PALORA: PARTIAL LOW-RANK ADAPTATION FOR LANGUAGE MODEL POST-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Low-rank adaptation (LoRA) reduces trainable parameters in language model post-training but applies the same update parameterization across target matrices. We introduce Partial Low-Rank Adaptation (PALORA), which directly trains selected Transformer projection matrices and retains LoRA elsewhere. Its probe–select–reinitialize procedure begins with short LoRA-only runs of Group Relative Policy Optimization (GRPO). Complementary binary masks yield absolute adapter-gate gradients of the GRPO loss on fixed held-out rollout groups. For each matrix, we normalize mean sensitivity by its standard deviation across probe questions plus a numerical stabilizer, then average scores across probe seeds. The lowest-scoring matrices receive direct updates: an empirical allocation rule rather than a diagnosis of insufficient adapter rank. After discarding probe weights, final GRPO training restarts from the original checkpoint with fresh adapters and a fixed allocation. We evaluate instruction-tuned Qwen2.5 models at 1.5B, 3B, and 7B parameters on five mathematical reasoning benchmarks. The 1.5B configuration directly trains 14 of 196 candidate matrices, retaining rank-16 LoRA at the remaining 182. At validation-selected checkpoints, PALORA improves mean accuracy across the five benchmarks over the reported LoRA baselines by 2.15, 3.18, and 4.26 percentage points, respectively. These single-seed comparisons use nonidentical parameter budgets and do not isolate placement effects.

1 INTRODUCTION

Reinforcement learning (RL) complements supervised fine-tuning (SFT) by allowing language models to learn from feedback on their own responses rather than demonstrations alone (Ouyang et al., 2022; Shao et al., 2024). Mathematical reasoning is a natural setting for this approach: final answers can often be verified automatically. Yet post-training remains computationally demanding, requiring repeated response generation, policy updates, and reference-model evaluations when a KL penalty is used. Training fewer parameters can reduce gradient and optimizer-state storage, but does not eliminate these other costs.

Parameter-efficient fine-tuning (PEFT) limits adaptation costs by restricting the trainable parameters or the form of their updates (Prottasha et al., 2025). Low-Rank Adaptation (LoRA) freezes pretrained matrices and learns low-rank additive updates that can be merged into the backbone for inference (Hu et al., 2022). Its effectiveness in reasoning-oriented RL, including Tina's results on a 1.5B model (Wang et al., 2025), suggests that full-model fine-tuning is not always necessary. The question is therefore not simply how many parameters to train, but where to allocate them and which update parameterization to use.

Existing methods address different aspects of this allocation problem. AdaLoRA redistributes low-rank capacity according to importance, while LISA selectively trains layers (Zhang et al., 2023; Pan et al., 2024). We instead consider each Transformer projection matrix: should it retain a LoRA update, or should its weights become directly trainable? Throughout this paper, a module denotes one projection matrix, not an entire Transformer block. Training selected matrices directly removes the rank constraint on their updates while retaining LoRA elsewhere. The selection criterion, however, requires care. An adapter's influence on the current policy need not predict the benefit of replacing it with unrestricted updates. Moreover, equal numbers of selected matrices need not imply equal trainable-parameter costs.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

We introduce Partial Low-Rank Adaptation (PALORA), which makes this matrix-level choice through a probe–select–reinitialize procedure. Short LoRA-only probes are trained with Group Relative Policy Optimization (GRPO; Shao et al., 2024). On fixed held-out rollout groups, complementary binary masks enable or disable adapters, and absolute gradients of the GRPO loss with respect to adapter gates measure sensitivity. Within each probe seed, we divide each module's mean sensitivity by its standard deviation across probe questions plus a numerical stabilizer; averaging these ratios across seeds gives q_j (Equation 5). The lowest-scoring matrices are selected for direct training. This tests whether unrestricted updates help where adapter sensitivity is small relative to its variability, rather than treating the score as a diagnosis of insufficient rank. All probe weights are then discarded, and final GRPO training restarts from the original backbone checkpoint with fresh adapters and a fixed allocation.

We evaluate PALORA on Qwen2.5 instruction-tuned models at 1.5B, 3B, and 7B parameters using AIME-24, AMC23, MATH-500, Minerva, and an OlympiadBench subset. The documented 1.5B configuration directly trains 14 of 196 candidate matrices and retains rank-16 LoRA at the remaining 182. Across the reported model sizes, PALORA improves mean accuracy over the listed LoRA baselines, although the comparisons are not exactly parameter-matched. Our evaluation examines both allocation and its interpretation: we distinguish selected-matrix counts from trainable-parameter costs and use internal validation for checkpoint selection, keeping validation-selected and final checkpoints distinct. These choices help separate the effects of placement, adaptation budget, and training duration.

Our contributions are:

1. **Matrix-level mixed adaptation.** We formulate PALORA as direct training of selected projection matrices combined with LoRA elsewhere, explicitly distinguishing selected-module counts from trainable-parameter costs.
2. **Attribution-guided placement.** We develop a complementary-mask GRPO probe with within-seed variability normalization, cross-seed score averaging, and reinitialized post-training, making low-score placement an explicit, testable allocation rule.
3. **Evaluation of mixed adaptation.** We compare PALORA with LoRA, full-model fine-tuning, and alternative allocations on mathematical reasoning benchmarks, examining scoring choices while accounting for parameter costs and checkpoint selection.

