160	0.159	0.158	✗
8-bit/16-bit	0.224	0.201	0.189	0.183	0.173	0.170	0.168	0.164	0.162	0.159	0.159	0.154
BF16	0.226	0.205	0.187	0.181	0.172	0.171	0.167	0.163	0.159	0.159	0.156	0.152

Table 12: Model configurations for PanGu-1B used in GNMR-guided 4-bit training.

Params	Hidden	Intermediate	Heads	Layers	KV groups	Channels	Learning Rate
1.2B	1536	6144	12	26	6	128	2E-3

Table 13: Training loss (↓) of pretrained PanGu-1B models with BF16+4-bit mixed-precision strategy under different precision settings.

Precision	Training Tokens (B)										
	0	0.4	0.8	1.2	1.6	2.0	2.4	2.8	3.2	3.6	4.0
4-bit	12.261	4.656	4.177	3.670	3.527	3.348	3.314	3.238	3.170	3.156	3.124
4-bit+GNMR	12.261	4.650	3.929	3.575	3.564	3.309	3.290	3.215	3.155	3.128	3.076
BF16	12.261	4.608	3.878	3.526	3.472	3.210	3.160	3.077	3.004	2.978	2.913

mimics ADAPT-style approaches that first estimate layer sensitivity and then assign a static bit-width to each layer.

Experimental results. Table 9 presents the validation perplexity during training across different precision strategies. The results show that our GNMR-driven dynamic policy achieves better training performance than the ADAPT-style baseline.

E.4 Pre-training GPT-based MoE model with DeepSeek-style mixed-precision strategy

Experiment setup. We pre-trained a GPT-based model (Radford et al., 2019; Brown et al., 2020) with 3.7B parameters that incorporating a Mixture-of-Experts (MoE) architecture (Shazeer et al., 2017; Jacobs et al., 1991). In the MoE layers, we applied the proposed dynamic-precision strategy, which utilizes both 8-bit and 16-bit numerical representations, and compared its performance against training under fixed 8-bit and 16-bit precision settings, respectively. For the dynamic precision, we set the fixed threshold $\alpha = 0.9$. The learning rate was initialized as 5×10^{-5} and scheduled using a cosine decay strategy. Complete architectural configurations and training protocol details are systematically documented in Table 10.

Experiment results. Table 11 summarizes the training loss under different precision strategies. It can be observed that the fixed 8-bit precision resulted in a sharp increase in loss and failed to converge, indicating poor training stability. In contrast, the 8-bit/16-bit dynamic strategy based on GNMR

achieved performance comparable to that of 16-bit training. These results demonstrate the potential applicability of the proposed method to MoE models.

E.5 PRE-TRAINING PANGU MODEL WITH 4-BIT STRATEGY

Experimental Setup. We pre-trained a Pangu model with 1B parameters under different mixed-precision strategies, including 4-bit, 8-bit, and our proposed dynamic 4-bit/8-bit precision strategy based on GNMR. A fixed GNMR value of 0.4 was used, and the model was trained on 4B tokens. The model was trained with a global batch size of 256 and a sequence length of 4096. The learning rate was initialized to 2×10^{-5} and followed a cosine scheduler. All architectural configurations and training details are fully provided in Table 12.

Experimental Results. Table 13 presents the training loss under different precision strategies. It can be observed that the dynamic precision strategy effectively improves model convergence compared to fixed-precision approaches, demonstrating its adaptability to different model architectures.