HiFloat4 Format for End-To-End Reinforcement Learning Post-Training of Large Language Models

Hei Yi Mak **Shadan Golestan*** **Hoang Le*** **Mehran Taghian*** **Yunke Peng**

Yaoyuan Wang **Yao Wang** **Junsong Wang** **Tianchi Hu** **Fengchen He**

Guipeng Hu **Tanzila Rahman** **Anandharaju Durai Raju**

Huawei
{hei.yi.mak1, pengyunke}@huawei.com

Abstract

We present, to our knowledge, the first *end-to-end* FP4 RL post-training, in which both the rollout and training policies, including their forward and backward passes, operate at 4-bit precision. A systematic study reveals that the dominant source of degradation in FP4 RL is not training-side quantization error but rollout activation quantization: outliers stretch the dynamic range so far that a large number of activation values underflow to zero under FP4. Counterintuitively, restoring the training policy to higher precision while keeping the rollout in FP4 makes accuracy *worse* than full FP4 baseline, exposing rollout–training mismatch as the principal failure mode and ruling out standard pretraining-style fixes. We address this with **Rollout Res**idual **Q**uantization (**Rollout-ResQ**): a single residual correction term constrained to a hardware-friendly sparsity pattern, added only to the FP4 rollout matmul — a lightweight correction that recovers most of the precision lost to outlier-driven underflow without inflating the rollout's compute footprint. On Qwen2.5-3B, Rollout-ResQ paired with the HiFloat4 (HiF4) format — whose three-level hierarchical scaling preserves resolution under FP4's tight 4-bit budget — closes the accuracy gap to BF16 from 3.59% to 1.39%. bringing fully quantized FP4 RL within close reach of full precision. Applied to the open-standard MXFP4, the same recipe narrows the gap from 13.6% to 5.3%, revealing that FP4 format choice is a key factor that determines the ceiling on recoverable accuracy. The experimental results on Qwen2.5-Math-7B further confirm the recovery benefits of Rollout-ResQ with HiF4. Together, these results establish HiF4 as the enabling format for end-to-end FP4 RL post-training, and Rollout-ResQ as the activation-side mechanism that makes the gap to BF16 closable.

1 Introduction

Large Language Models (LLMs) have achieved remarkable performance across a wide range of applications [5, 43, 25]. By leveraging extensive Chain-of-Thought reasoning, LLMs have demonstrated strong capabilities in challenging domains [55, 67] such as mathematical problem solving [13], code generation [66], and agentic workflows [56, 4]. In doing so, these models produce long reasoning trajectories that interleave exploration, verification, and self-correction. Eliciting such behavior at scale has driven the widespread adoption of Reinforcement Learning (RL) for LLM

*indicates equal contribution. These authors are ordered alphabetically by last name.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

post-training [28, 64, 33, 59], most notably in the form of RL with Verifiable Rewards (RLVR) using policy-gradient algorithms such as Group Relative Policy Optimization (GRPO) [48] and its variants [63, 36, 32].

Despite its success, RL post-training incurs substantial computational, memory, and monetary costs [20, 28, 50]. Each iteration consists of two phases: a *rollout phase*, in which a *rollout policy* samples responses for each prompt, and a *training phase*, in which a *training policy* evaluates those responses and is updated. The rollout phase is typically the dominant bottleneck, since responses must be sampled autoregressively over long horizons [28, 57, 17]— a cost that is amplified in reasoning tasks, where longer trajectories provide stronger learning signals [57]. A natural remedy is low-precision computation, in particular quantization [9], which reduces memory usage and accelerates the matrix multiplications that dominate both generation and the forward/backward passes. Quantizing the rollout policy addresses this bottleneck, but quantizing the training policy is equally important for end-to-end efficiency [57]: beyond accelerating the forward and backward passes, it narrows the numerical mismatch between the two policies, yielding more stable post-training.

To improve the efficiency of RL post-training, prior work has explored low-precision computation with 8-bit formats such as FP8 or INT8, applied either to the rollout policy alone [27] or to both the rollout and training policies [57]. A central concern in these methods is the numerical mismatch between a quantized rollout policy and a higher-precision training policy, which they address through importance-sampling correction [27] or a unified low-precision pipeline [57]. More recently, QeRL [22] has pushed the rollout policy to 4-bit (NVFP4) while keeping gradients in higher precision through LoRA adapters, demonstrating that 4-bit rollouts are feasible without destabilizing training. Together, these results show that reduced precision can substantially accelerate training and lower memory overhead. In this work, we go a step further and study *end-to-end* FP4 RL post-training, in which both the rollout and training policies— including their forward and backward passes—operate in 4-bit. This regime is considerably more challenging than the settings above: unlike QeRL, we do not retain a higher-precision gradient path, and unlike 8-bit methods, we operate at a precision where the limited numerical range and resolution can destabilize optimization, yielding fragile and often non-convergent training dynamics.

We build on HiFloat4 (HiF4) [37], which uses a three-level hierarchical scaling scheme to expand dynamic range while preserving precision within FP4's tight 4-bit budget. We additionally evaluate against MXFP4 [46], the open-standard 4-bit format with a single shared scale per block, to test whether our findings depend on HiF4 specifically. Prior work [52] showed that HiF4 enables fully quantized FP4 pretraining with only minimal precision-recovery techniques on the training pass, matching full-precision baselines. The RL post-training setting, however, shifts where the difficulty lies, and whether the pretraining recipe transfers—and, if not, where the new difficulty resides— is the question we take up next.

We address this question through a systematic study of where precision actually matters in FP4 RL post-training. We localize the dominant source of degradation to rollout-side *activation* quantization, rather than to the training-pass quantization. Strikingly, restoring the training policy to higher precision while keeping the rollout in FP4 does not just fail to recover accuracy— it performs *worse* than fully FP4 training, suggesting that the rollout–training mismatch, not training-side noise, is the principal failure mode. To understand the rollout-side problem, we inspect the activation distributions and find that outliers stretch the dynamic range so far that the bulk of activation values underflow to zero under FP4. Preliminary experiments reveal that keeping the rollout activations in higher-precision representation recovers most of the accuracy lost relative to the baseline— but this fix forfeits the very benefit we set out to capture, since mixed-precision GEMMs (higher-precision activations $\times$ FP4 weights) give up the throughput of true FP4 matmul.

A natural next step is to draw on the post-training quantization (PTQ) literature, which has developed a rich toolkit for handling activation outliers: smoothing [58], scaling [30], clamping [54], learnable calibration transforms [47], and rotation- or permutation-based methods [3, 34, 29]. These methods, however, all assume a *static* inference setting: activation statistics are estimated once on a calibration set, quantization parameters are fit offline, and the resulting transforms are reused unchanged at deployment. RL post-training violates each of these assumptions. As the policy is updated, the rollout distribution— and with it the activation statistics—shifts continuously, so any calibration done at step t is partially stale by step $t + 1$, and recalibrating frequently to track this drift would erode the very efficiency gains motivating FP4 in the first place. We therefore take a different route, adopting

a per-tensor fallback mechanism inspired by [65], where outliers are separated from the activation tensor, resulting in two FP4 tensors to fully take advantage of low-precision GEMM. This recovers most of the gap to the BF16 baseline and, unlike the higher-precision activation alternatives, keeps the rollout in 4-bit end-to-end. The catch is that splitting the activation into two FP4 tensors doubles the number of FP4 GEMMs in the rollout, trading one form of inefficiency for another.

To remove this overhead while preserving the accuracy gain, we propose **Rollout Res**idual **Q**uantization (**Rollout-ResQ**), which augments the standard FP4 rollout computation with a residual correction term constrained to a hardware-friendly sparsity pattern [42]. The two ingredients are inseparable: the residual, inspired by the decomposition of [65], captures the outlier-driven error that vanilla FP4 rollouts incur, while the imposed sparsity ensures the correction itself runs at a small fraction of a dense FP4 GEMM. The result is a single, lightweight correction added to the FP4 rollout path— preserving accuracy close to the higher-precision-activation alternatives while retaining true FP4 GEMM throughput end-to-end. Our contributions are as follows:

- **First end-to-end FP4 RL post-training.** To our knowledge, we present the first end-to-end FP4 RL post-training in which both the rollout and training policies— including their forward and backward passes—operate at 4-bit precision, without LoRA adapters or higher-precision gradient paths.
- **Diagnosis.** We localize the dominant source of degradation to rollout-side *activation* quantization, where outliers cause the bulk of activation values to underflow under FP4. Restoring only the training policy to higher precision performs *worse* than uniform FP4 training, indicating that the rollout–training mismatch, not training-side noise, is the principal failure mode.
- **Rollout-ResQ.** A single residual correction with a hardware-friendly sparsity pattern, added to the FP4 rollout matmul. The correction recovers most of the accuracy lost to outlier-driven underflow while keeping the rollout path in FP4 GEMMs.
- **Results.** On Qwen2.5-3B and Qwen2.5-Math-7B, Rollout-ResQ with HiF4 closes the BF16 accuracy gap from 4.9% to 1.1%; with MXFP4, from 13.6% to 5.3%. The method is format-agnostic, and HiF4 is the superior format for extreme-precision training.