2 Related Work

Reinforcement learning for language model post-training. RL complements SFT by learning from model-generated responses, using learned rewards in reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022) or verifiable rewards for mathematical reasoning (Shao et al., 2024). PPO typically relies on a learned value function (Schulman et al., 2017), whereas GRPO derives advantages from rewards within response groups, avoiding a separately trained critic (Shao et al., 2024). Later work refines policy optimization through decoupled clipping and dynamic sampling in DAPO (Yu et al., 2025), and sequence-level importance ratios in GSPO (Zheng et al., 2025). These methods primarily concern the optimization objective and training procedure, rather than the placement of trainable parameters. Tina demonstrates LoRA-based reasoning RL on a small language model (Wang et al., 2025), providing a starting point for studying parameter allocation during post-training.

Parameter-efficient adaptation and capacity allocation. PEFT restricts trainable parameters through additive, selective, or reparameterized adaptation (Prottasha et al., 2025). LoRA learns low-rank updates while freezing pretrained matrices (Hu et al., 2022). Several extensions replace uniform rank assignment with adaptive allocation: AdaLoRA uses importance-based allocation (Zhang et al., 2023), while SoRA, AutoLoRA, and ALoRA study sparse rank selection, meta-learned allocation, and rank redistribution, respectively (Ding et al., 2023; Zhang et al., 2024; Liu et al., 2024). These methods vary capacity within a low-rank parameterization. LISA instead samples layers for full-parameter updates (Pan et al., 2024), changing which locations are trained directly. Our work examines a different allocation decision at the projection-matrix level: whether to retain a LoRA update or train the underlying matrix without a rank constraint. The remaining candidate matrices retain LoRA.

Positioning PaLoRA. Partial Low-Rank Adaptation (PALORA) makes this matrix-level choice through a probe–select–reinitialize procedure. LoRA-only GRPO probes measure adapter-gate sensitivity under complementary masks on fixed held-out rollout groups. Mean sensitivity is normalized by its variability within each seed, then averaged across seeds (Equation 5). The lowest-scoring matrices are selected for direct training; probe weights are discarded before final GRPO training restarts from the original backbone checkpoint with fresh adapters. The scoring approach has conceptual links to connection sensitivity, continuous architecture optimization, and path attribution (Lee et al., 2019; Liu et al., 2019; Sundararajan et al., 2017), but does not compute exact Shapley values or inherit attribution guarantees. Measured adapter influence need not predict the benefit of unrestricted updates. We therefore test low-score placement empirically, distinguishing fixed-matrix-count comparisons from parameter-constrained comparisons rather than interpreting sensitivity as a diagnosis of insufficient adapter rank.

3 METHOD

3.1 MIXED FULL-MATRIX AND LOW-RANK ADAPTATION

Partial Low-Rank Adaptation (PALORA) chooses an update parameterization for each candidate Transformer projection matrix. For a backbone with d blocks, let $\mathcal{J} = \{1, \ldots, 7d\}$ index matrices in block order, followed by query, key, value, output, gate, up, and down projections within each block. A module denotes one matrix, not a whole block; Qwen2.5-1.5B-Instruct has $|\mathcal{J}| = 196$ modules. Write its pretrained weights as $W_j^0 \in \mathbb{R}^{d_j^{\text{out}} \times d_j^{\text{in}}}$. A fixed indicator $\xi_j \in \{0, 1\}$ determines the effective weights:

$$W_j^{\text{eff}} = \begin{cases} W_j, & \xi_j = 1, \\ W_j^0 + \dfrac{\alpha}{r} B_j A_j, & \xi_j = 0, \end{cases} \qquad B_j \in \mathbb{R}^{d_j^{\text{out}} \times r}, \quad A_j \in \mathbb{R}^{r \times d_j^{\text{in}}}. \tag{1}$$

Here, r and α are the LoRA rank and scaling parameter. When $\xi_j = 1$, W_j is initialized to W_j^0 and trained directly, without an adapter. Otherwise, W_j^0 remains frozen and only B_j, A_j are trained. Direct training removes the rank constraint on the update; it does not imply that the learned update has maximal rank.

Assuming all parameters outside these matrices and adapters remain frozen, the trainable-parameter count is the all-LoRA cost plus the cost of replacing selected adapters:

$$\begin{aligned} \mathcal{C}_{\text{LoRA}} &= r \sum_{j \in \mathcal{J}} \big(d_j^{\text{out}} + d_j^{\text{in}}\big), \\ \Delta\mathcal{C}_j &= d_j^{\text{out}} d_j^{\text{in}} - r\big(d_j^{\text{out}} + d_j^{\text{in}}\big), \\ \mathcal{C}(\boldsymbol{\xi}) &= \mathcal{C}_{\text{LoRA}} + \sum_{j \in \mathcal{J}} \xi_j \Delta\mathcal{C}_j, \qquad \rho_{\text{full}}(\boldsymbol{\xi}) = \frac{\sum_{j \in \mathcal{J}} \xi_j}{|\mathcal{J}|}. \end{aligned} \tag{2}$$

The selected-module fraction ρ_{full} is not a parameter fraction: attention and feed-forward matrices differ in size. We therefore distinguish fixed-count placement, $\sum_j \xi_j = n_{\text{full}}$, from parameter-constrained placement, $\mathcal{C}(\boldsymbol{\xi}) \le b_{\max}$. The former fixes the number of directly trained matrices; the latter limits the trainable-parameter budget.

