NORMTUNE: NORM-GUIDED PARAMETER-EFFICIENT FINE-TUNING FOR REASONING-INTENSIVE TASKS

Anonymous authors
Paper under double-blind review

ABSTRACT

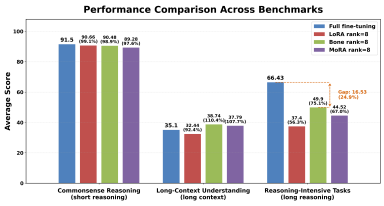
Full fine-tuning has achieved remarkable success on reasoning-intensive tasks, but it often requires substantial GPU resource consumption. Parameter-efficient fine-tuning (PEFT) provides a promising alternative for reducing this cost, yet we observe a substantial performance gap on reasoning-intensive tasks. In this paper, we present three key observations showing that this gap is closely associated with norm-related under-adaptation, where PEFT produces markedly smaller weight-update norms than full fine-tuning. Based on these observations, we propose Norm-Guided Parameter-Efficient Fine-Tuning (NormTune), a simple yet effective strategy that adaptively strengthens insufficiently updated layers according to their weight-update norms. Extensive experiments show that NormTune consistently enhances multiple PEFT methods across challenging reasoning benchmarks. Specifically, when trained on S1K-1.1, integrating NormTune into rank-8 LoRA, Bone, and MoRA yields substantial average accuracy improvements of 24.24%, 11.33%, and 16.21% , respectively, across AIME 2024, MATH 500, GPQA, and AIME 2025. Moreover, NormTune-enhanced rank-256 LoRA, Bone, and MoRA achieve performance comparable to S1 full fine-tuning, with respective average gains of 2.17%, 0.93%, and 3.00% over this baseline.

1 INTRODUCTION

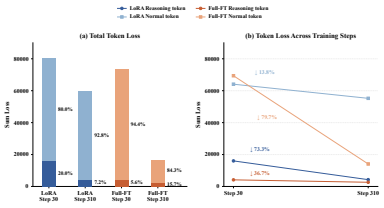
Recent studies (Wu et al., 2025; OpenAI, 2024; Guo et al., 2025a; Yu et al., 2025a; Xie et al., 2024) find that, for challenging reasoning tasks, carefully increasing inference-time compute can significantly boost performance. However, most of these approaches depend on substantial compute budgets, specialized training pipelines, or complex inference-time search, making them expensive, hard to reproduce, and impractical for broad adoption. To address these limitations, many studies(Muennighoff et al., 2025; Ye et al., 2025; Liu et al., 2025; Yu et al., 2025b) have explored simpler and more reproducible alternatives based on supervised fine-tuning (SFT). These methods, however, still depend on full fine-tuning, which becomes progressively less practical as model scale increases, particularly in compute-constrained settings. For example, full fine-tuning of Qwen2.5-32B-Instruct (Team, 2024) on a dataset with a 32K context length requires 562.07 GB of memory even with gradient checkpointing, making it difficult to apply under limited compute resources. To address this challenge, we consider parameter-efficient fine-tuning (PEFT) methods (Hu et al., 2021; Lin et al., 2024a; Feng et al., 2025), which adapt large pretrained models to downstream tasks by updating only a small subset of parameters while keeping the backbone model frozen.

Motivated by this efficiency, we apply PEFT to reasoning-intensive adaptation, while noting that its reduced training cost may come with non-negligible performance risks in such challenging settings. Thus we investigate the question: **Can low-rank PEFT retain the Full-FT comparable adaptation capability?** As shown on the right side of Figure 1a, existing PEFT methods exhibit a substantial performance drop on reasoning-intensive tasks, failing to reach 90 % of the full fine-tuning performance commonly used criterion for effective low-dimensional adaptation (Aghajanyan et al., 2021). In contrast, PEFT remains effective on tasks involving shorter reasoning chains and long-context understanding, as shown in the left and middle panels of Figure 1a. To examine this gap, we present three key observations. **Observation 1:** PEFT methods exhibit a much larger loss contribution on reasoning tokens such as transition, self-verification, template, and reasoning-boundary tokens compared with Full-FT at earlier training. During training, PEFT methods exhibit a smaller relative reduction in loss on normal tokens (i.e., all tokens other than reasoning tokens)

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.



(a) Performance comparison across evaluation regimes. PEFT methods exhibit a substantial gap from full fine-tuning only on reasoning-intensive tasks using Qwen2.5-32B-Instruct.



(b) Comparison of token-level and total loss values between rank-256 LoRA and full fine-tuning on 16 identical sequences from reasoning-intensive tasks using Qwen2.5-32B-Instruct.

Figure 1: Performance across evaluation regimes and token-level loss under PEFT and full fine-tuning. Detailed results are reported in Appendix Tables 16, 17, and 11.

than on reasoning tokens, whereas Full-FT exhibits the opposite pattern, as shown in Figure 1b. **Observation 2:** The gradients induced by reasoning token and normal token losses exhibit substantial conflicts across most trainable parameters in earlier layers. However, the total token gradients remain aligned with the normal token optimization direction, while its effective magnitude is partially reduced by the conflicting reasoning token gradients. This gradient interference weakens the effective parameter updates in earlier layers. **Observation 3:** PEFT methods induce substantially smaller weight-update norms than full fine-tuning, especially in earlier layers. To achieve performance comparable to Full-FT, we expect PEFT methods to exhibit similar layer-wise weight-update magnitudes as Full-FT. Based on this observation, we propose Norm-Guided Parameter-Efficient Fine-Tuning (NormTune), which reduces the gradients conflicts in earlier layers, reallocates optimization strength toward under-adapted layers and improves model performance.

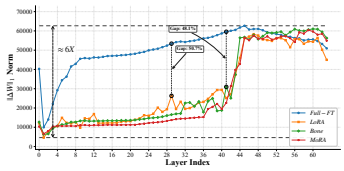
Our contributions can be summarized as follows:

- We systematically investigate the limitations of PEFT on reasoning-intensive tasks and present three key observation, revealing that PEFT induces substantially smaller fine-tuning-induced weight changes, which is closely associated with its performance degradation.
- Motivated by these observations, we propose a novel PEFT method named NormTune, which reduces the gradients conflicts in earlier layers and reallocates optimization strength toward under-adapted layers. To the best of our knowledge, NormTune is the first approach that enables PEFT methods to achieve comparable performance relative to full fine-tuning on reasoning-intensive tasks.
- Extensive experiments demonstrate the effectiveness and generality of NormTune. Specifically, when trained on S1K-1.1, NormTune improves rank-8 LoRA, Bone, and MoRA by 24.24%, 11.33%, and 16.21% in average accuracy across AIME 2024, MATH 500, GPQA, and AIME 2025, respectively. Moreover, NormTune-enhanced rank-256 LoRA, Bone, and MoRA achieve performance comparable to S1 full fine-tuning, with respective average gains of 2.17%, 0.93%, and 3.00% over this baseline.