2 Related Work

Quantization-aware training (QAT) for RL. Prior QAT methods for low-precision RL post-training have mainly framed the problem as a train-rollout policy mismatch, since the rollout policy is quantized (e.g., FP8) while the training policy remains in higher precision (e.g., BF16), addressing this through truncated importance sampling [61, 31], adaptive quantization noise [22], adaptive clipping, and update-aware quantization [27]. JetRL [57] eliminates the mismatch by enforcing a unified low-precision pipeline across both policies. However, none of these methods account for the specific challenges of rollout activation quantization, which our results identify as the key bottleneck.

Scaling-based Methods. A rich body of work addresses outliers in low-precision quantization through mixed-precision protection [11, 26, 24, 12], where a small subset of salient channels or weights is preserved in higher precision to reduce quantization error. AWQ [30] uses activation-aware scaling to protect salient weight channels, SmoothQuant [58] migrates quantization difficulty from activations to weights via an equivalent transformation, and OmniQuant [47] further introduces learnable weight clipping and equivalent transformations. All of these assume a frozen model with offline calibration data, making them ill-suited for RL post-training where activation statistics shift continuously as the policy evolves.

Rotation-based Methods. QuaRot [3], SpinQuant [34], DuQuant [29], and [53] redistribute extreme activation values across channels using rotation or random Hadamard transforms (RHT) to ease quantization. Like scaling-based methods, these transforms are typically fixed or optimized offline for a frozen model, making them less natural for RL post-training's evolving policy and non-iid rollout distribution.

Residual-based Methods. ARCQuant [41] compensates quantization error through augmented residual channels, assuming a static setting where activation statistics can be estimated once. [54]

mitigates activation collapse via an outlier clamping-and-compensation (OCC) strategy, which does not require offline calibration and is thus more applicable to RL post-training. [65] propose a dynamic block-level fallback that augments the quantized activation with a residual term for outlier-detected blocks. Our method likewise separates the outlier-induced residual from the main activation term, but operates on a tensor level and replaces the dense fallback with a sparse residual correction, preserving most of the accuracy gain while substantially reducing FP4 GEMM overhead. Unlike dynamic block fallback, which targets static inference, we adapt a fallback mechanism to fully FP4 RL post-training where the correction must preserve rollout quality without sacrificing end-to-end low-precision efficiency.

3 Preliminaries

Block-Scaled Quantization Modern hardware-friendly 4-bit formats such as MXFP4 [10] and HiF4 [37] utilize block-scaled quantization to maintain numerical precision. Instead of a single per-tensor scale, a tensor is partitioned into blocks of size B where the values in the same block share block or sub-block level scales to improve the representation capacity. MXFP4 [10] uses block-wise microscaling over blocks of 32 elements to balance dynamic range and hardware efficiency. It employs an E8M0 format for the shared block-level scale and an FP4 E2M1 format for the individual elements. This design increases the representable dynamic range through a shared exponent per block, while keeping the per-element representation compact. HiF4 [37] uses a three-level hierarchical scaling scheme over blocks of 64 elements to balance dynamic range and numerical precision. Concretely, it employs an E6M2 format for the first-level scaler, E1 scalers for the second and third levels, and an S1P2 representation for the FP4 values, which is equivalent to E1M2. This hierarchical design increases the representable dynamic range while retaining precision through mantissa bits in both the top-level scaler and the FP4 values.

For a given linear layer ℓ, let $X_\ell \in \mathbb{R}^{m\times d}$ denote the input activation matrix, $W_\ell \in \mathbb{R}^{n\times d}$ the weight matrix, and $Y_\ell = X_\ell W_\ell^\top$ the high-precision output, where m, d, n represent the token count, input dimension, and output dimension, respectively. We denote a quantization configuration with x-bit weights and y-bit activations as $WxAy$. This work focuses on the $W4A4$ setting, where both weights and activations are quantized to FP4 format. Applying this to both the activations and the weights of a linear layer yields the low-precision approximation

$$\hat{Y}_\ell = \hat{X}_\ell \hat{W}_\ell^\top = Q(X_\ell)Q(W_\ell)^\top, \tag{1}$$

where Q is the quantization operator that maps values from high precision to low precision format.

Group Relative Policy Optimization Group Relative Policy Optimization (GRPO) is a value function-free variant of PPO for LLM post-training, where the objective for a training policy π_θ is derived directly from the relative rewards of multiple sampled responses generated by a rollout policy. For a given prompt q from the training dataset, the rollout policy $\mu_{\theta_{\text{old}}}$ samples a group of G responses $\{o_1, \ldots, o_G\}$, where $o_i \sim \mu_{\theta_{\text{old}}}(\cdot \mid q)$. Each response o_i receives a scalar reward based on task correctness or a verifiable objective. To evaluate the current action, GRPO computes a group-relative advantage by normalizing rewards within the sampled group, eliminating the need to fit a value function. The GRPO objective is:

$$J(\theta) = \mathbb{E}_{q,o_i\sim\mu_{\theta_{\text{old}}}(\cdot|q)}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\min\big(R_{i,t}A_{i,t},\ \text{clip}(R_{i,t}, 1-\epsilon_{\text{low}}, 1+\epsilon_{\text{high}})A_{i,t}\big)\right]. \tag{2}$$

where $A_{i,t}$ is the token-level advantage derived from the rewards, $R_{i,t} = \frac{\pi_\theta(o_{i,t}|q,o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q,o_{i,<t})}$ is the importance sampling ratio , π_θ and $\pi_{\theta_{\text{old}}}$ are the target and reference policies respectively, ϵ_{low} and ϵ_{high} are the clipping parameters. Note that, the objective also contains a KL penalty term but is omitted here for simplicity.

4 Method

4.1 Problem Setting

Following previous works [22, 27, 57], we consider the standard Markov decision process (MDP) setting for LLMs post-training, where generation is viewed as a sequential decision-making problem.

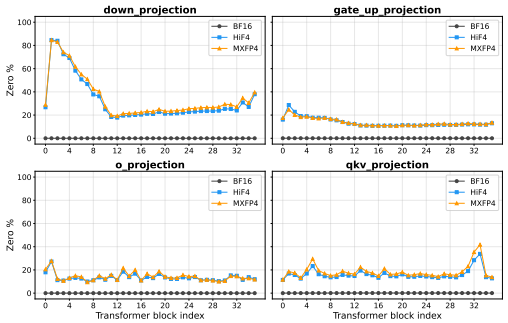


Figure 1: Percentage of zero values in activation tensors of each layer before quantization (BF16) and after quantization (HiF4, MXFP4) with GSM8K and Qwen2.5-3B.

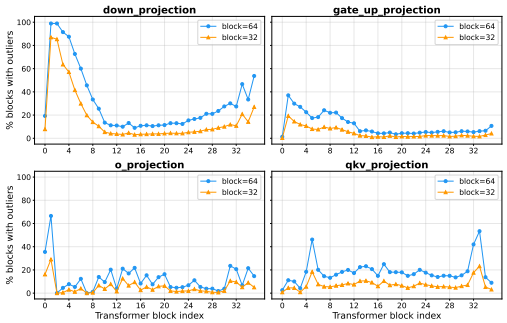


Figure 2: Percentage of blocks with outliers in activation tensors of each layer with GSM8K and Qwen2.5-3B for HiF4 and MXFP4 block size.

At each timestep, the state consists of the prompt together with the previously generated tokens (prefix), the action is the next token, and the episode ends when an end-of-sequence (EOS) token is produced or a maximum response length is reached.

Our focus is fully quantized RL post-training, where both rollout and training are performed using quantized policies. We study this setting in the GRPO-based post-training framework implemented in VeRL [49], using HiF4 [37] and MXFP4 [10] as the FP4 instantiations of Q.