Rather than evaluate $2^{|\mathcal{J}|}$ allocations, PALORA follows a probe–select–reinitialize procedure. Short LoRA-only probes are trained with Group Relative Policy Optimization (GRPO; Shao et al., 2024), then scored on held-out questions using complementary adapter masks. Scores are normalized within each seed and averaged across seeds to choose a fixed allocation. This procedure measures adapter sensitivity, not the expected gain from replacing an adapter with an unrestricted update; the selection rule is therefore an empirical hypothesis. All probe weights are discarded before final GRPO training, which restarts from the original backbone checkpoint.

3.2 SENSITIVITY PROBE AND CROSS-SEED AVERAGING

Each probe uses the same reward and GRPO objective as final post-training, with stage-specific settings listed in Table 1. A scalar gate τ_j controls each adapter:

$$W_j^{\text{probe}}(\tau_j) = W_j^0 + \tau_j \frac{\alpha}{r} B_j A_j. \tag{3}$$

Probe fitting freezes the backbone, fixes all gates to one, and optimizes the adapters for 300 GRPO steps. A pool of 500 questions, disjoint from the training pool by question ID, is reserved for scoring. Each seed independently samples 4,096 training questions and 16 probe questions. Both the fitted adapters and the scoring questions may therefore vary across seeds.

After fitting, all model weights are frozen. For seed e and probe question h, the all-gates-on policy generates $G = 8$ completions. These completions, their rewards and advantages, and their old-policy and reference log-probabilities remain fixed throughout scoring. We sample independent mask entries $\omega_{j,h}^{(e)} \sim \text{Bernoulli}(1/2)$ and set $\boldsymbol{\tau}_h^{(e,+)} = \boldsymbol{\omega}_h^{(e)}$ and $\boldsymbol{\tau}_h^{(e,-)} = \mathbf{1} - \boldsymbol{\omega}_h^{(e)}$. Let $\mathcal{L}_h^{(e)}(\boldsymbol{\tau})$ be the GRPO loss on the cached group, with current-policy probabilities recomputed under gates $\boldsymbol{\tau}$. The paired sensitivity is

$$s_{j,h}^{(e)} = \sum_{\nu \in \{+,-\}} \tau_{j,h}^{(e,\nu)} \left| \frac{\partial \mathcal{L}_h^{(e)}(\boldsymbol{\tau})}{\partial \tau_j} \bigg|_{\boldsymbol{\tau} = \boldsymbol{\tau}_h^{(e,\nu)}} \right|. \tag{4}$$

Each module is active in exactly one mask, so the score retains its absolute gate gradient from that evaluation. Only the gates require gradients; neither model weights nor the answer verifier are differentiated with respect to their parameters. The gate derivative includes both the clipped policy-surrogate term and the configured KL penalty.

For each module and seed, we summarize the $H = 16$ paired observations by their mean relative to their population standard deviation, then average these ratios across seeds:

$$\begin{aligned} \widehat{m}_{j,e} &= \frac{1}{H} \sum_{h=1}^{H} s_{j,h}^{(e)}, \\ \widehat{\sigma}_{j,e} &= \left[\frac{1}{H} \sum_{h=1}^{H} \left(s_{j,h}^{(e)} - \widehat{m}_{j,e} \right)^2 \right]^{1/2}, \\ q_{j,e} &= \frac{\widehat{m}_{j,e}}{\widehat{\sigma}_{j,e} + \delta}, \qquad q_j = \frac{1}{|\mathcal{E}|} \sum_{e \in \mathcal{E}} q_{j,e}, \end{aligned} \tag{5}$$

where $\delta = 10^{-12}$ and $\mathcal{E} = \{42, 43, 44\}$. The denominator is a standard deviation, not a standard error. Thus, q_j measures sensitivity relative to its variability, rather than absolute sensitivity alone. A low score can reflect substantial relative variation even when gradients are not uniformly small. We average raw seed-specific scores, without pooling observations before normalization or converting scores to ranks; rank averaging is only a separate diagnostic.

3.3 PLACEMENT AND REINITIALIZED POST-TRAINING

For fixed-count placement, we sort modules by increasing q_j, breaking ties by numeric block index and then by the projection order defined above. The first n_{full} modules receive $\xi_j = 1$; the remainder receive $\xi_j = 0$. The documented 1.5B configuration selects 14 matrices and retains rank-16 LoRA at the other 182. This low-score rule tests whether unrestricted updates help where adapter sensitivity is small relative to its variability; it does not diagnose insufficient adapter rank. High-score, random, magnitude-based, and type-balanced placements provide controls. Parameter-constrained comparisons must additionally satisfy Equation 2 and are distinguished from fixed-count comparisons.