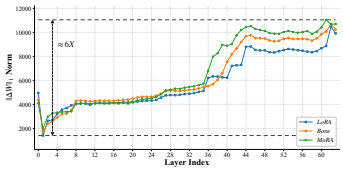
2 Methodology

2.1 Key Observations On Reasoning-Intensive Tasks

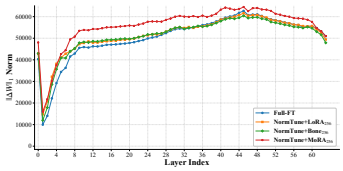
Observation 1: PEFT methods exhibit a much larger loss contribution on reasoning tokens compared with Full-FT at earlier training. To investigate the accuracy gap between PEFT and Full-FT on reasoning-intensive tasks, we analyze how training loss is distributed across tokens and how it evolves during fine-tuning, motivated by prior work Lin et al. (2024b); Pang et al. (2025). Specifically, we sample 16 sequences from S1K-1.1 (Muennighoff et al., 2025) and evaluate token-level cross-entropy losses on the same sequences at training step 30, immediately after warm-up, and step 310, at the end of training, using Qwen2.5-32B-Instruct. We inspect the high-loss tail using



(a) Visualization of entry-wise weight-update norms of full fine-tuning and PEFT methods

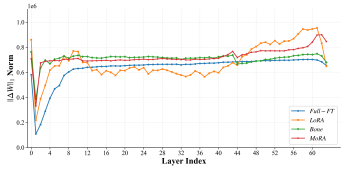


(b) Visualization of weight-update norm of different PEFT methods after 30 steps training

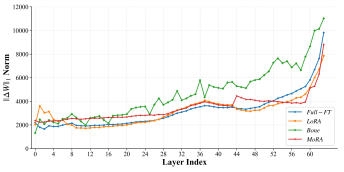


(c) Visualization of entry-wise weight-update norms of full fine-tuning and NormTune-enhanced PEFT methods

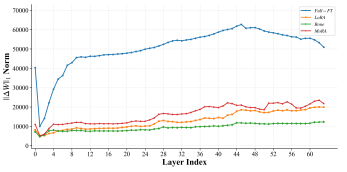
Figure 2: Layer-wise norm comparison among Full-FT, rank-256 PEFT methods, early-stage PEFT statistics, and NormTune+PEFT methods.



(a) Commonsense reasoning task



(b) GovReport task



(c) Reasoning-intensive task

Figure 3: Layer-wise norm comparison between full fine-tuning (Full-FT) and rank-8 PEFT methods on (a) commonsense reasoning, (b) GovReport, and (c) reasoning-intensive tasks.

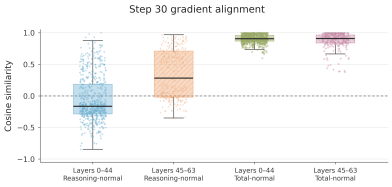
the empirical 99th percentile (P_{99})Sato et al. (2022) of token-level losses as a cutoff. Most tokens above this threshold correspond to transitions, self-verification expressions, templates, or reasoning boundaries such as "Alternatively" and "Wait".. For this analysis, we refer to tokens serving these four roles as reasoning tokens, and to all remaining tokens as normal tokens. As shown in Figure 1b, PEFT methods exhibit a much larger loss contribution on reasoning tokens compared with Full-FT at earlier training. After training at step 310 , PEFT methods exhibit a smaller relative reduction in loss on normal tokens than on reasoning tokens, whereas Full-FT exhibits the opposite pattern. This indicates that PEFT disproportionately allocates its optimization capacity toward adapting reasoning token patterns, while providing insufficient adaptation for normal token representations compared with Full-FT.

Observation 2: The gradients induced by reasoning token and normal token losses exhibit substantial conflicts across most trainable parameters in earlier layers.

To understand how different token groups affect the optimization dynamics of PEFT methods, we decompose the training loss into reasoning token and normal token components. Let θ denote all trainable LoRA parameters. The total loss can be rewritten as

$$\mathcal{L}_{\text{total}}(\theta) = \frac{1}{|\mathcal{I}|}\left(\sum_{(i,t)\in\mathcal{I}_{\text{reasoning}}} \ell_{i,t}(\theta) + \sum_{(i,t)\in\mathcal{I}_{\text{normal}}} \ell_{i,t}(\theta)\right) = \mathcal{L}_{\text{reasoning}}(\theta) + \mathcal{L}_{\text{normal}}(\theta), \quad (1)$$

where $\mathcal{I}$ denote total tokens, i is sequence and the t is prediction position. Because both loss terms are differentiable with respect to the LoRA parameters θ, taking the gradient of Eq. 1 yields $\nabla_\theta \mathcal{L}_{\text{total}} = \nabla_\theta \left(\mathcal{L}_{\text{special}} + \mathcal{L}_{\text{normal}}\right) = \nabla_\theta \mathcal{L}_{\text{special}} + \nabla_\theta \mathcal{L}_{\text{normal}}$. Therefore, the update direction of LoRA parameters is determined by the combination of two independent optimization signals: the gradient induced by reasoning tokens and the gradient induced by normal tokens. We define their gradient alignment using cosine similarity. Specifically, for $\nabla_\theta \mathcal{L}_{\text{special}}$ and $\nabla_\theta \mathcal{L}_{\text{normal}}$, the cosine similarity is defined as $\text{CosSim}(\nabla_\theta \mathcal{L}_{\text{special}}, \nabla_\theta \mathcal{L}_{\text{normal}}) = \frac{\nabla_\theta \mathcal{L}_{\text{special}}^\top \nabla_\theta \mathcal{L}_{\text{normal}}}{\|\nabla_\theta \mathcal{L}_{\text{special}}\|_2 \|\nabla_\theta \mathcal{L}_{\text{normal}}\|_2}$. We also measure their alignment with the overall optimization direction $\text{CosSim}(\nabla_\theta \mathcal{L}_{\text{special}}, \nabla_\theta \mathcal{L}_{\text{total}})$ and $\text{CosSim}(\nabla_\theta \mathcal{L}_{\text{normal}}, \nabla_\theta \mathcal{L}_{\text{total}})$. As shown in Figure 4, the gradients induced by special-token and normal-token losses exhibit substantial conflicts across most trainable parameters in earlier layers. However, this behavior is reversed in deeper layers, where the two optimization signals become more compatible. Although earlier layers exhibit conflicts between the two token-group gradients,



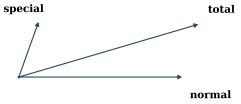
(a) Gradient alignment across early and late layers, measured by $\mathrm{CosSim}(\nabla_\theta \mathcal{L}_{\mathrm{reasoning}}, \nabla_\theta \mathcal{L}_{\mathrm{normal}})$ and $\mathrm{CosSim}(\nabla_\theta \mathcal{L}_{\mathrm{total}}, \nabla_\theta \mathcal{L}_{\mathrm{normal}})$. The corresponding Full-FT results are provided in Figure 7.



(b) Layer 4 MLP down (A)



(c) Layer 41 attention Q (B)



(d) Layer 53 attention V (A)



(e) Layer 57 MLP gate (B)

Figure 4: Gradient alignment analysis between token groups and layer-wise optimization directions. The left panel shows the cosine similarity between different gradients. The right panels visualize representative gradient directions and the L1 norms of the gradients for different LoRA parameters (LoRA A and B) across early and late layers after 30 training steps.

the total gradient remains positively aligned with the normal-token gradient. This indicates that the special-token gradient does not reverse the normal-token optimization direction, but reduces its effective update magnitude.

Observation 3: PEFT methods induce substantially smaller weight-update norms than full fine-tuning, especially in earlier layers.