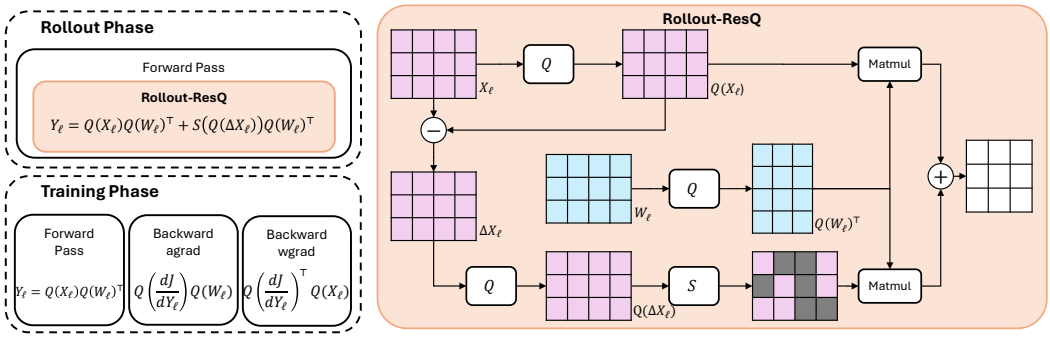
4.2 The Cause of Performance Degradation in FP4 RL

Table 1 presents a comparison of training results for Qwen2.5-3B on GSM8K across different precision configurations for training and rollout. In the full BF16 baseline (Row 1), the model achieves 86.96% accuracy. When both training and rollout are in HiF4 (Row 5), the accuracy drops to 82.03%. This degradation is even more pronounced for MXFP4, where accuracy falls to 73.31%. These results raise a fundamental question for low-precision RL — is the loss in performance driven by training quantization or rollout quantization? To identify the source of this degradation, we compare the performance of applying quantization to specific components. Our result indicates that the gap between full BF16 RL and full FP4 RL is primarily driven by rollout quantization. Specifically, training in HiF4 or MXFP4 while maintaining a BF16 rollout (Row 2) yields accuracies of 86.88% and 86.20%, respectively — matching the full BF16 baseline. In other words, by simply keeping rollout in high precision, the accuracy is recovered to nearly fully high precision level. Conversely, applying FP4 quantization to the rollout phase while keeping the training in BF16 (Row 6) results in a catastrophic drop to 78.01% for HiF4 and 50.57% for MXFP4, performing significantly worse than even the fully quantized settings. We observe that the KL divergence between the inference and training policies throughout the RL training is substantially higher than all the other settings (see Appendix E), indicating that policy mismatch is the primary mode of failure if only training is in BF16. Furthermore, by isolating the components of the rollout phase, we identify activation quantization as the primary bottleneck. While weight-only quantization (W4A16) in the rollout (Row 3) maintains a competitive accuracy of 85.22% for HiF4 and 81.57% for MXFP4, activation-only quantization (W16A4, Row 4) leads to a sharper decline to 84.30% for HiF4 and 77.71% for MXFP4. These findings reveal that, in low-precision RL, the dominant source of model degradation is the quantization error of activations during rollout, rather than the precision loss due to quantization during training.

		Accuracy (%)	
Train	Rollout	FP4 = HiF4	FP4 = MXFP4
BF16	BF16	86.96	
FP4	BF16	86.88	86.20
FP4	W4A16	85.22	81.57
FP4	W16A4	84.30	77.71
FP4	FP4	82.03	73.31
BF16	FP4	78.01	50.57

Table 1: Accuracy on GSM8K test set for Qwen2.5-3B with GRPO under different training and rollout precisions. FP4 formats include HiF4 and MXFP4, with W4A16 and W16A4 denoting weight-only and activation-only quantization, respectively.

To further understand the issue with rollout activation, we analyze the distribution of the rollout activation tensors. Figure 1 illustrates the percentage of zero values within activation tensors across



(a) FP4 RL recipe with Rollout-ResQ (b) The computation of Rollout-ResQ for rollout linear projections.

Figure 3: Rollout-ResQ overview.

layers. While the original BF16 tensors are dense, post-quantization tensors (HiF4, MXFP4) show a dramatic increase in sparsity. In specific layers, such as the down projection of transformer block 1 and 2, sparsity exceeds 80%, indicating a "zero collapse" where the majority of non-zero values are mapped to zero. This phenomenon is driven by the presence of outliers, as emphasized in [54] and [65]. As shown in Figure 2, the percentage of blocks containing outliers strongly correlates with the severity of zero collapse in the activation tensor. During quantization, since the block scale factor is governed by the large values in a block, outliers can lead to enormous scale factors that collapse the non-outliers to zeros. Consequently, the tensor's distribution collapses, resulting in significant information loss and precision degradation.

4.3 Rollout Residual Quantization (Rollout-ResQ) for FP4 RL

We present Rollout-ResQ, an effective strategy for recovering precision loss caused by activation outliers during rollout in fully FP4 RL. As shown in Figure 3a, Rollout-ResQ is used in the rollout phase of our fully FP4 RL training recipe, augmenting the standard FP4 matrix multiplication in the linear projection layers of all transformer blocks with a residual correction term. Figure 3b illustrates this design in detail. During rollout, for each linear layer ℓ, the Rollout-ResQ takes as inputs the activation tensor X_ℓ and weight tensor W_ℓ and computes an activation residual,

$$\Delta X_\ell = X_\ell - Q(X_\ell), \tag{3}$$

where Q is a quantization method. The residual captures the activation quantization error which can then be quantized to recover the precision loss through two FP4 GEMMs,

$$\hat{Y}_\ell = Q(X_\ell)Q(W_\ell)^\top + Q(\Delta X_\ell)Q(W_\ell)^\top, \tag{4}$$

Note that, unlike [65] which computes residuals at block level, this formulation operates on a tensor level, which can be implemented more efficiently with modern deep learning framework. However, this approach still requires an extra dense matrix multiplication which can incur substantial computation cost. To maintain computation efficiency, Rollout-ResQ utilizes a sparsity function S, which is applied to the quantized activation residual $Q(\Delta X_\ell)$. The resulting rollout projection is defined as:

$$\hat{Y}_\ell = Q(X_\ell)Q(W_\ell)^\top + S(Q(\Delta X_\ell))Q(W_\ell)^\top, \tag{5}$$

By integrating these components, Equation 5 preserves the efficiency of the standard FP4 projection while introducing a lightweight, sparse matrix multiplication, which is intended to reduce computation and can provide faster execution, especially when its granularity matches hardware-supported sparse kernels [15, 2, 44, 35]. We instantiate S using different sparsity functions: (1) structured channel-wise sparsity, denoted by $S_{\alpha\%}$, which retains only the top $\alpha\%$ of channels with the largest L2-norm; (2) semi-structured sparsity, denoted by $S_{M:N}$, which retains only the M largest magnitude elements within every contiguous block of N elements; and (3) block sparsity, denoted by $S^{M\times N}_{\alpha\%}$, which retains the top $\alpha\%$ of blocks of size $M\times N$ with largest L2-norm. These sparsity functions are motivated by prior work showing the practical benefits of channel-wise, semi-structured, and block sparsity for efficient matrix multiplication [6, 2, 44, 35, 42, 18, 14, 38]. In our experiments, we choose $\alpha = 50$, a commonly used pruning level in prior work [6]. We choose $S_{2:4}$ since it is supported by

modern sparse kernels and offers a trade-off between acceleration and accuracy [21, 18], and has been shown to be able to achieve a speedup of 1.3-1.5x [39, 18]. Finally, for block sparsity we choose $S_{50\%}^{32\times64}$ where half of the 32×64 blocks are kept. In all three cases, we sparsify $Q(\Delta X_\ell)$ on a 50% level. We denote the channel-wise, semi-structured, and block-sparse variants of Rollout-ResQ as Rollout-ResQ-$S_{50\%}$, Rollout-ResQ-$S_{2:4}$, and Rollout-ResQ-$S_{50\%}^{32\times64}$, respectively. Note that setting S to the identity function, recovers Equation 4; we refer to this method as Rollout-ResQ-Dense.

The choice to sparsify the residual rather than the primary activation is motivated by their robustness to sparsification. We measure MSE and RMSE of $Q(X_\ell)$ and $Q(\Delta X_\ell)$ with their corresponding $S_{2:4}$ sparsified versions. As illustrated in Figure 4, our empirical measurements confirm that $Q(\Delta X_\ell)$ is more robust to $S_{2:4}$, incurring smaller error, making it an ideal candidate for hardware-friendly acceleration.