Final training reloads the complete original checkpoint. Selected matrices become trainable with no active adapters; all other candidate matrices receive fresh adapters with random A_j and zero B_j, giving zero initial updates. Probe gates are removed, parameters outside the allocation remain frozen, and no probe weights are reused. The allocation stays fixed throughout GRPO training,

Algorithm 1 PALORA: probe, select, and reinitialize

Require: Original checkpoint Θ^0; disjoint training and probe pools; candidate modules $\mathcal{J}$; seeds $\mathcal{E}$; rank r; scaling α; selected-matrix count n_{full}

for each seed $e \in \mathcal{E}$ **do**
  Reload Θ^0; attach fresh LoRA adapters to all candidates
  Freeze the backbone and fix all gates to one
  Sample 4,096 training questions and $H = 16$ probe questions
  Train adapters for 300 GRPO steps; freeze all model weights
  for each probe question h **do**
    Generate $G = 8$ completions with all gates set to one
    Cache rewards, advantages, and old-policy/reference log-probabilities
    Sample $\boldsymbol{\omega}_h^{(e)}$ and form its complement
    Evaluate both masks on the cached rollout group
    Compute $s_{j,h}^{(e)}$ for all j using Equation 4
  end for
  Compute $q_{j,e}$ using Equation 5
end for
Average raw scores across seeds to obtain q_j
Select the lowest-scoring n_{full} modules, breaking ties by module index
Set $\xi_j = 1$ at selected modules and $\xi_j = 0$ elsewhere
Discard probe weights and reload Θ^0
Initialize Equation 1 with $W_j = W_j^0$ or fresh adapters
Remove probe gates; train the fixed allocation with GRPO
Select the best checkpoint on internal validation; retain the final checkpoint

although directly trained matrices and adapter factors may use different learning rates. Algorithm 1 summarizes the procedure.

For completion a in a group of size G, let $c_a \in \{0, 1\}$ indicate verified correctness, $f_a \in \{0, 1\}$ indicate a parseable boxed answer, and ℓ_a denote the completion length. Given a length cap $\ell_{\text{cap}} > 0$, the reward R_a and group-centered advantage $\widehat{A}_a$ are

$$\begin{aligned} R_a &= f_a + c_a \left[2 + 0.05 \frac{(\ell_{\text{cap}} - \ell_a)_+}{\ell_{\text{cap}}} \right], \\ \widehat{A}_a &= R_a - \frac{1}{G} \sum_{b=1}^{G} R_b, \qquad (z)_+ = \max\{z, 0\}. \end{aligned} \tag{6}$$

Advantages are not divided by the group reward standard deviation; this is separate from the sensitivity normalization in Equation 5. Following Shao et al. (2024), GRPO uses token-level probability-ratio clipping, averaging token losses within each completion before averaging across completions. The KL coefficient is $\lambda_{\text{KL}} = 0.04$. Internal validation selects the best checkpoint; the final checkpoint is retained and reported separately.

4 EXPERIMENTS

4.1 DATASETS

For reinforcement learning with verifiable rewards (RLVR), we use the 40,315-example DeepScaleR preview dataset (Agentica, 2025). Preprocessing removes duplicates, invalid answers, and potential overlap with the five evaluation benchmarks, identified by exact matching, normalized Indel similarity of at least 75%, or shared 13-token n-grams. The remaining 36,582 examples are partitioned into 35,582 training, 500 internal-validation, and 500 probe questions. The probe pool informs allocation and is not an independent test set. The documented final GRPO run processes the first 2,000 questions in a fixed training order generated with selection seed 2026.

We evaluate on AIME-24 (30 questions), AMC23 (40), MATH-500 (500), Minerva (272), and an OlympiadBench subset (675) (Hendrycks et al., 2021; Lewkowycz et al., 2022; He et al., 2024).

For benchmark b, let n_b denote its size and $c_{b,h,a} \in \{0, 1\}$ indicate correctness of completion a for question h. With $G_{\text{eval}} = 8$ sampled completions per question, benchmark accuracy and the five-benchmark macro-average are

$$\begin{aligned} \text{Acc}_b &= \frac{1}{n_b G_{\text{eval}}} \sum_{h=1}^{n_b} \sum_{a=1}^{G_{\text{eval}}} c_{b,h,a}, \\ \overline{\text{Acc}} &= \frac{1}{5} \sum_{b=1}^{5} \text{Acc}_b \,. \end{aligned} \tag{7}$$

Tables report these quantities as percentages. The metric estimates pass@1, not pass@8 or voting accuracy; eight completions do not represent eight independent training runs.

Evaluation uses vLLM with temperature 0.8, top-p 0.95, repetition penalty 1.0, and an 8,192-token generation cap. Answers are extracted from the final parseable boxed expression and checked with a shared mathematical verifier. We also track token-cap hits and a repeated-ngram heuristic; neither establishes nontermination.

4.2 Compared methods

We compare PALoRA with the following baselines and alternative allocation strategies: (a) **LLM-CoT**: the frozen instruction-tuned backbone with chain-of-thought prompting Wei et al. (2022) and no parameter updates. (b) **Vanilla LoRA**: LoRA adaptation Hu et al. (2022) at all configured target linear projections, with the underlying backbone weights frozen. (c) **Full-FT**: full-model fine-tuning, rather than direct training of only selected projection matrices. (d) **Hybrid-Sensitivity**: mixed adaptation whose matrix–adapter allocation is determined by gradient-times-value sensitivity scores following Michel et al. (2019). (e) **Hybrid-Bilevel**: mixed adaptation whose allocation is obtained through bilevel optimization of relaxed selection variables Liu et al. (2019), rather than the probe score q_j. (f) **Hybrid-Perturbation**: mixed adaptation using perturbation-based scores Wang et al. (2021) to determine the allocation. (g) **Hybrid-Random**: mixed adaptation with randomly selected matrices assigned direct updates and LoRA retained elsewhere.