Motivated by norm-based generalization theory Bartlett et al. (2017); Galanti et al. (2023); Nagarajan & Kolter (2019), which links network norm to model capacity and generalization, we analyze the magnitude of fine-tuning-induced weight changes as a proxy for layer-wise adaptation strength. Specifically, for module m in layer l, we define the parameter update under fine-tuning strategy τ as $\Delta W^{\tau}_{l,m} = W^{\tau}_{l,m} - W^{0}_{l,m}$, where $W^{0}_{l,m}$ denotes the corresponding base-model weight and $W^{\tau}_{l,m}$ denotes the fine-tuned weight. Here, $\tau \in \{\mathrm{Full}, \mathrm{PEFT}\}$, where Full represents full fine-tuning and $\mathrm{PEFT} \in \{\mathrm{LoRA}, \mathrm{Bone}, \mathrm{MoRA}\}$. To quantify the adaptation strength of each layer, we aggregate the entry-wise L_1 norm of the effective weight updates over all target modules in that layer:

$$U^{\tau}_{l} = \sum_{m \in \mathcal{M}_l} \left\| \Delta W^{\tau}_{l,m} \right\|_{1,1}, where \left\| \Delta W^{\tau}_{l,m} \right\|_{1,1} = \sum_{i,j} \left| \Delta W^{\tau}_{l,m,ij} \right|. \tag{2}$$

Here, $\mathcal{M}_l$ denotes the set of target modules in layer l, including the attention projections $\{Q, K, V, O\}$ and the MLP projections $\{\mathrm{Gate}, \mathrm{Up}, \mathrm{Down}\}$. In addition, the LoRA scaling factor $s = \alpha/r = 2$, which is already incorporated in this function. Based on this metric, Figure 2a shows that PEFT methods with rank 256 induce substantially smaller weight-update norms than full fine-tuning, with the gap being particularly pronounced in earlier layers. This indicates that PEFT provides much weaker effective adaptation in these layers. Furthermore, we visualize the layer-wise weight-update norms of rank-8 PEFT methods on other tasks, as shown in Figure 3. Unlike reasoning-intensive tasks, these benchmarks do not exhibit significant under-adaptation in earlier layers, suggesting that the insufficient layer adaptation observed in PEFT is particularly associated with reasoning-intensive fine-tuning.

2.2 NormTune

Overall, the three observations suggest that improving PEFT on reasoning-intensive tasks requires stronger adaptation for normal tokens, particularly in earlier layers. Observation 1 reveals an imbalance in token-level loss reduction, suggesting insufficient adaptation for normal tokens relative to Full-FT. Observation 2 provides a explanation: although reasoning-token and normal-token gradients conflict in earlier layers, the total gradient remains positively aligned with the normal-token gradient. This interference therefore attenuates the gradient component along the normal-token direction while preserving its positive alignment. This pattern is consistent with Observation 3, which

shows substantially smaller weight-update norms in these layers under PEFT. These findings motivate increasing the learning rates of under-adapted layers to compensate for attenuated updates. Specially, for fixed parameters, gradients, and optimizer state, increasing a layer's learning rate scales the magnitude of its parameter update without changing its direction.

Motivated by these observations, we propose Norm-Guided Parameter-Efficient Fine-Tuning (NormTune), which assigns larger learning rates to weakly updated layers based on their weight-update norms, thereby improving the adaptation of under-trained layers. In general, U_l^{Full}, i.e., the layer-wise weight-update norm obtained from full fine-tuning, can serve as an ideal reference scale for calibrating PEFT learning, encouraging U_l^{PEFT} to align with the adaptation magnitude of full fine-tuning. However, obtaining U_l^{Full} requires performing full fine-tuning, which is impractical due to its substantial training time and computational cost. Here,we first examine whether the layer-wise norm statistics from PEFT can serve as a practical alternative. Empirically, we observe that after a complete PEFT run, the maximum-to-minimum ratio of layer-wise update norms, $\max_l U_l^{\mathrm{PEFT}} / \min_l U_l^{\mathrm{PEFT}}$, is close to that of full fine-tuning, with both ratios being approximately $6\times$. This indicates that although PEFT produces much smaller absolute update norms, it still captures a similar relative range of layer-wise adaptation magnitudes.

Nevertheless, obtaining U_l^{PEFT} from a completed PEFT run remains costly. We therefore introduce a lightweight probing procedure that performs only 30 PEFT training steps. As shown in Figure 2b, the early-stage update norms $U_l^{\mathrm{PEFT},s}$ already exhibit a comparable maximum-to-minimum ratio of approximately $6\times$. Based on this observation, we adopt this early-stage statistic as an efficient proxy for estimating the desired layer-wise adaptation range. Directly assigning a distinct learning rate to each layer may be sensitive to local fluctuations in $U_l^{\mathrm{PEFT},s}$. We therefore group layers into n buckets according to their update magnitudes inspired by Zhang et al. (2025). Let $U_{\min}^{\mathrm{PEFT},s} = \min_l U_l^{\mathrm{PEFT},s}$, $U_{\max}^{\mathrm{PEFT},s} = \max_l U_l^{\mathrm{PEFT},s}$. The range $[U_{\min}^{\mathrm{PEFT},s}, U_{\max}^{\mathrm{PEFT},s}]$ is uniformly divided into n buckets. For the k-th bucket, we denote by $\bar{U}_k^{\mathrm{PEFT},s}$ the mean update magnitude of the layers inside this bucket. Following the standard min-max normalization used in data preprocessing, which rescales numerical values into a fixed range while preserving their relative order Mining (2006), we normalize $\bar{U}_k^{\mathrm{PEFT},s}$ as

$$q_k = \frac{\bar{U}_k^{\mathrm{PEFT},s} - \bar{U}_{min}^{\mathrm{PEFT},s}}{\bar{U}_{max}^{\mathrm{PEFT},s} - \bar{U}_{min}^{\mathrm{PEFT},s}}, \qquad q_k \in [0,1]. \tag{3}$$

Here, q_k measures the relative adaptation sufficiency of bucket k. We define the bucket-wise learning rate by mapping q_k to the interval $[\eta_{\min}, \eta_{\max}]$:

$$\eta_k = \eta_{\min} + (\eta_{\max} - \eta_{\min}) h(q_k), \tag{4}$$

where $\eta_{\min}$ denotes the base learning rate, $\eta_{\max} > \eta_{\min}$. The function h should satisfy three simple requirements. First, the weakest adapted bucket should receive the largest learning rate, and the strongest adapted bucket should receive the smallest learning rate:$h(0) = 1, h(1) = 0$. Second, the learning rate should decrease as the adaptation magnitude increases, so $h(q)$ should be monotonic non-increasing on $[0, 1]$. Third, buckets near $q = 0$ should remain close to the maximum learning rate, because they all correspond to insufficiently adapted layers. We encode this weak-adaptation smoothing condition as $h'(0) = 0$. To keep the rule simple, we choose the lowest-degree polynomial satisfying these constraints. Let $h(q) = a + bq + cq^2$. Given $h(0) = 1$, $h'(0) = 0$, and $h(1) = 0$, the solution of $h(q)$ is:

$$h(q) = 1 - q^2. \tag{5}$$

Substituting Eq. 5 into Eq. 4, we obtain the final bucket-wise learning rate:

$$\eta_k = \eta_{\min} + (\eta_{\max} - \eta_{\min}) \left(1 - q_k^2\right). \tag{6}$$

All layers in the same bucket share the same learning rate:

$$\eta_l = \eta_k, \qquad \text{if layer } l \text{ belongs to bucket } k. \tag{7}$$

As shown in Figure 2c, our method consistently increases the update norms of insufficiently adapted layers, aligning them more closely with full fine-tuning and improving model performance.