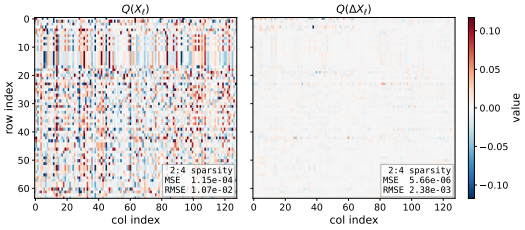


Figure 4: Heatmap of quantized activation and residual during rollout. The MSE and RMSE before and after performing 2:4 sparsity.

5 Experiments

5.1 Experimental Setup

We conduct all experiments using the training framework via VeRL [49], a hybrid-engine architecture that enables different execution engines to be combined for improved hardware efficiency. In our experiments, we use vLLM for the rollout policy and FSDP for the training policy, with GRPO as the RL backbone. We study two FP4 quantization format, HiF4 and MXFP4, applying quantization in the linear layers of all transformer blocks. We train Qwen2.5-3B [45] on GSM8K [8], and Qwen2.5-Math-7B [60] on DAPO-Math-17K [62] which is a more challenging task. See Appendix A for additional details.

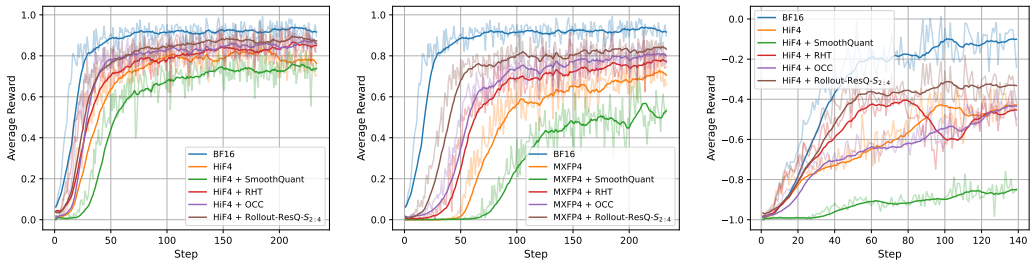
Format	Recovery Method	Accuracy	
		GSM8K-test	Math500
BF16	–	86.96	61.2
HiF4	–	82.03 (−4.93)	44.6 (−16.6)
	SmoothQuant	78.54 (−8.42)	39.0 (−22.2)
	RHT	82.87 (−4.09)	41.0 (−20.2)
	OCC	84.23 (−2.73)	51.8 (−9.4)
	Rollout-ResQ-Dense	85.52 (−1.44)	53.0 (−8.2)
	Rollout-ResQ-$S_{50\%}$	84.61 (−2.35)	52.6 (−8.6)
	Rollout-ResQ-$S_{50\%}^{32\times64}$	84.61 (−2.35)	**55.8 (−5.4)**
	Rollout-ResQ-$S_{2:4}$	**85.90 (−1.06)**	52.6 (−8.6)
MXFP4	–	73.31 (−13.65)	27.8 (−33.4)
	SmoothQuant	64.22 (−22.74)	23.6 (−37.6)
	RHT	78.01 (−8.95)	39.6 (−21.6)
	OCC	78.70 (−8.26)	42.8 (−18.4)
	Rollout-ResQ-Dense	**81.65 (−5.31)**	**48.2 (−13.0)**
	Rollout-ResQ-$S_{50\%}$	80.82 (−6.14)	42.8 (−18.4)
	Rollout-ResQ-$S_{50\%}^{32\times64}$	80.36 (−6.60)	45.6 (−15.6)
	Rollout-ResQ-$S_{2:4}$	**81.65 (−5.31)**	46.4 (−14.8)

Table 2: Results for Qwen2.5-3B trained on GSM8K with GRPO. Values in parentheses show gaps to BF16.

For evaluation, we use several widely used mathematical reasoning benchmarks. For Qwen2.5-3B, we report results on held-out GSM8K-test samples and on Math-500 [19]. For Qwen2.5-Math-7B, we report results on AIME-2024 [23], AIME-2025 [40], AMC-2023 [43], and Math-500 [19]. We report Mean@1 for Qwen2.5-3B and Mean@32 for Qwen2.5-Math-7B, where Mean@1 denotes accuracy under greedy decoding and Mean@32 denotes average accuracy over 32 sampled responses. We compare all the variants of Rollout-ResQ against several baselines: (1) BF16, which uses full-precision BF16 for both rollout and training; (2) HiF4 and MXFP4, which quantizes both rollout and training using the corresponding quantization method without any rollout recovery method; (3) HiF4 and MXFP4 with SmoothQuant [58], RHT [53] and OCC [54] applied to rollout activations. In SmoothQuant experiments, the smoothing scales are calibrated at the start of each RL iteration.

5.2 GSM8K Training Results

Table 2 reports Mean@1 accuracies of training 3B model on GSM8K. Naively applying HiF4 and MXFP4 to low-precision RL training degrades performance by 4.93% and 13.65% on GSM8K-test, and 16.6% and 33.4% on MATH500, respectively, relative to BF16. Among the baselines, OCC [54]



(a) Qwen2.5-3B under HiF4 (b) Qwen2.5-3B under MXFP4 (c) Qwen2.5-Math-7B under HiF4

Figure 5: Training performance (average reward) for Qwen2.5-3B on GSM8K under HiF4 and MXFP4, and for Qwen2.5-Math-7B on DAPO-Math-17K under HiF4.

performs better, but still leaves gaps of 2.73% and 8.26% on GSM8K-test, and 9.4% and 18.4% on Math500, under HiF4 and MXFP4, respectively. By contrast, Rollout-ResQ-$S_{2:4}$ reduces these gaps to 1.06% and 5.31% on GSM8K-test for HiF4 and MXFP4, respectively. On Math500, under HiF4, Rollout-ResQ-$S_{50\%}^{32\times64}$ achieves the highest accuracy (55.8%), followed by other Rollout-ResQ variants. Under MXFP4, Rollout-ResQ-Dense leads (48.2%), followed by Rollout-ResQ-$S_{2:4}$(46.4%) and Rollout-ResQ-$S_{50\%}$(45.6%). Across both formats, all Rollout-ResQ variants outperform all baselines and close the gap to BF16 significantly. Overall, these results show that Rollout-ResQ substantially outperforms the naive-FP4 baseline on both formats and closes the accuracy gap to BF16 more effectively than other baseline methods. The consistent advantage of Rollout-ResQ over all baselines under both HiF4 and MXFP4 confirms that the sparse residual correction is format-agnostic. In all settings, HiF4 establishes a significantly higher baseline and recovery ceiling than MXFP4 due to its hierarchical scaling, revealing that HiF4 is the more robust format for end-to-end FP4 RL post-training. And Rollout-ResQ acts as the crucial mechanism for closing the remaining gap to BF16 across both formats – providing substantial gains even on the weaker format (MXFP4). Figure 5a and 5b present the training average reward for the 3B model on HiF4 and MXFP4, respectively. The figure reports the best-performing variant of Rollout-ResQ, i.e., Rollout-ResQ-$S_{2:4}$, during the training. We make the following observations: (1) Under both HiF4 and MXFP4, SmoothQuant exhibits slow and suboptimal convergence, which we attribute to its reliance on accurate calibration scales — a requirement that is hard to satisfy under RL's non-stationary rollout distribution. (2) RHT and OCC consistently outperform SmoothQuant, including the early stage of training. (3) Under both HiF4 and MXFP4, Rollout-ResQ-$S_{2:4}$ consistently outperforms the baselines across training and achieves substantially faster convergence; (4) HiF4 appears to be more compatible with both Rollout-ResQ-$S_{2:4}$ and the baselines, as MXFP4 leads to less stable training dynamics across methods.

We also performed multi-seed experiments for BF16, HiF4, OCC, and Rollout-ResQ-$S_{2:4}$ to confirm the stability of relative effectiveness of these methods. Table 3 reports the means and sample standard deviations of GSM8K-test and Math500 accuracy for 3-seed runs of BF16, HiF4, OCC, and Rollout-ResQ-$S_{2:4}$ for Qwen2.5-3B with GSM8K training. The results are consistent with our 1-seed experiments, where the HiF4 baseline exhibits the most degradation (-3.59% on GSM8K-test, -13.87% on Math500), and Rollout-ResQ-$S_{2:4}$ (-1.39% on GSM8K-test, -6.94% on Math500) improves over OCC (-2.13% on GSM8K-test, -8.80% on Math500), with a clear gain on Math500 and a modest gain on GSM8K-test.