We report each configuration's trainable-parameter percentage. These methods need not use identical parameterizations or matched budgets: fixing the selected-matrix count does not fix $\mathcal{C}(\boldsymbol{\xi})$ (Equation 2).

4.3 Experimental setup

Hardware. All experiments run on NVIDIA H800 GPUs.

Backbones. We evaluate the instruction-tuned Qwen2.5 models at 1.5B, 3B, and 7B parameters. Table 1 documents the 1.5B PALoRA configuration; its settings should not be assumed to describe every comparison.

We use vLLM for rollout generation. Evaluation follows the decoding and verification protocol above; training uses the stage-specific sampling settings in Table 1.

Post-training pipeline. All RLVR methods start from instruction-tuned checkpoints without an additional supervised fine-tuning warm-up. For PALoRA, probe weights are discarded before final training, which restarts from the original checkpoint with a fixed allocation.

Prompt template. Training and evaluation use the same prompt template.

Implementation. Our implementation builds on Open-R1 (Hugging Face, 2025), vLLM, and TRL. Unless specified otherwise, training follows the Open-R1 and Tina (Wang et al., 2025) settings.

For the run in Table 1, three probe seeds determine one allocation, followed by a single final-training run at seed 42. Training stops at step 2,000 of a 5,000-step cosine schedule, with a minimum learning-rate ratio of 0.1 and 500 warmup steps. With one process, micro-batch size one, accumulation over eight micro-batches, and eight completions per question, each optimizer update uses one completion group. We use zero weight decay, gradient-norm clipping at 1.0, and gradient checkpointing. The colocated vLLM engine enters sleep mode between rollout generation and optimization.

Table 1: Documented Qwen2.5-1.5B-Instruct PALORA configuration. Probe fitting and final-policy training are separate stages. Settings are taken from saved experiment manifests.

Setting	RL probe	Final GRPO
Base	Qwen2.5-1.5B-Instruct	
Candidate projections	28 blocks $\times$ 7 matrices = 196	
LoRA rank / scaling (r / α)	16 / 32	16 / 32
Directly trained matrices	0	14; remaining 182 use LoRA
Training seeds	42, 43, 44	42
Optimizer steps	300 per seed	2,000; schedule horizon 5,000
Probe questions	16 per seed	Placement fixed before training
Micro-batch / accumulation	1 / 8	1 / 8
Completions per question	8	8
Prompt / completion cap	4,096 / 1,024	4,096 / 8,192
Sampling temperature / top-p	0.7 / 0.95	0.7 / 0.95
LoRA / direct-matrix learning rate	10^{-5} / not used	10^{-5} / 10^{-6}
Warmup fraction	0.1	0.1 (500 schedule steps)
KL coefficient (λ_{KL})	0.04	0.04
Dropout during GRPO	Disabled	Disabled
Reward standard-deviation scaling	None	None
Trainable parameter storage	BF16 in recorded probe	FP32; frozen weights BF16
Loss implementation	Explicit TRL GRPO	Liger fused GRPO, BF16 autocast
Internal validation	None during probing	Every 250 steps, 500 questions

For this run, internal validation evaluates all 500 questions every 250 optimizer steps, sampling one completion per question. The checkpoint with the highest validation answer accuracy is designated *best*; step 2,000 is designated *final*. Both are retained, and external benchmark scores never determine checkpoint selection. Table 2 reports validation-selected checkpoints; steps from earlier comparison runs refer to actual training steps, not evaluation-directory names.

4.4 MAIN EXPERIMENTAL RESULTS

Table 2 compares PALORA with the frozen chain-of-thought baseline, Vanilla LoRA, Full-FT, and alternative hybrid allocations. Using the macro-average in Equation 7, PALORA achieves 21.99%, 39.24%, and 49.14% accuracy at 1.5B, 3B, and 7B, respectively. These results exceed the reported Vanilla LoRA averages by 2.15, 3.18, and 4.26 percentage points. The improvement spans all five benchmarks at every model size, rather than being driven by one task. The reported trainable-parameter fraction is lower than Vanilla LoRA's at 1.5B and 7B and slightly higher at 3B. Thus, mixed adaptation improves on the listed LoRA configurations, although these are not exactly parameter-matched comparisons.

Relative to Full-FT, PALORA offers a different accuracy–parameter trade-off at each scale. At 1.5B, it reaches 21.99% mean accuracy versus 20.72% while training a reported 5.58% of parameters. It matches Full-FT on AIME-24 and improves on the other four benchmarks. At 3B and 7B, it trails Full-FT by 0.91 and 1.05 percentage points while training 6.45% and 8.87% of parameters, respectively. The remaining gap is task-dependent: PALORA outperforms Full-FT on AMC23 at both scales and comes within 0.27 points on MATH-500 at 7B, but falls behind by 3.40 points on Minerva at 7B. Direct training of selected matrices therefore retains much, but not all, of Full-FT's accuracy in these comparisons. The parameter fractions describe final-policy optimization; they do not measure end-to-end savings, which must include probing and rollout generation.