3 EXPERIMENTAL RESULTS

Datasets. For training, we use the S1K-1.1 dataset Muennighoff et al. (2025), the Z1-Code-Reasoning107K dataset Yu et al. (2025b), the Data-repetition dataset Kopiczko et al. (2026), the GovReport dataset for long-context understandingHuang et al. (2021), and the Commonsense Reasoning dataset for short reasoning. We further evaluate different models on several challenging tasks. The reasoning-intensive tasks consist of AIME 2024 Art of Problem Solving (2024), MATH 500 Hendrycks et al. (2021), GPQA Rein et al. (2024), and AIME 2025 Art of Problem Solving (2025). The code-scientific reasoning tasks consist of MATH 500, GPQA, LiveCodeBench Jain et al. (2024), and BigCodeBench-Hard Zhuo et al. (2024).

Compared Methods. The baseline methods are broadly divided into two categories: reinforcement learning (RL) and supervised fine-tuning (SFT). In the RL category, we compare against OpenAI o1 OpenAI (2024), o3 OpenAI (2025), DeepSeek-R1 Guo et al. (2025b), and QwQ-32B Qwen Team (2025). In the SFT category, we compare against S1(Full-FT) Muennighoff et al. (2025), Z1-7B(Full-FT) Yu et al. (2025b), Qwen3-4B(Full-FT) Kopiczko et al. (2026) and OLMo-3-7B(Full-FT) Kopiczko et al. (2026). In addition, we compare against several representative PEFT methods, including DoRA Liu et al. (2024), AdaLoRA Zhang et al. (2023), LoRA+ Hayou et al. (2024), rsLoRA Kalajdzievski (2023), LoRA Hu et al. (2021), BoneKang (2024) and MoRAJiang et al. (2024). We further incorporate TempBalance Zhou et al. (2023) into the PEFT baselines for comparison.

Evaluation Metrics. For the S1K-1.1 and Z1-Code-Reasoning107K settings, we evaluate each locally evaluated checkpoint using greedy decoding with temperature $T = 0$, generating one completion per problem in each evaluation run. We repeat evaluation on the same checkpoint and the same benchmark questions for $R = 4$ runs on MATH 500 and GPQA, and $R = 16$ runs on AIME 2024, AIME 2025, LiveCodeBench, and BigCodeBench-Hard, where applicable. Tables 1 and 2 report the mean accuracy across these evaluation runs. These repetitions are evaluation runs of a fixed checkpoint, rather than independent training runs or stochastic decoding samples. The Avg. column is the unweighted mean of the four benchmark accuracies. For Qwen3-4B and OLMo-3-7B, we retain the sampling-based Avg@k and Pass@k metrics under the decoding configurations specified in Table 22. Results taken from prior publications retain their original evaluation protocols and are explicitly distinguished from locally evaluated results. Details of the uncertainty estimation are provided in Table 22. For the environment and settings also provided in Appendix A.3.

3.1 MAIN COMPARISON RESULTS

In this section, we present the main comparison results by applying NormTune to representative PEFT methods, including LoRA, Bone, and MoRA, and evaluating their effectiveness for test-time scaling. As shown in Table 1, rank-8 PEFT methods underperform on reasoning-intensive tasks. Incorporating NormTune substantially improves their performance, yielding average gains of 24.24%, 11.33%, and 16.21% for LoRA, Bone, and MoRA, respectively. Rank-8 NormTune-enhanced PEFT surpasses 90% of full fine-tuning performance and consistently outperform AdaLoRA, LoRA+, rsLoRA, DoRA and TempBalance+PEFT under the same rank-8 setting. When increasing the rank to 256, NormTune still brings consistent improvements to PEFT methods, with average gains of 13.44%, 10.46%, and 9.77% for LoRA, Bone, and MoRA, respectively. Notably, all three NormTune-enhanced variants surpass S1 (Full-FT) by 2.17%, 0.93%, and 3.00%, respectively. Beyond supervised fine-tuning, we further compare NormTune with state-of-the-art reinforcement-learning-based reasoning models. Using the same Qwen2.5-32B-Instruct backbone, all NormTune-enhanced variants outperform T1 (Qwen2.5-32B), and NormTune+MoRA further surpasses SSFT-32B-GFPO by 0.58%. These results show that NormTune substantially improves model performance with lower resource consumption.

For Z1-7B based one Qwen-2.5-Coder-7B-Instruct, we further evaluate code-related reasoning performance, as shown in Table 2. Under supervised fine-tuning with a similar model size, NormTune+LoRA, NormTune+Bone, and NormTune+MoRA all achieve performance comparable to the state-of-the-art full fine-tuning baseline Z1-7B (Full-FT). Notably, NormTune+LoRA further surpasses Z1-7B by over 0.6%. For RL-based training, NormTune+LoRA also achieves a performance

Table 1: Accuracy (%) of different test-time scaling methods on reasoning-intensive tasks with Qwen2.5-32B-Instruct. * denotes results reported in Muennighoff et al. (2025). The results of T1 (Qwen2.5-32B) and SSFT-32B-GFPO are sourced from Hou et al. (2025) and Jia et al. (2025).

Dataset	Method	Rank	AIME 2024 Avg@16	MATH 500 Avg@4	GPQA Avg@4	AIME 2025 Avg@16	Avg.
S1K-1.1	o3-mini-low*	-	56.3	95.8	70.6	42.1	66.2
	QwQ-32B*	-	46.7	90.6	54.5	32.7	56.13
	T1 (Qwen2.5-32B)	-	50.6	92.4	56.1	-	-
	SSFT-32B-GFPO	-	64.22	89.90	62.48	58.80	68.85
	r1*	-	79.8	97.3	71.5	70.0	79.65
	S1 (Full-FT)*	-	56.7	95.4	63.6	50	66.43
	r1-distill-Llama-70B*	-	57.1	94.5	65.2	56.3	68.28
	r1-distill-Qwen-14B*	-	61.7	93.9	59.1	48.0	65.68
	r1-distill-Qwen-32B*	-	58.3	94.3	62.1	49.6	66.08
	AdaLoRA	8	6.67	67.4	27.27	6.67	27.00
	DoRA	8	26.67	78.8	47.98	6.67	40.03
	RSLoRA	8	16.67	74.6	50.51	0	35.45
	LoRA+	8	33.33	88.8	51.52	33.33	51.75
	LoRA	8	10	78.8	47.47	13.33	37.4
	TempBalance + LoRA	8	20	80.4	51.01	13.33	41.19
	NormTune+LoRA(Ours)	8	53.33	92.6	60.61	40	61.64
	Bone	8	26.67	80.4	52.53	40	49.9
	TempBalance + Bone	8	36.67	89.4	55.05	40	55.28
	NormTune+ Bone (Ours)	8	50	92.8	62.12	40	61.23
	MoRA	8	26.67	80.4	51.01	20	44.52
	TempBalance + MoRA	8	23.33	89.2	51.01	26.67	47.55
	NormTune+MoRA (Ours)	8	40	91.6	64.65	46.67	60.73
	AdaLoRA	256	13.33	67.4	37.37	6.67	31.19
	DoRA	256	23.33	88.6	53.03	33.33	49.57
	RSLoRA	256	43.3	90.2	60.1	33.3	56.73
	LoRA+	256	50	90.6	58.59	33.33	58.13
	LoRA	256	36.67	92.6	58.08	33.33	55.16
	TempBalance + LoRA	256	40	90.2	54.4	40	56.15
	NormTune+LoRA (Ours)	256	63.33	93.6	64.14	53.33	**68.6**
	Bone	256	43.33	93.4	57.58	33.3	56.90
	TempBalance + Bone	256	50	90.8	60.61	40	60.35
	NormTune+Bone (Ours)	256	56.67	94.8	64.65	53.33	**67.36**
	MoRA	256	50	93.2	62.12	33.3	59.66
	TempBalance + MoRA	256	56.67	93.2	62.63	33.33	61.46
	NormTune+MoRA (Ours)	256	60	94.6	63.13	60	**69.43**