	GSM8K-test		Math500	
Qwen2.5-3B	Mean	Std.	Mean	Std.
BF16	86.56	0.42	60.47	1.63
HiF4	82.97 (-3.59)	0.88	46.60 (-13.87)	2.11
OCC	84.43 (-2.13)	0.41	51.67 (-8.80)	0.42
Rollout-ResQ-$S_{2:4}$	85.17 (-1.39)	0.64	53.53 (-6.94)	0.81

Table 3: Results of Qwen2.5-3B trained on GSM8K with GRPO over 3 seeds for BF16, HiF4, OCC and Rollout-ResQ-$S_{2:4}$ in the HiF4 context. The mean and sample standard deviation of GSM8K-test and Math500 are reported. Values in parentheses show gaps to BF16.

5.3 DAPO-Math-17K Training Results

We focus on HiF4 for the 7B model, as the 3B results in Section 5.2 indicate that HiF4 is a more reliable format for FP4 RL. Table 4 reports Mean@32 accuracies of training Math-7B model on DAPO-Math-17K. The table shows that naive HiF4 incurs drops of 11.14%, 4.79%, 9.37% and 15.19%, relative to full-precision BF16. Consistent with the previous section, SmoothQuant performs poorly in this setting. In contrast, Rollout-ResQ-$S_{2:4}$ narrows the gaps to BF16 to 6.56%, 3.54%, 8.75%, and 5.64%, respectively, with Rollout-ResQ-Dense performs slightly better on AIME25 and Math500. Overall, Rollout-ResQ-$S_{2:4}$ offers the most consistent improvement across all benchmarks. These results highlight both the effectiveness of Rollout-ResQ-$S_{2:4}$ and its generalization to diverse math benchmarks. Figure 5c shows the training average reward of the Math-7B model. We make the following observations: (1) Most baseline methods exhibit either unstable behavior or slow convergence. In contrast, Rollout-ResQ-$S_{2:4}$ remains relatively stable throughout training and achieves higher performance; (2) Although RHT is competitive with Rollout-ResQ-$S_{2:4}$ early in training, its reward begins to deteriorate later on, indicating less stable training dynamics; (3) OCC shows a similar training progress to naive HiF4, but this does not translate to downstream performance — exhibiting a significantly drop-off from HiF4 in all four benchmarks in Table 4. (4) Rollout-ResQ-$S_{2:4}$ reaches a higher reward level earlier than baselines and maintains it more consistently, exhibiting better convergence speed and stability.

Precision	Recovery Method	Accuracy (Mean@32)			
		AIME24	AIME25	AMC23	Math500
BF16	–	27.29	12.08	64.53	74.59
HiF4	–	16.15 (−11.14)	7.29 (−4.79)	55.16 (−9.37)	59.40 (−15.19)
	SmoothQuant	0.70 (−26.59)	0.10 (−11.98)	23.36 (−41.17)	53.51 (−21.08)
	RHT	13.43 (−13.86)	8.64 (−3.44)	53.75 (−10.78)	64.67 (−9.92)
	OCC	6.77 (−20.52)	4.90 (−7.18)	41.33 (−23.20)	43.59 (−31.00)
	Rollout-ResQ-Dense	14.48 (−12.81)	**8.96 (−3.12)**	55.62 (−8.91)	**69.02 (−5.57)**
	Rollout-ResQ-$S_{50\%}$	13.44 (−14.46)	7.60 (−4.48)	43.51 (−21.01)	64.30 (−10.29)
	Rollout-ResQ-$S^{32\times64}_{50\%}$	14.06 (−13.84)	8.33 (−3.75)	53.28 (−11.25)	47.89 (−26.70)
	Rollout-ResQ-$S_{2:4}$	**20.73 (−6.56)**	8.54 (−3.54)	**55.78 (−8.75)**	68.95 (−5.64)

Table 4: Results for Qwen2.5-Math-7B trained on DAPO-Math-17K with GRPO. Parentheses show gaps to the BF16 baseline.

5.4 Overhead Analysis

We analyze the theoretical computational overhead introduced by Rollout-ResQ, specifically Rollout-ResQ-$S_{2:4}$. For a linear layer with input activation $X \in \mathbb{R}^{m\times d}$ and weight matrix $W \in \mathbb{R}^{n\times d}$, a dense FP16/BF16 GEMM for computing $Y = XW^\top$ has a FLOP cost of $2mdn$. In Rollout-ResQ-$S_{2:4}$, the rollout projection is computed as the sum of a dense GEMM and a sparse residual GEMM. Under the $S_{2:4}$ sparsity function, only half of the residual activation elements are retained, so the number of nonzero entries is $\frac{1}{2}md$. Multiplying this sparse residual by $W^\top$ costs mdn FLOPs. Assuming FP4 execution provides a $4\times$ throughput advantage over FP16 on supported hardware [1, 7, 51, 16], the effective costs of the dense and sparse GEMMs become $\frac{1}{2}mdn$ and $\frac{1}{4}mdn$, respectively. Hence, the total effective compute cost of Rollout-ResQ-Dense is $\frac{1}{2}mdn + \frac{1}{2}mdn = mdn$ and Rollout-ResQ-$S_{2:4}$ is $\frac{1}{2}mdn + \frac{1}{4}mdn = \frac{3}{4}mdn$. Relative to FP16, while Rollout-ResQ-Dense achieves a theoretical speedup of $\frac{2mdn}{mdn} = 2\times$, Rollout-ResQ-$S_{2:4}$ can achieve up to $\frac{2mdn}{3mdn/4} \approx 2.67\times$ speedup. Overall, Rollout-ResQ-$S_{2:4}$ and its variants introduce minimal overhead, which is critical for preserving FP4 efficiency. We provide more analysis on the memory overhead of Rollout-ResQ-$S_{2:4}$ in Appendix B.

6 Limitations and Future Directions

Due to the lack of native FP4 hardware support, all experiments rely on FP4 simulations[2] where simulation overhead dominates training time, preventing direct measurement of real speedup and efficiency gains. Additionally, the prolonged runtime has prevented us from extending experiments to

[2]The simulated quantization code is publicly available at `https://github.com/global-computing-consortium/HiFloat4`.

extremely large models or datasets, which remain important directions for future work. Furthermore, this work focuses exclusively on 50% sparsity across all sparsity patterns, with the $S_{2:4}$ variant driven by modern hardware support for semi-structured sparsity; exploring varying sparsity levels to better characterize the accuracy-efficiency trade-off remains an open direction.

7 Conclusion

This work presents the first end-to-end FP4 RL post-training for LLMs, with both rollout and training policies operating at 4-bit precision. We show that the central obstacle is not the training pass but the rollout: activation outliers lead to unwanted underflow in FP4 setting, degrading rollout quality and learning outcomes. To recover accuracy without significantly compromising compute efficiency, we proposed Rollout-ResQ — a sparse residual correction appended to the FP4 rollout matrix multiplication that neutralizes outlier-driven error while preserving all GEMMs in low precision. Rollout-ResQ substantially closes the gap on Qwen2.5-3B and Qwen2.5-Math-7B across benchmarks, outperforming all outlier-mitigation baselines. We further show empirically that HiF4 is the recommended format for end-to-end FP4 RL, achieving a substantially tighter gap to full precision than MXFP4. These results demonstrate that robust fully FP4 RL post-training is possible when the dominant source of error — rollout activation quantization — is sufficiently addressed, and that format choice plays a key role in determining the recoverable accuracy.

References

[1] AmirAli Abdolrashidi, Lisa Wang, Shivani Agrawal, Jonathan Malmaud, Oleg Rybakov, Chas Leichner, and Lukasz Lew. Pareto-optimal quantized resnet is mostly 4-bit. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3085–3093. IEEE, 2021.

[2] Sajid Anwar, Kyuyeon Hwang, and Wonyong Sung. Structured pruning of deep convolutional neural networks. *J. Emerg. Technol. Comput. Syst.*, 13(3), February 2017. ISSN 1550-4832. doi: 10.1145/3005348. URL https://doi.org/10.1145/3005348.

[3] Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. Quarot: Outlier-free 4-bit inference in rotated llms. *Advances in Neural Information Processing Systems*, 37:100213–100240, 2024.

[4] Xiaohe Bo, Zeyu Zhang, Quanyu Dai, Xueyang Feng, Lei Wang, Rui Li, Xu Chen, and Ji-Rong Wen. Reflective multi-agent collaboration based on large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=wWiAR5mqXq.

[5] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 2020.