Among the 1.5B hybrid methods, PALORA has the highest mean accuracy, exceeding perturbation-based and random placement by 1.16 and 1.57 percentage points. At 3B, it exceeds the sensitivity-based and bilevel variants by 1.70 and 1.42 points. Its parameter fraction lies within the range of the other hybrid configurations at both scales, so the gains do not simply track the largest trainable parameter set. However, unequal parameter costs prevent attributing the differences solely to placement. The results support studying where unrestricted updates are allocated, not interpreting

Table 2: Validation-selected results on Qwen2.5 instruction-tuned backbones. Accuracy is reported as a percentage, with percentage signs omitted; Avg. is the unweighted mean across five benchmarks. Evaluation uses eight completions per question and an 8K new-token cap. Post-training uses seed 42. Param.% denotes the reported trainable-parameter percentage, and Best step is selected by internal validation accuracy.

Method	Param.%	Best step	AIME-24	AMC23	MATH-500	Minerva	Olympiad	Avg.
Qwen2.5-1.5B-Instruct								
LLM-CoT	0	–	1.25	19.69	47.68	14.15	17.46	20.05
Full-FT	100	750	1.67	20.00	48.45	14.48	19.02	20.72
Vanilla LoRA	5.76	1750	0.83	20.31	46.60	13.19	18.28	19.84
Hybrid-Sensitivity	5.83	1700	1.67	18.75	48.10	14.34	18.83	20.34
Hybrid-Bilevel	5.95	1750	0.83	21.56	47.45	14.06	18.56	20.49
Hybrid-Perturbation	5.47	1500	1.25	21.25	47.70	14.98	18.94	20.83
Hybrid-Random	5.52	1250	1.67	18.44	47.48	15.49	19.04	20.42
PALORA	5.58	1750	1.67	22.81	49.80	16.45	19.22	21.99
Qwen2.5-3B-Instruct								
LLM-CoT	0	–	2.92	27.50	55.10	19.21	23.30	25.60
Full-FT	100	1200	15.42	50.94	69.65	31.66	33.09	40.15
Vanilla LoRA	6.22	1500	10.83	47.81	63.43	27.21	31.00	36.06
Hybrid-Sensitivity	6.53	1050	12.92	49.38	65.70	28.17	31.56	37.54
Hybrid-Bilevel	6.07	1250	12.50	50.31	66.18	28.68	31.43	37.82
PALORA	6.45	1550	13.75	52.19	68.25	29.96	32.07	39.24
Qwen2.5-7B-Instruct								
LLM-CoT	0	–	7.92	41.88	64.60	26.38	31.59	34.47
Full-FT	100	1250	25.83	62.81	78.80	40.21	43.30	50.19
Vanilla LoRA	8.94	1750	20.83	58.13	72.65	34.15	38.67	44.88
PALORA	8.87	1800	24.58	63.44	78.53	36.81	42.35	49.14

low sensitivity as a diagnosis of insufficient adapter rank. All reported trained models use final-training seed 42; cross-seed probe averaging supplies no independent replication of final training. The following ablations examine the scoring and selection choices more directly.

4.5 Ablation studies and analysis

Table 3: Ablations of PALORA on Qwen2.5-3B-Instruct. Accuracy is reported as a percentage, with percentage signs omitted. Evaluation uses eight completions per question and an 8K new-token cap; final training uses seed 42.

Method	AMC23	MATH-500	Minerva
Full-FT	50.94	69.65	31.66
PALORA	52.19	68.25	29.96
- Select highest-scoring matrices	50.31	66.35	28.45
- Mean-only score q_j^{mean}	51.25	67.22	29.32
- Full-model probe	52.50	68.10	30.10
- Single-seed probe	50.94	67.58	29.46

Ablations of PALORA. Table 3 examines four variants on Qwen2.5-3B-Instruct. First, high-score placement directly trains the n_{full} matrices with the largest q_j, reversing the default selection order. Second, mean-only scoring removes the within-seed standard-deviation normalization while retaining cross-seed averaging. Third, a full-model probe replaces LoRA-only probe fitting with full-model fine-tuning, scores the candidate matrices, and selects the highest-scoring ones for direct training. This changes both the probe parameterization and selection direction. Finally, single-seed probing

uses only seed 42. The two alternative scores are

$$q_j^{\text{mean}} = \frac{1}{|\mathcal{E}|} \sum_{e \in \mathcal{E}} \widehat{m}_{j,e},$$
$$q_j^{\text{single}} = q_{j,42}, \qquad \mathcal{E} = \{42, 43, 44\}, \tag{8}$$

where $\widehat{m}_{j,e}$ and $q_{j,e}$ are defined in Equation 5.

High-score placement reduces accuracy by 1.88, 1.90, and 1.51 percentage points on AMC23, MATH-500, and Minerva, respectively, favoring the default low-score rule in this setting. Removing variability normalization lowers accuracy by 0.94, 1.03, and 0.64 points, indicating that mean sensitivity alone yields a less effective allocation here. Single-seed probing produces corresponding drops of 1.25, 0.67, and 0.50 points. The full-model probe gives mixed, numerically small differences: gains of 0.31 points on AMC23 and 0.14 on Minerva, with a 0.15-point decrease on MATH-500. Because that variant also reverses selection direction, these differences cannot be assigned to probe parameterization alone. Overall, the comparisons favor variability normalization and cross-seed averaging for the reported training seed. They do not establish robustness across final-training seeds. Nor does equal selected-matrix count ensure equal parameter cost, so the results do not fully isolate placement effects.