Table 2: Accuracy (%) of different test-time scaling methods on code-scientific reasoning tasks with Z1. * denotes results reported in Yu et al. (2025b).

Dataset	Method	Rank	MATH 500 AVG@4	GPQA AVG@4	LiveCodeBench AVG@16	BigCodeBench-Hard AVG@16	Avg.
Z1-Code-Reasoning107K	o1-preview*	-	85.5	73.3	43.2	23.0	56.3
	o1-mini*	-	90.0	60.0	53.7	27.7	57.9
	Deepseek-R1*	-	97.3	71.5	77.9	29.7	67.6
	QwQ-32B-Preview*	-	90.6	60.0	59.9	25.0	58.9
	Deepseek-V3*	-	90.2	59.1	56.3	27.7	58.3
	GPT-4o-0513*	-	75.8	46.5	43.4	25.0	47.7
	Z1-7B (Full-FT)*	-	76.4	47.5	35.3	22.3	<u>45.4</u>
	R1-Distill-Qwen-7B*	-	83.3	49.1	40.5	3.4	44.1
	s1.1-7B*	-	79.2	31.8	15.2	4.7	31.7
	OpenThinker-7B*	-	83.0	42.4	25.3	17.6	42.1
	AdaLoRA	256	72	51.5	29.4	12.8	41.4
	DoRA	256	73.2	49.5	30.2	18.2	42.8
	RSLoRA	256	72.4	46.5	29.7	13.5	40.5
	LoRA+	256	72	49.5	30.9	17.6	42.5
	LoRA	256	73.4	50	30.3	10.8	41.1
	TempBalance + LoRA	256	72.4	53.5	32.3	16.2	43.6
	NormTune+LoRA(Ours)	256	75.2	50	36.3	22.3	**46.0**
	Bone	256	75.4	46.5	32.3	17.6	43.0
	TempBalance + Bone	256	75.2	49.5	33.5	16.9	43.8
	NormTune+Bone (Ours)	256	76.8	47.5	34.2	22.7	45.3
	MoRA	256	71.6	44.4	30.9	17.6	41.1
	TempBalance + MoRA	256	76.6	46	32.6	18.2	43.4
	NormTune+MoRA(Ours)	256	76.6	50	35.4	19.6	**45.4**

level comparable to GPT-4o-0513, with an average score of 46.0 versus 47.7. This demonstrates the effectiveness of NormTune across various testing datasets.

For OLMo-3-7B, as shown in Table 3, all NormTune-enhanced methods, including NormTune+LoRA, NormTune+Bone, and NormTune+MoRA, outperform Full-FT. These results show that NormTune remains effective even for smaller models. Qwen3-4B experiments are presented in the Appendix in Table 7.

Table 3: Performance of Full-FT, PEFT, and NormTune-enhanced PEFT methods on OLMo-3-7B across GPQA, AIME'24, and AIME'25. * denotes results reported in Kopiczko et al. (2026).

Model	Method	GPQA		AIME'24		AIME'25		Average
		Avg@4	Pass@4	Avg@16	Pass@16	Avg@16	Pass@16	
OLMo-3-7B	Full-FT*	34.0	62.1	42.3	80.0	39.2	63.3	53.5
	AdaLoRA	42.6	66.2	22.1	43.3	22.7	50	41.2
	DoRA	30.6	57.5	33.3	73.3	30.8	56.7	47.0
	RSLoRA	42.6	70.2	22.7	56.6	21.7	53.3	44.5
	LoRA+	40.9	65.2	21.3	53.3	16.5	46.7	40.7
	LoRA(256)	31.9	62.1	26.7	63.3	24	43.3	41.9
	TempBalance + LoRA	40.9	65.2	21.3	53.3	20.2	56.7	42.9
	NormTune+LoRA(256)	38.8	69.2	42.3	80	35.4	60	**54.3**
	Bone(256)	35.9	63.6	33.3	73.3	31	56.7	49.0
	TempBalance + Bone	30.6	57.5	35.4	66.7	38.5	67.2	49.3
	NormTune+Bone(256)	38.1	60	45.2	76.7	37.1	64.6	**53.6**
	MoRA(256)	37.6	65.7	37.5	76.7	29.4	50	49.5
	TempBalance + MoRA	32.7	60	42.3	76.7	35.8	63.3	51.8
	NormTune+MoRA(256)	39.9	68.7	41	76.7	35.8	63.3	**54.2**

Table 4: Accuracy (%) of PEFT methods under various adjustment styles and ranks on S1.

METHOD	ADJUST. STYLE	RANK	Avg.
Full-FT	-	-	66.43
Full-FT	Stage	-	46.43
LoRA	-	256	55.16
LoRA	Linear	256	60.76
LoRA	Stage	256	63.2
LoRA	Exponential	256	62.79
LoRA	NormTune	256	**68.6**
Bone	-	256	56.90
Bone	Linear	256	59.79
Bone	Stage	256	63.28
Bone	Exponential	256	60.17
Bone	NormTune	256	**67.36**
MoRA	-	256	59.66
MoRA	Linear	256	61.79
MoRA	Stage	256	60.78
MoRA	Exponential	256	63.03
MoRA	NormTune	256	**69.43**

Table 5: Accuracy (%) of full fine-tuning and rank-256 PEFT methods under two learning-rate settings on reasoning-intensive tasks. See Table 9 for detailed results.