[6] Minsik Cho, Saurabh Adya, and Devang Naik. PDP: Parameter-free differentiable pruning is all you need. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=lLztVBaBVU.

[7] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018.

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

[9] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830*, 2016.

[10] Bita Darvish Rouhani, Ritchie Zhao, Ankit More, Mathew Hall, Alireza Khodamoradi, Summer Deng, Dhruv Choudhary, Marius Cornea, Eric Dellinger, Kristof Denolf, Stosic Dusan, Venmugil Elango, Maximilian Golub, Alexander Heinecke, Phil James-Roxby, Dharmesh Jani, Gaurav Kolhe, Martin Langhammer, Ada Li, Levi Melnick, Maral Mesmakhosroshahi, Andres Rodriguez, Michael Schulte, Rasoul Shafipour, Lei Shao, Michael Siu, Pradeep Dubey, Paulius Micikevicius, Maxim Naumov, Colin Verrilli, Ralph Wittig, Doug Burger, and Eric Chung. Microscaling data formats for deep learning. October 2023. URL https://www.microsoft.com/en-us/research/publication/microscaling-data-formats-for-deep-learning/.

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

[12] Tim Dettmers, Ruslan A. Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. SpQR: A sparse-quantized representation for near-lossless LLM weight compression. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Q1u25ahSuy.

[13] David Dinucu-Jianu, Jakub Macina, Nico Daheim, Ido Hakimi, Iryna Gurevych, and Mrinmaya Sachan. From problem-solving to teaching problem-solving: Aligning llms with pedagogy using reinforcement learning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 272–292, 2025.

[14] Chao Fang, Aojun Zhou, and Zhongfeng Wang. An Algorithm–Hardware Co-Optimized Framework for Accelerating N:M Sparse Transformers . *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 30(11):1573–1586, November 2022. ISSN 1557-9999. doi: 10.1109/TVLSI.2022.3197282. URL https://doi.ieeecomputersociety.org/10.1109/TVLSI.2022.3197282.

[15] Trevor Gale, Deepak Narayanan, Cliff Young, and Matei Zaharia. Megablocks: Efficient sparse training with mixture-of-experts. In D. Song, M. Carbin, and T. Chen, editors, *Proceedings of Machine Learning and Systems*, volume 5, pages 288–304. Curran, 2023. URL https://proceedings.mlsys.org/paper_files/paper/2023/file/5a54f79333768effe7e8927bcccffe40-Paper-mlsys2023.pdf.

[16] Yubo Gao, Renbo Tu, Gennady Pekhimenko, and Nandita Vijaykumar. Dpquant: Efficient and private model training via dynamic quantization scheduling. In *International Conference on Learning Representations*, volume 2026, pages 157109–157131, 2026.

[17] Avelina Asada Hadji-Kyriacou and Ognjen Arandjelovic. Would i lie to you? inference time alignment of language models using direct preference heads. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=NKGuLthW80.

[18] Daniel HAZIZA, Timothy Chou, Dhruv Choudhary, Jesse Cai, Luca Wehrstedt, Francisco Massa, Jiecao Yu, Geonhwa Jeong, Supriya Rao, and Patrick Labatut. Accelerating transformer inference and training with 2:4 activation sparsity. In *Sparsity in LLMs (SLLM): Deep Dive into Mixture of Experts, Quantization, Hardware, and Inference*, 2025. URL https://openreview.net/forum?id=O5feVk7p6Y.

[19] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=7Bywt2mQsCe.

[20] Jian Hu, Xibin Wu, Wei Shen, Jason Klein Liu, Weixun Wang, Songlin Jiang, Haoran Wang, Hao Chen, Bin Chen, Wenkai Fang, Xianyu, Yu Cao, Haotian Xu, and Yiming Liu. OpenRLHF: A ray-based easy-to-use, scalable and high-performance RLHF framework. In Ivan Habernal, Peter Schulam, and Jörg Tiedemann, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 656–666, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-334-0. doi: 10.18653/v1/2025.emnlp-demos.48. URL https://aclanthology.org/2025.emnlp-demos.48/.

[21] Yuezhou Hu, Kang Zhao, Weiyu Huang, Jianfei Chen, and Jun Zhu. Accelerating transformer pre-training with 2:4 sparsity. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

[22] Wei Huang, Yi Ge, Shuai Yang, Yicheng Xiao, Huizi Mao, Yujun Lin, Hanrong Ye, Sifei Liu, Ka Chun Cheung, Hongxu Yin, Yao Lu, Xiaojuan Qi, Song Han, and Yukang Chen. QeRL: Beyond efficiency - quantization-enhanced reinforcement learning for LLMs. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=zw8zxMJJlm.

[23] Maxwell Jia. Aime_2024. Hugging Face dataset, 2024. https://huggingface.co/datasets/Maxwell-Jia/AIME_2024.

[24] Sehoon Kim, Coleman Richard Charles Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W. Mahoney, and Kurt Keutzer. SqueezeLLM: Dense-and-sparse quantization. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=0jpbpFia8m.

[25] Md Tahmid Rahman Laskar, Sawsan Alqahtani, M Saiful Bari, Mizanur Rahman, Mohammad Abdullah Matin Khan, Haidar Khan, Israt Jahan, Amran Bhuiyan, Chee Wei Tan, Md Rizwan Parvez, et al. A systematic survey and critical review on evaluating large language models: Challenges, limitations, and recommendations. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 13785–13816, 2024.

[26] Changhun Lee, Jungyu Jin, Taesu Kim, Hyungjun Kim, and Eunhyeok Park. Owq: Outlier-aware weight quantization for efficient fine-tuning and inference of large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 13355–13364, 2024.

[27] Yuhang Li, Reena Elangovan, Xin Dong, Priyadarshini Panda, and Brucek Khailany. QuRL: Low-precision reinforcement learning for efficient reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=eG0bpCwdKn.

[28] Mengqi Liao, Xiangyu Xi, Chen Ruinian, Jia Leng, Yangen Hu, Ke Zeng, Shuai Liu, and Huaiyu Wan. Enhancing efficiency and exploration in reinforcement learning for llms. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 1451–1463, 2025.

[29] Haokun Lin, Haobo Xu, Yichen Wu, Jingzhi Cui, Yingtao Zhang, Linzhan Mou, Linqi Song, Zhenan Sun, and Ying Wei. Duquant: Distributing outliers via dual transformation makes stronger quantized LLMs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=mp8u2Pcmqz.

[30] 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.

[31] Liyuan Liu, Feng Yao, Dinghuai Zhang, Chengyu Dong, Jingbo Shang, and Jianfeng Gao. Flashrl: 8bit rollouts, full power rl, August 2025. URL https://fengyao.notion.site/flash-rl. Preprint.

[32] Shih-Yang Liu, Xin Dong, Ximing Lu, Shizhe Diao, Peter Belcak, Mingjie Liu, Min-Hung Chen, Hongxu Yin, Yu-Chiang Frank Wang, Kwang-Ting Cheng, et al. Gdpo: Group reward-decoupled normalization policy optimization for multi-reward rl optimization. *arXiv preprint arXiv:2601.05242*, 2026.

[33] Wei Liu, Siya Qi, Xinyu Wang, Chen Qian, Yali Du, and Yulan He. Nover: Incentive training for language models via verifier-free reinforcement learning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 7450–7469, 2025.

[34] Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. Spinquant: LLM quantization with learned rotations. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=ogO6DGE6FZ.

[35] Zhi Gang Liu, Paul N. Whatmough, Yuhao Zhu, and Matthew Mattina. S2TA: exploiting structured sparsity for energy-efficient mobile CNN acceleration. In *IEEE International Symposium on High-Performance Computer Architecture, HPCA 2022, Seoul, South Korea, April 2-6, 2022*, pages 573–586. IEEE, 2022. doi: 10.1109/HPCA53966.2022.00049. URL https://doi.org/10.1109/HPCA53966.2022.00049.

[36] Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=5PAF7PAY2Y.

[37] Yuanyong Luo, Jing Huang, Yu Cheng, Ziwei Yu, Kaihua Zhang, Kehong Hong, Xinda Ma, Xin Wang, Anping Tong, Guipeng Hu, et al. Hifloat4 format for language model inference. *arXiv preprint arXiv:2602.11287*, 2026.