Visualization of adapter sensitivity scores. Figure 1 in Appendix A shows the scores used by PALORA and the matrices selected for direct training. Each location represents one Transformer projection matrix, not an entire block. The plotted q_j is the cross-seed average defined in Equation 5: within each seed, mean absolute gate sensitivity is divided by its population standard deviation across probe questions, with stabilizer δ. The visualization therefore shows sensitivity relative to variability, rather than gradient magnitude alone. A low score may reflect large relative variation even when individual gradients are substantial. Seed-specific scores are averaged directly, without rank conversion, so the displayed values match those used for selection.

The n_{full} lowest-scoring matrices receive $\xi_j = 1$ and are trained directly; the remaining candidates retain LoRA with $\xi_j = 0$. The figure thus documents how measured adapter sensitivity translates into a matrix-level allocation. It does not identify intrinsically unimportant matrices or demonstrate that the selected adapters lack sufficient rank. The scores describe the fitted LoRA policy on cached probe rollouts, whereas the benefit of unrestricted updates must be tested through training comparisons. Once selection is complete, the allocation remains fixed, all probe weights are discarded, and final GRPO training restarts from the original backbone checkpoint with fresh adapters at the unselected matrices.

5 CONCLUSION

We introduced Partial Low-Rank Adaptation (PALORA), which combines direct training of selected Transformer projection matrices with LoRA at the remaining candidate matrices for language model post-training. Its probe–select–reinitialize procedure uses short LoRA-only GRPO probes and complementary adapter masks to measure gate sensitivity on fixed held-out rollout groups. The resulting scores q_j summarize mean sensitivity relative to its variability across probe questions and are averaged across probe seeds; the lowest-scoring matrices are selected for unrestricted updates. Final GRPO training then restarts from the original backbone checkpoint with fresh adapters and a fixed allocation. Across the reported Qwen2.5 experiments, PALORA improves mean accuracy over Vanilla LoRA at all three model sizes, outperforming Full-FT at 1.5B while remaining below it at 3B and 7B. The ablations favor low-score placement, variability normalization, and cross-seed averaging in the evaluated setting. These findings support matrix-level mixed adaptation as an alternative to uniform low-rank updates, but do not establish that low sensitivity diagnoses insufficient adapter rank. Because the comparisons use nonidentical parameter budgets and a single final-training seed, parameter-matched evaluations across additional seeds remain necessary to isolate placement effects and assess their reliability. Measuring end-to-end cost, including probing, is also needed to determine when the accuracy gains justify the additional selection stage.

AI USE STATEMENT

We used generative AI tools to assist with LaTeX formatting and to edit, rephrase, and polish the English of text written by the authors, improving clarity and readability. The authors reviewed the AI-assisted revisions and remain fully responsible for the manuscript's scientific content, claims, citations, experiments, and final presentation.

ETHICS STATEMENT

PALORA studies the allocation of trainable parameters in language model post-training, with evaluation focused on mathematical reasoning. It is not a safety intervention. Adapted models may retain or amplify biases, generate harmful or misleading content, or lose safeguards present in the original checkpoint. Rewards based on answer correctness and output format do not assess these risks, nor does a verified final answer establish the validity of the preceding reasoning. The reported experiments do not include dedicated safety evaluations, so improvements on mathematical benchmarks should not be interpreted as evidence of safe deployment. Use beyond the evaluated setting requires application-specific testing and appropriate safeguards, particularly where incorrect outputs could cause harm.

The study relies on existing model checkpoints and mathematical datasets. Their use and any subsequent redistribution should respect the applicable licenses, access conditions, and attribution requirements. The preprocessing described in this paper filters potential overlap between the post-training data and evaluation benchmarks, but such filtering cannot rule out all semantic duplicates or contamination inherited from pretraining. The held-out probe pool informs matrix selection and therefore does not constitute an independent test set. Checkpoint selection uses internal validation rather than external benchmark results. These distinctions matter because overstating evaluation independence could give a misleading impression of generalization.

Reducing adaptation costs could make language model research more accessible, while also lowering barriers to harmful model customization. Moreover, fewer trainable parameters do not by themselves imply lower end-to-end energy use or emissions. PALORA requires a separate probing stage, and final post-training still incurs rollout generation and policy optimization costs. Any claim of computational or environmental benefit therefore requires measurement of the complete pipeline. The present results support neither a quantified emissions reduction nor a general claim of deployment readiness.

REFERENCES

Agentica. DeepScaleR preview dataset. Hugging Face dataset, 2025. URL https://huggingface.co/datasets/agentica-org/DeepScaleR-Preview-Dataset.

Ning Ding, Xingtai Lv, Qiaosen Wang, Yulin Chen, Bowen Zhou, Zhiyuan Liu, and Maosong Sun. Sparse low-rank adaptation of pre-trained language models. In *Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://api.semanticscholar.org/CorpusID:265294736.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In *Annual Meeting of the Association for Computational Linguistics*, 2024. URL https://arxiv.org/abs/2402.14008.

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. *arXiv preprint arXiv:2103.03874*, 2021. URL https://arxiv.org/abs/2103.03874.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2106.09685.