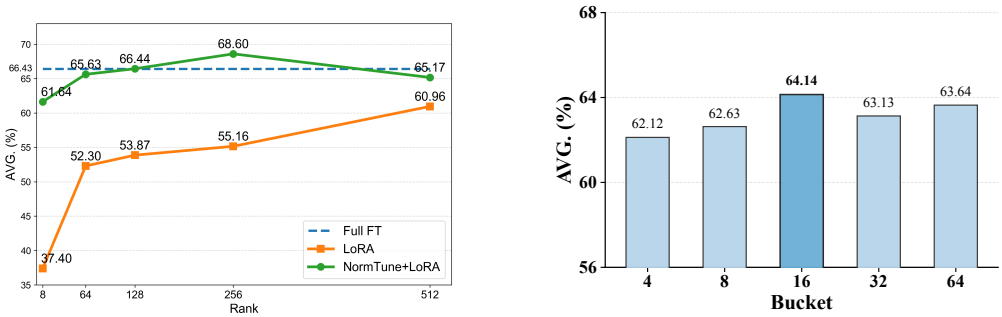
Method	Learning rate	Rank	Avg.
S1 (Full-FT)	1e-5	-	66.43
S1 (Full-FT)	1e-4	-	4.9
LoRA	1e-5	256	55.16
LoRA	1e-4	256	49.7
Bone	1e-5	256	56.90
Bone	1e-4	256	55.65
MoRA	1e-5	256	59.66
MoRA	1e-4	256	49.47

Across scales from 4B to 32B and across mathematical reasoning, code reasoning, and scientific question-answering benchmarks, NormTune consistently improves different PEFT methods. This demonstrates that NormTune is a general and scalable solution for improving test-time scaling with strong performance.

3.2 Ablation Study

Learning Rate. We fine-tune Qwen2.5-32B-Instruct on the S1K-1.1 reasoning dataset with different rank-256 PEFT methods under two learning-rate settings. As shown in Table 5, increasing the learning rate from 1×10^{-5} to 1×10^{-4} consistently hurts performance: the average accuracy decreases from 55.16% to 49.70% for LoRA, from 56.90% to 55.65% for Bone, and from 59.66% to 49.47% for MoRA. Full fine-tuning also collapses under the larger learning rate, with the average score dropping from 66.43% to 4.90%. These results suggest that performance gains cannot be obtained by uniformly increasing the learning rate, motivating NormTune to adaptively allocate larger update scales only to under-adapted layers.

Rank Sensitivity. As shown in Figure 5a, increasing the rank generally improves both LoRA and NormTune+LoRA, while NormTune+LoRA consistently outperforms LoRA across all ranks. The gains are particularly pronounced in low-rank regimes: at rank 8, NormTune+LoRA improves the average score from 37.40% to 61.64%. NormTune+LoRA achieves its best average performance at rank 256, reaching 68.6% and slightly exceeding the Full-FT baseline of 66.43%. Further increasing the rank to 512 reduces the score to 65.17%, indicating that larger ranks do not necessarily yield



(a) Average performance across reasoning-intensive tasks under different rank settings.

(b) Effect of the NormTune bucket number n on GPQA.

Figure 5: Results after training on S1K-1.1.

better reasoning-intensive performance. These results show that NormTune effectively improves LoRA across different rank budgets, especially when the adaptation capacity is highly constrained.

Adjustment Style. Based on the norm distribution of PEFT methods on reasoning-intensive tasks, and inspired by Howard & Ruder (2018), we further propose three naive layer-wise learning-rate adjustment strategies. The first one uses a linear decay across layers: $\eta_i = \eta_{\max} + (\eta_{\min} - \eta_{\max}) \cdot \frac{i}{N-1}$, where i is the layer index and N is the total number of layers. The second strategy, termed stage decay, assigns $\eta_{\max}$ to layers below layer 45 and $\eta_{\min}$ to layer 45 and above, since the former exhibit under-adaptation in layer-wise norms, as shown in Figure 2a. The last one, we further evaluate exponential-decay strategy, $e^{\frac{\max_i(norm_i)}{norm_i}-1}$. As shown in Table 4, both linear, stage and exponential-decay strategies improve the performance of all PEFT methods over their original versions. However, their gains are consistently smaller than those achieved by NormTune. For example, NormTune improves the average scores to 68.6%, 67.36%, and 69.43% for LoRA, Bone, and MoRA, respectively, outperforming both linear, stage and exponential-decay strategies. See Table 10 for detailed results. These results indicate that simply imposing a predefined learning-rate decay is beneficial, but the adaptive design of NormTune is more effective for reasoning-intensive adaptation. Moreover, full fine-tuning generally provides sufficient adaptation across layers, making it less suitable for our stage decay strategy.

3.3 Hyperparameter Sensitivity Analysis

We analyze the sensitivity of NormTune to the bucket number n, which determines the granularity of layer-wise learning-rate allocation. As shown in Figure 5b, NormTune is robust to a wide range of n. Across $n \in \{4, 8, 16, 32, 64\}$, the GPQA accuracy only changes from 62.12% to 64.14%, showing limited sensitivity to this hyperparameter. The best result is obtained at $n = 16$, where NormTune achieves 64.14%. This indicates that a moderate bucket number provides sufficient layer-wise differentiation without introducing unnecessary fragmentation.

4 Conclusion

In this work, we studied parameter-efficient fine-tuning for reasoning-intensive tasks with test-time scaling. Through three key observations, we propose Norm-Guided Parameter-Efficient Fine-Tuning (NormTune), which adaptively enhances under-trained layers and thereby substantially improves performance on reasoning-intensive tasks. Extensive experiments on three benchmarks demonstrate that NormTune consistently improves different PEFT methods and can achieve performance comparable to full fine-tuning with substantially lower resource consumption.

AI USE STATEMENT

Generative AI was used solely to assist with the organization and formatting of early manuscript drafts. It was not used for method conception, experimental implementation, or the substantive writing of the paper.

ETHICS STATEMENT

This work investigates Norm-Guided Parameter-Efficient Fine-Tuning for reasoning-intensive tasks. We do not involve human subjects or introduce new datasets. The primary foreseeable impact is improved performance of PEFT methods on reasoning-intensive tasks.

REPRODUCIBILITY STATEMENT

An anonymized supplementary package will include the exact coordinate, calibration, reconstruction, and terminal-evaluation configurations; fixed data splits and random seeds; per-cell manifests; and fresh-process evaluation scripts.

REFERENCES

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: long papers)*, pp. 7319–7328, 2021.

Chenxin An, Zhihui Xie, Xiaonan Li, Ming Zhong, Shansan Gong, Lei Li, Jun Zhang, Jingjing Xu, and Lingpeng Kong. Long chain-of-thought fine-tuning via understanding-to-reasoning transition. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 34506–34522, 2025.

Art of Problem Solving. 2024 american invitational mathematics examination (aime) problems and solutions. `https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions`, 2024. Includes AIME I and AIME II for 2024.

Art of Problem Solving. 2025 american invitational mathematics examination (aime) problems and solutions. `https://artofproblemsolving.com/wiki/index.php/2025_AIME_I`, 2025. Includes 2025 AIME I and linked AIME II materials.

Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. *Advances in neural information processing systems*, 30, 2017.

Yonatan Bisk, Rowan Zellers, Ronan Le bras, Jianfeng Gao, and Yejin Choi. Piqa: Reasoning about physical commonsense in natural language. *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 7432–7439, Jun 2020. doi: 10.1609/aaai.v34i05.6239. URL `http://dx.doi.org/10.1609/aaai.v34i05.6239`.

Sondos Mahmoud Bsharat and Zhiqiang Shen. Prompting test-time scaling is a strong llm reasoning data augmentation. *arXiv preprint arXiv:2510.09599*, 2025.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North*, Jan 2019. doi: 10.18653/v1/n19-1300. URL `http://dx.doi.org/10.18653/v1/n19-1300`.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Jinyuan Feng, Zhiqiang Pu, Tianyi Hu, Dongmin Li, Xiaolin Ai, and Huimu Wang. Omoe: Diversifying mixture of low-rank adaptation by orthogonal finetuning. *arXiv preprint arXiv:2501.10062*, 2025.