[38] Haoyu Ma, Chengming Zhang, lizhi xiang, Xiaolong Ma, Geng Yuan, Wenkai Zhang, Shiwei Liu, Tianlong Chen, Dingwen Tao, Yanzhi Wang, Zhangyang Wang, and Xiaohui Xie. Hrbp: Hardware-friendly regrouping towards block-based pruning for sparse cnn training. In Yuejie Chi, Gintare Karolina Dziugaite, Qing Qu, Atlas Wang Wang, and Zhihui Zhu, editors, *Conference on Parsimony and Learning*, volume 234 of *Proceedings of Machine Learning Research*, pages 282–301. PMLR, 03–06 Jan 2024. URL https://proceedings.mlr.press/v234/ma24a.html.

[39] Meghana Madhyastha, Daniel Haziza, Jesse Cai, Newsha Ardalani, Zhiqi Bu, and Carole-Jean Wu. To 2: 4 sparsity and beyond: Neuron-level activation function to accelerate llm pre-training. *arXiv preprint arXiv:2602.06183*, 2026.

[40] MathArena. Aime 2025. Hugging Face dataset, 2025. https://huggingface.co/datasets/MathArena/aime_2025.

[41] Haoqian Meng, Yilun Luo, Yafei Zhao, Wenyuan Liu, Peng Zhang, and Xindian Ma. Arcquant: Boosting nvfp4 quantization with augmented residual channels for llms. *arXiv preprint arXiv:2601.07475*, 2026.

[42] Asit Mishra, Jorge Albericio Latorre, Jeff Pool, Darko Stosic, Dusan Stosic, Ganesh Venkatesh, Chong Yu, and Paulius Micikevicius. Accelerating sparse deep neural networks, 2021. URL https://arxiv.org/abs/2104.08378.

[43] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

[44] Jeff Pool and Chong Yu. Channel permutations for n:m sparsity. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 13316–13327. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/6e8404c3b93a9527c8db241a1846599a-Paper.pdf.

[45] Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

[46] Bita Darvish Rouhani, Nitin Garegrat, Tom Savell, Ankit More, Kyung-Nam Han, Mathew Zhao, Ritchie Hall, Jasmine Klar, Eric Chung, Yuan Yu, Michael Schulte, Ralph Wittig, Ian Bratt, Nigel Stephens, Jelena Milanovic, John Brothers, Pradeep Dubey, Marius Cornea, Alexander Heinecke, Andres Rodriguez, Martin Langhammer, Summer Deng, Maxim Naumov, Paulius Micikevicius, Michael Siu, and Colin Verrilli. Ocp microscaling (mx) specification. Technical report, Open Compute Project, 2023.

[47] Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. Omniquant: Omnidirectionally calibrated quantization for large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=8Wuvhh0LYW.

[48] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[49] Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

[50] Mingyang Song, Mao Zheng, Zheng Li, Wenjie Yang, Xuan Luo, Yue Pan, and Feng Zhang. Fastcurl: Curriculum reinforcement learning with stage-wise context scaling for efficient training r1-like reasoning models. *arXiv preprint arXiv:2503.17287*, 2025.

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

[52] Mehran Taghian, Yunke Peng, Xing Huang, Yao Wang, Yaoyuan Wang, Wei Guo, Yuanyong Luo, Tianchi Hu, Junsong Wang, Xin Wang, Hu Liu, Yu Cheng, Ziwei Yu, Hongliang Li, Mehdi Rahimifar, Lei Yan, Xuefei Wang, Zhuang Ma, Lei Liu, Hui Yu, Anandharaju Durai Raju, Hoang Le, Hei Yi Mak, Tanzila Rahman, and Shadan Golestan. Hifloat4 format for language model pre-training on ascend npus, 2026. URL https://arxiv.org/abs/2604.08826.

[53] Albert Tseng, Tao Yu, and Youngsuk Park. Training llms with mxfp4, 2025. URL https://arxiv.org/abs/2502.20586.

[54] Ruizhe Wang, Yeyun Gong, Xiao Liu, Guoshuai Zhao, Ziyue Yang, Baining Guo, Zheng-Jun Zha, and Peng CHENG. Optimizing large language model training using FP4 quantization. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=uK7JArZEJM.

[55] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf.

[56] Zhepei Wei, Wenlin Yao, Yao Liu, Weizhi Zhang, Qin Lu, Liang Qiu, Changlong Yu, Puyang Xu, Chao Zhang, Bing Yin, et al. Webagent-r1: Training web agents via end-to-end multi-turn reinforcement learning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 7920–7939, 2025.

[57] Haocheng Xi, Charlie Ruan, Peiyuan Liao, Yujun Lin, Han Cai, Yilong Zhao, Shuo Yang, Kurt Keutzer, Song Han, and Ligeng Zhu. Jet-rl: Enabling on-policy fp8 reinforcement learning with unified training and rollout precision flow. *arXiv preprint arXiv:2601.14243*, 2026.

[58] 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*, pages 38087–38099. PMLR, 2023.

[59] Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, and Chong Luo. Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning. *arXiv preprint arXiv:2502.14768*, 2025.

[60] An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement, 2024. URL https://arxiv.org/abs/2409.12122.

[61] Feng Yao, Liyuan Liu, Dinghuai Zhang, Chengyu Dong, Jingbo Shang, and Jianfeng Gao. On the rollout-training mismatch in modern RL systems. In *OPT 2025: Optimization for Machine Learning*, 2025. URL https://openreview.net/forum?id=xsonlP8DRV.

[62] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, YuYue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=2a36EMSSTp.

[63] Jixiao Zhang and Chunsheng Zuo. Grpo-lead: A difficulty-aware reinforcement learning approach for concise mathematical reasoning in language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5642–5665, 2025.

[64] Le Zhang, Bo Wang, Xipeng Qiu, Siva Reddy, and Aishwarya Agrawal. Rearank: Reasoning re-ranking agent via reinforcement learning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 2458–2471, 2025.

[65] Pengle Zhang, Jia Wei, Jintao Zhang, Jun Zhu, and Jianfei Chen. Accurate int8 training through dynamic block-level fallback. *arXiv preprint arXiv:2503.08040*, 2025.

[66] Juntao Zhao, Wenhao Lu, Sheng Wang, Lingpeng Kong, and Chuan Wu. Qspec: Speculative decoding with complementary quantization schemes. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 4779–4795, 2025.

[67] Pei Zhou, Jay Pujara, Xiang Ren, Xinyun Chen, Heng-Tze Cheng, Quoc V Le, Ed H. Chi, Denny Zhou, Swaroop Mishra, and Steven Zheng. SELF-DISCOVER: Large language models self-compose reasoning structures. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL `https://openreview.net/forum?id=BROvXhmzYK`.

A Training Details

GRPO Training Configurations Table 6 summarizes the hyperparameters of GRPO training for Qwen2.5-3B and Qwen2.5-Math-7B experiments.

Table 5: Hyperparameters for GRPO Training

Hyperparameter	**Qwen2.5-3B**	**Qwen2.5-Math-7B**
Policy learning rate	1×10^{-6}	1×10^{-6}
Training batch size	32	128
Group Size (G)	16	16
Max Prompt Length	512	2048
Max Response Length	1024	4096
Optimizer	AdamW	AdamW
Clip range ϵ_{low}, ϵ_{high}	0.2, 0.2	0.2, 0.28

In all experiments, we apply a token-level truncated importance sampling threshold [61] of 5.0.

Compute Resources All experiments were conducted on an internal cluster. The Qwen2.5-3B experiments consumed around 100 GB of peak memory and took approximately 10 hours to run. The Qwen2.5-Math-7B experiments consumed around 370 GB of peak memory and took approximately 20 hours to run. For OCC experiments, the runtime was notably longer due to its global top-percentile computation over the entire tensor.

B Memory Overhead of Rollout-ResQ-$S_{2:4}$

Per linear layer, let $X \in \mathbb{R}^{M\times K}, W \in \mathbb{R}^{K\times N}$ with $M = 1$ per sequence. Rollout-ResQ-$S_{2:4}$ computes the quantized activation $Q(X)$, the sparse quantized residual $S(Q(\Delta X))$, the quantized weight $Q(W)$ with the sparse branch (density $d = 0.5$) stored as a bitmask plus values.

We write B_m^c for the per-layer memory traffic (in bytes) of component $c \in \{W, X\}$ (weight, activation) under method $m \in \{BF16, ours\}$ ($ours$ refers to Rollout-ResQ-$S_{2:4}$), and $B_m = B_m^W + B_m^X$ for the total.

Weight. $B_{BF16}^W = 2KN$, $B_{ours}^W = \frac{1}{2}KN$, giving a $4\times$ reduction ($\frac{1}{2}KN/2KN = \frac{1}{4}$).