Hugging Face. Open r1: A fully open reproduction of deepseek-r1, January 2025. URL https://github.com/huggingface/open-r1.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip H. S. Torr. SNIP: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1810.02340.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. *arXiv preprint arXiv:2206.14858*, 2022. URL https://arxiv.org/abs/2206.14858.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. In *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1806.09055.

Zequan Liu, Jiawen Lyn, Wei Zhu, Xing Tian, and Yvette Graham. Alora: Allocating low-rank adaptation for fine-tuning large language models. *arXiv preprint arXiv:2403.16187*, 2024.

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32, 2019.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, 2022. URL https://arxiv.org/abs/2203.02155.

Rui Pan, Xiang Liu, Shizhe Diao, Renjie Pi, Jipeng Zhang, Chi Han, and Tong Zhang. LISA: Layerwise importance sampling for memory-efficient large language model fine-tuning. *arXiv preprint arXiv:2403.17919*, 2024. URL https://arxiv.org/abs/2403.17919.

Nusrat Jahan Prottasha, Upama Roy Chowdhury, Shetu Mohanto, Tasfia Nuzhat, Abdullah As Sami, Md Shamol Ali, Md Shohanur Islam Sobuj, Hafijur Raman, Md Kowsher, and Ozlem Ozmen Garibay. PEFT A2Z: Parameter-efficient fine-tuning survey for large language and vision models. *arXiv preprint arXiv:2504.14117*, 2025. URL https://arxiv.org/abs/2504.14117.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, 2017. URL https://arxiv.org/abs/1703.01365.

Ruochen Wang, Minhao Cheng, Xiangning Chen, Xiaocheng Tang, and Cho-Jui Hsieh. Rethinking architecture selection in differentiable nas. *arXiv preprint arXiv:2108.04392*, 2021.

Shangshang Wang, Julian Asilis, Ömer Faruk Akgül, Enes Burak Bilgin, Ollie Liu, and Willie Neiswanger. Tina: Tiny reasoning models via LoRA. *arXiv preprint arXiv:2504.15777*, 2025. URL https://arxiv.org/abs/2504.15777.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Huai hsin Chi, F. Xia, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *ArXiv*, abs/2201.11903, 2022. URL https://api.semanticscholar.org/CorpusID:246411621.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

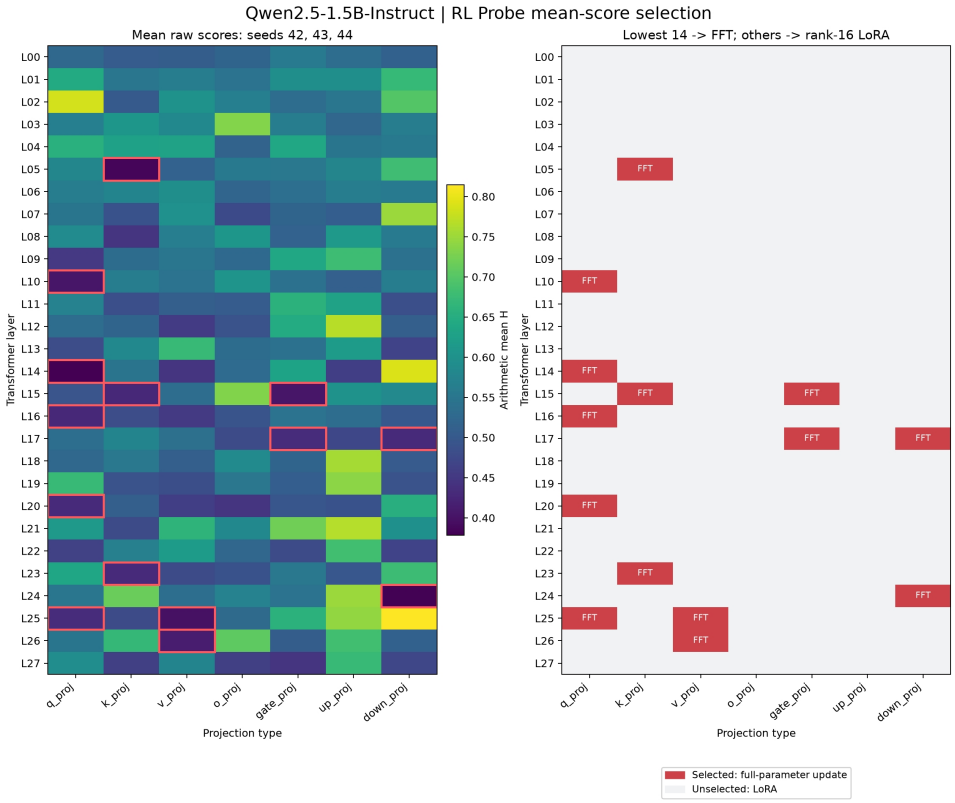


Figure 1: Matrix scores and selected matrices .

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. AdaLoRA: Adaptive budget allocation for parameter-efficient fine-tuning. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2303.10512.

Ruiyi Zhang, Rushi Qiang, Sai Ashish Somayajula, and Pengtao Xie. Autolora: Automatically tuning matrix ranks in low-rank adaptation based on meta learning. *arXiv preprint arXiv:2403.09113*, 2024.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

A Appendix for Experiments

Visualization of LoRA importance scores. Figure 1 presents the module-level scores used by PALORA together with the matrices selected for direct training.