Tomer Galanti, Mengjia Xu, Liane Galanti, and Tomaso Poggio. Norm-based generalization bounds for compositionally sparse neural networks. *arXiv preprint arXiv:2301.12033*, 2023.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025a.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025b.

Soufiane Hayou, Nikhil Ghosh, and Bin Yu. Lora+: Efficient low rank adaptation of large models. *arXiv preprint arXiv:2402.12354*, 2024.

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.

Zhenyu Hou, Xin Lv, Rui Lu, Jiajie Zhang, Yujiang Li, Zijun Yao, Juanzi Li, Jie Tang, and Yuxiao Dong. T1: Advancing language model reasoning through reinforcement learning and inference scaling. *arXiv preprint arXiv:2501.11651*, 2025.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pp. 2790–2799. PMLR, 2019.

Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 328–339, 2018.

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. *arXiv preprint arXiv:2106.09685*, 2021.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. Efficient attentions for long document summarization, 2021.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

Sheng Jia, Xiao Wang, and Shiva Prasad Kasiviswanathan. Training large language models to reason in parallel with global forking tokens. *arXiv preprint arXiv:2510.05132*, 2025.

Ting Jiang, Shaohan Huang, Shengyue Luo, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, et al. Mora: High-rank updating for parameter-efficient fine-tuning. *arXiv preprint arXiv:2405.12130*, 2024.

Damjan Kalajdzievski. A rank stabilization scaling factor for fine-tuning with lora. *arXiv preprint arXiv:2312.03732*, 2023.

Jiale Kang. Bone: block affine transformation as parameter efficient fine-tuning methods for large language models. 2024.

Dawid J Kopiczko, Sagar Vaze, Tijmen Blankevoort, and Yuki M Asano. Data repetition beats data scaling in long-cot supervised fine-tuning. *arXiv preprint arXiv:2602.11149*, 2026.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

Cheng Lin, Lujun Li, Dezhi Li, Jie Zou, Wenhan Luo, Wei Xue, and Yike Guo. Nora: Nested low-rank adaptation for efficient fine-tuning large models. *arXiv preprint arXiv:2408.10280*, 2024a.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Yelong Shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, and Nan Duan. Rho-1: Not all tokens are what you need. 2024b.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353*, 2024.

Wanlong Liu, Junxiao Xu, Fei Yu, Yukang Lin, Ke Ji, Wenyu Chen, Yan Xu, Yasheng Wang, Lifeng Shang, and Benyou Wang. Qfft, question-free fine-tuning for adaptive reasoning. *arXiv preprint arXiv:2506.12860*, 2025.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Jan 2018. doi: 10.18653/v1/d18-1260. URL http://dx.doi.org/10.18653/v1/d18-1260.

What Is Data Mining. Data mining: Concepts and techniques. *Morgan Kaufinann*, 10(559-569):4, 2006.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori B Hashimoto. s1: Simple test-time scaling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 20286–20332, 2025.

Vaishnavh Nagarajan and J Zico Kolter. Generalization in deep networks: The role of distance from initialization. *arXiv preprint arXiv:1901.01672*, 2019.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, et al. Olmo 3. *arXiv preprint arXiv:2512.13961*, 2025.

OpenAI. Learning to reason with llms, 2024. September 2024. URL https://openai.com/index/learning-to-reason-with-llms/.

OpenAI. Openai o3-mini. OpenAI release, 2025. Published January 31, 2025.

Jinlong Pang, Na Di, Zhaowei Zhu, Jiaheng Wei, Hao Cheng, Chen Qian, and Yang Liu. Token cleaning: Fine-grained data selection for llm supervised fine-tuning. 2025.

Qwen Team. Qwq-32b: Embracing the power of reinforcement learning. Qwen blog, 2025. Published March 6, 2025.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. In *First conference on language modeling*, 2024.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 8732–8740, Jun 2020. doi: 10.1609/aaai.v34i05.6399. URL http://dx.doi.org/10.1609/aaai.v34i05.6399.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. Socialiqa: Commonsense reasoning about social interactions. *Cornell University - arXiv,Cornell University - arXiv*, Apr 2019.

Matthew M Sato, Vivian Wen Hui Wong, Kincho H Law, Ho Yeung, Zhuo Yang, Brandon Lane, and Paul Witherell. Anomaly detection of laser powder bed fusion melt pool images using combined unsupervised and supervised learning methods. In *International design engineering technical conferences and computers and information in engineering conference*, volume 86212, pp. V002T02A070. American Society of Mechanical Engineers, 2022.

Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for llm problem-solving. In *The Thirteenth International Conference on Learning Representations*, 2025.

Yuxi Xie, Anirudh Goyal, Wenyue Zheng, Min-Yen Kan, Timothy P Lillicrap, Kenji Kawaguchi, and Michael Shieh. Monte carlo tree search boosts reasoning via iterative preference learning. *arXiv preprint arXiv:2405.00451*, 2024.

Xin Xu, Tianhao Chen, Fan Zhang, Wanlong Liu, Pengxiang Li, Ajay Kumar Jaiswal, Yuchen Yan, Jishan Hu, Yang Wang, Hao Chen, et al. Double-checker: Enhancing reasoning of slow-thinking llms via self-critical fine-tuning. *arXiv preprint arXiv:2506.21285*, 2025.

Kai Yao, Penglei Gao, Lichun Li, Yuan Zhao, Xiaofeng Wang, Wei Wang, and Jianke Zhu. Layer-wise importance matters: Less memory for better performance in parameter-efficient fine-tuning of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 1977–1992, 2024.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. Limo: Less is more for reasoning. *arXiv preprint arXiv:2502.03387*, 2025.

Fei Yu, Yingru Li, and Benyou Wang. Scaling flaws of verifier-guided search in mathematical reasoning. *arXiv preprint arXiv:2502.00271*, 2025a.

Zhaojian Yu, Yinghao Wu, Yilun Zhao, Arman Cohan, and Xiao-Ping Zhang. Z1: Efficient test-time scaling with code. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 2688–2712, 2025b.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Jan 2019. doi: 10.18653/v1/p19-1472. URL `http://dx.doi.org/10.18653/v1/p19-1472`.

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. *arXiv preprint arXiv:2303.10512*, 2023.

Yushun Zhang, Congliang Chen, Ziniu Li, Tian Ding, Chenwei Wu, Diederik Durk Kingma, Yinyu Ye, Zhi-Quan Luo, and Ruoyu Sun. Adam-mini: Use fewer learning rates to gain more. In *International Conference on Learning Representations*, volume 2025, pp. 28033–28063, 2025.

Yefan Zhou, Tianyu Pang, Keqin Liu, Michael W Mahoney, Yaoqing Yang, et al. Temperature balancing, layer-wise weight analysis, and neural network training. *Advances in Neural Information Processing Systems*, 36:63542–63572, 2023.

Terry Yue Zhuo, Minh Chien Vu, Jenny Chim, Han Hu, Wenhao Yu, Ratnadira Widyasari, Imam Nur Bani Yusuf, Haolan Zhan, Junda He, Indraneil Paul, et al. Bigcodebench: Benchmarking code generation with diverse function calls and complex instructions. *arXiv preprint arXiv:2406.15877*, 2024.