Activation. $B_{BF16}^X = 2MK$, $B_{ours}^X = \underbrace{\frac{4}{8}MK}_{Q(X)} + \underbrace{\frac{MK}{8}}_{\text{mask}} + \underbrace{\frac{4}{8}dMK}_{S(Q(\Delta X))} = MK(\frac{5}{8} + \frac{1}{4}) = \frac{7}{8}MK$,

giving a $2.29\times$ reduction ($\frac{\frac{7}{8}MK}{2MK} = \frac{7}{16}$).

Total ratio and decode regime. Summing weight and activation traffic, the total memory relative to BF16 is

$$\frac{B_{ours}}{B_{BF16}} = \frac{\frac{7}{8}MK + \frac{1}{2}KN}{2MK + 2KN} = \frac{\frac{7}{8}M + \frac{1}{2}N}{2M + 2N}.$$

In the decode phase when $M = 1$, this gives

$$\frac{B_{ours}}{B_{BF16}} = \frac{\frac{7}{8} + \frac{N}{2}}{2(N+1)} = \frac{7}{16(N+1)} + \frac{1}{4} \cdot \frac{1}{1 + 1/N}.$$

As N grows, the first term vanishes and the second approaches $\frac{1}{4}$, so $\frac{B_{ours}}{B_{BF16}} \to \frac{1}{4}$: a $4\times$ reduction, matching the FP4 baseline.

Forming $\Delta X = X - Q(X)$ needs the high-precision X only transiently: it is already present as the incoming activation. A single pass consumes it and emits $Q(X)$ and $S(Q(\Delta X))$, after which X is freed. No separate high-precision copy is stored, so the persistent activation footprint remains the $\frac{7}{8}MK$ above.

C Ablation Studies on Sparsity Patterns and Sparsity Levels

We conducted an ablation study of different sparsity levels ($25\%, 50\%, 75\%$) for the three sparsity patterns Rollout-ResQ-$S^{32\times64}_{50\%}$, Rollout-ResQ-$S_{50\%}$, Rollout-ResQ-$S_{2:4}$.

Table 6: GSM8K-test scores on different sparsity levels.

Density level	Rollout-ResQ-$S^{32\times64}_{50\%}$	Rollout-ResQ-$S_{50\%}$	Rollout-ResQ-$S_{2:4}$
25%	84.53	84.31	84.31
50%	84.61	84.61	85.90
75%	84.99	84.84	85.90

Table 6 shows that Rollout-ResQ is relatively robust to the choice of sparsity level, where the values are GSM8K-test scores after training. While increasing density from 25% to 75% leads to modest accuracy improvements, the gains beyond 50% are small. The 50% setting provides a favorable trade-off, achieving a significant inference speedup while matching the highest accuracy observed at 75% (for the S2:4 configuration). Therefore, we use 50% as the default setting in our experiments. Lower density levels can lead to reduced accuracy or unpredictable results. Based on our empirical observations, 50% density was the most stable setting, providing significant reductions in memory usage and improved inference speed while preserving accuracy comparable to that of higher-density or dense matrices.

D Comparison with High Precision Gradient via LoRA (QeRL)

Tables 7 and 8 show that, on the 3B model, QeRL improves over the HiF4 baseline on both benchmarks and achieves the best MATH500 score overall. This pattern does not hold at 7B, however: QeRL not only trails Rollout-ResQ-$S_{2:4}$ across all four benchmarks but also falls below the HiF4 baseline on three of them—AIME24, AIME25, and AMC23—improving on HiF4 only on MATH500.

Table 7: Accuracy (%) on Qwen2.5-3B.

Method	GSM8K-test	MATH500
HiF4	82.03	44.60
QeRL	83.92	**52.80**
Rollout-ResQ-$S_{2:4}$	**85.90**	52.60

E KL Divergence between Inference and Training Policies

Figure 6 shows the policy KL divergence curves of the six experiments discussed in Section 4.2. We use the following metric to approximate the KL divergence between the training and inference

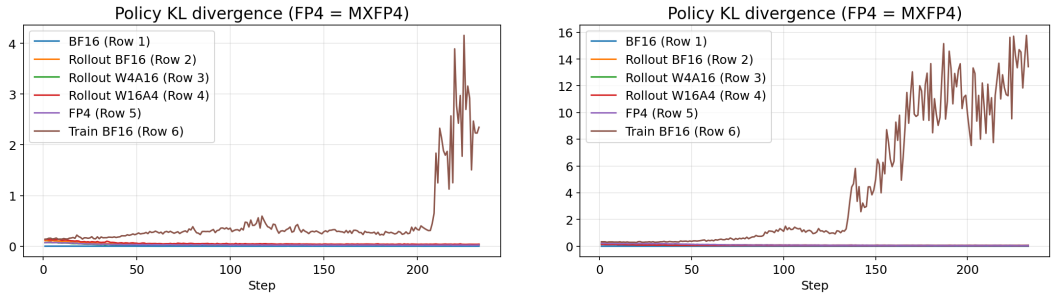
Table 8: Accuracy (%) on Qwen2.5-7B.

Method	AIME24	AIME25	AMC23	MATH500
HiF4	16.15	7.29	55.16	59.40
QeRL	12.29	4.69	47.19	67.65
Rollout-ResQ-$S_{2:4}$	**20.73**	**8.54**	**55.78**	**68.95**

policies,

$$D_{\text{KL}}(\pi_{\text{inference}} \parallel \pi_{\text{train}}) \approx \frac{1}{\sum_t m_t} \sum_t m_t \big(\log \pi_{\text{inference}}(a_t) - \log \pi_{\text{train}}(a_t) \big) \tag{6}$$

where $\pi_{\text{inference}}$ is the inference policy, π_{train} is the training policy, and m_t is the response mask of token a_t. Throughout the learning, training in BF16 exhibits significantly higher KL divergence between the polices than the other five settings, revealing that the performance collapse is associated with policy mismatch. Moreover, for HiF4, the mismatch keeps amplifying as the training continues for BF16-train, reaching a peak KL divergence of 4.157 towards the end. In comparison, the HiF4 baseline just peaks at 0.0812 and BF16 baseline peaks at 0.000721. All other runs in Table 1 never exceeds 0.13. Similarly, for MXFP4, BF16-train + FP4-rollout reaches 15.76 but all other runs never exceed 0.5.



(a) HiF4 (b) MXFP4

Figure 6: Policy KL divergence across differen precision configurations for training and rollout for HiF4 and MXFP4.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our main contributions are explicitly listed in the Introduction (Section 1, lines 100 - 116). These claims are supported by the experimental results in Section 5.2 and 5.3.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: We discussed the computational efficiency of our method in Section 5.4. Section 6 (Limitations and Future Directions) explains that our experiments are conducted in simulated FP4 due to the absence of FP4 support in hardware accelerators. The prolonged runtime limits us from studying larger models, datasets, and extensive exploration on sparsity levels.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper is primarily empirical, though it provides the mathematical formulation for the proposed method.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We are not able to provide the code at the time of submission due to company policy. However, we provide comprehensive details of the methodology, specify the training configurations, framework, models, datasets, as well as the link to a publicly available simulated FP4 code (Section 6), offering sufficient information to implement and reproduce the main experimental results that would align with our conclusions.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: Due to company policy, we cannot disclose the code at the time of submitting the paper. To support reproducibility, we provide the details of the method, specify the training configurations, framework, models, datasets and the link to a publicly available simulated FP4 code, enabling sufficient information to implement and reproduce the paper results.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not

including code, unless this is central to the contribution (e.g., for a new open-source benchmark).

- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Detailed experimental settings, including the framework (VeRL), models (Qwen2.5), and datasets (GSM8K, DAPO-Math), are described in Section 5.1. Training hyperparameter configurations are provided in Appendix A, Table 4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Due to high computational cost of performing full reinforcement learning post-training under simulated quantization, we report single-run results for the experiments. But the results across different models, datasets and quantization formats support the robustness of our claims.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.

- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: In Appendix A, we indicate that the experiments were conducted on an internal cluster, and reported the memory required and the time of execution.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our research focuses on improving the efficiency of existing LLM training techniques and conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This work introduces an efficient post-training methodology rather than releasing new models, datasets, or applications, and therefore poses no direct societal risks or benefits beyond the broader LLM research ecosystem.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This work introduces a training methodology rather than releasing a new high-risk model or sensitive dataset.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All used assets, including the models, the VeRL framework, datasets, and quantization code, are properly credited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: This paper does not release any new dataset, code or model.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This research does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.