A Appendix

This appendix provides related work, additional results and implementation details to complement the main paper. It is organized as follows. Appendix A.1 introduce the related work. Appendix A.2 presents extended experimental results, including reasoning-intensive evaluations on Qwen3-4B and additional analyses across ranks and task types. Appendix A.2.1 further evaluates NormTune on GPQA, AIME'24, and AIME'25, while Appendix A.2.2 reports detailed results corresponding to the rank analysis and task comparison in the main paper. Appendix A.3 describes the additional benchmarks, experimental settings, and hardware configurations used in our experiments. Finally, Appendix A.4 describes the Prompts for Evaluation.

A.1 Related Work

A.1.1 Test-Time Scaling for Reasoning

Test-time scaling (OpenAI, 2024; Guo et al., 2025a; OpenAI, 2025; Bsharat & Shen, 2025; Xu et al., 2025; Kopiczko et al., 2026) has recently emerged as an effective way to improve reasoning performance by allocating more computation at inference time. Existing methods can be broadly grouped into two categories: reinforcement learning (RL) (OpenAI, 2025; 2024; Guo et al., 2025a) and supervised fine-tuning (SFT)(Bsharat & Shen, 2025; Xu et al., 2025; Kopiczko et al., 2026). For reinforcement learning (RL), despite their strong performance, these methods often depend on expensive post-training pipelines, are challenging to reproduce, or involve additional inference-time search components.

To address these limitations, recent studies have explored simpler and more reproducible forms of test-time scaling based on supervised fine-tuning (Xu et al., 2025; An et al., 2025; Kopiczko et al., 2026). Although these methods are generally easier to reproduce and adopt than reinforcement-learning-based approaches, training them still requires substantial computational resources, especially for large models, and often depends on access to high-memory GPUs.

A.1.2 Parameter-Efficient Fine-Tuning

Parameter-efficient fine-tuning (PEFT)(Houlsby et al., 2019; Li & Liang, 2021; Liu et al., 2024; Feng et al., 2025; Yao et al., 2024) is introduced to adapt pretrained models without updating all backbone parameters, thereby substantially reducing GPU memory consumption. Early PEFT methods(Houlsby et al., 2019; Li & Liang, 2021) add small task-specific modules or prompts while keeping the main model frozen. However, these methods often introduce additional inference latency and increase the computational cost of generation. To address this issue, low-rank adaptation(Zhang et al., 2023; Kalajdzievski, 2023; Hayou et al., 2024; Hu et al., 2021; Liu et al., 2024; Jiang et al., 2024), which can be merged into the backbone model during inference, has become the most widely used PEFT approach for large language models. Although these methods greatly improve memory efficiency during fine-tuning, they often underperform full fine-tuning, especially under low-rank settings. In this paper, we propose Norm-Guided Parameter-Efficient Fine-Tuning (NormTune), which adaptively enhances under-trained layers and thereby substantially improves performance on reasoning-intensive tasks.

A.2 Experiments

A.2.1 Reasoning-Intensive Experiments

We further evaluate NormTune on Qwen3-4B across GPQA, AIME'24, and AIME'25 benchmarks. As shown in Table 7, NormTune consistently improves different PEFT backbones across model families and reasoning benchmarks. On Qwen3-4B, NormTune+LoRA improves the average score from 37.7% to 44.2%, surpassing Full-FT by 6%, while NormTune+MoRA improves MoRA from 33.9% to 42.9%. NormTune+Bone also achieves consistent gains and reaches performance comparable to Full-FT. However, TempBalance+LoRA, TempBalance+Bone, and TempBalance+MoRA reduced performance by 0.7%, 11%, and 17.3% compared with LoRA, Bone, and MoRA, respectively.

A.2.2 Other Experiments

We provide additional results to further support the observations in the main experiments. Table 6 reports the detailed results corresponding to Figure 5a. Across all ranks, NormTune consistently improves LoRA on reasoning-intensive tasks. The gains are especially pronounced at low ranks. Notably, when the rank is set to 256, NormTune+LoRA achieves an average accuracy of 68.6%, slightly surpassing Full-FT, which further demonstrates the effectiveness of the proposed method.

Tables 11, 16, and 17 provide the detailed results corresponding to Figure 1a. These results highlight a clear difference between reasoning-intensive tasks and other task types. On commonsense reasoning and long-context understanding tasks, PEFT methods with rank 8 already achieve performance close to Full-FT. In contrast, the same low-rank PEFT setting performs substantially worse on reasoning-intensive tasks, as shown in Table 11.

Table 6: Accuracy (%) of LoRA and NormTune+LoRA with different ranks on reasoning-intensive tasks

MODEL	METHOD	RANK	AIME 2024 AVG@16	MATH 500 AVG@4	GPQA AVG@4	AIME 2025 AVG@16	AVG.
QWEN2.5-32B -INSTRUCT	FULL-FT	-	56.7	95.4	63.6	50	66.43
	LORA	8	10	78.8	47.47	13.33	37.4
	NORMTUNE+LORA	8	53.33	92.6	60.61	40	61.64
	LORA	64	30	90.8	55.05	33.33	52.3
	NORMTUNE+LORA	64	66.67	90.6	58.59	46.67	65.63
	LORA	128	26.67	90.2	58.59	40	53.87
	LORA +NORMTUNE	128	60	91.8	60.61	53.33	66.44
	LORA	256	36.67	92.6	58.08	33.3	55.16
	LORA +NORMTUNE	256	63.33	93.6	64.14	53.33	**68.6**
	LORA	512	40	91.4	59.09	53.33	60.96
	LORA +NORMTUNE	512	56.67	94.2	63.13	46.67	65.17

Table 7: Performance of Full-FT, PEFT, and NormTune-enhanced PEFT methods on Qwen3-4B across GPQA, AIME'24, and AIME'25. * denotes results reported in Kopiczko et al. (2026).

Model	Method	GPQA		AIME'24		AIME'25		Average
		Avg@4	Pass@4	Avg@16	Pass@16	Avg@16	Pass@16	
Qwen3-4B	Full-FT*	39.3	68.7	19.2	43.3	18.8	40.0	38.2
	AdaLoRA	21.8	54	7.1	30	9.8	26.7	24.9
	DoRA	38.1	72.2	15	33.3	16.3	43.3	36.4
	RSLoRA	21.3	51.5	6.9	26.7	5.8	20	22.0
	LoRA+	27	62.1	12.5	36.7	8.1	30	29.4
	LoRA(256)	39	66.2	15	40	19.4	46.7	37.7
	TempBalance + LoRA	38.8	70.7	14.2	40	14.8	43.3	37
	NormTune+LoRA(256)	42.6	70.2	22.7	56.6	20.0	53.3	**44.2**
	Bone(256)	35.1	63.6	15	33.3	14.17	40	33.5
	TempBalance + Bone	27.3	59.6	4.6	20	3.6	20	22.5
	NormTune+Bone(256)	42.3	67.7	16.5	36.7	21.9	50	**39.2**
	MoRA(256)	38.3	69.7	15.2	30	10	40	33.9
	TempBalance + MoRA	23.4	58.6	0.4	6.7	0.6	10	16.6
	NormTune+MoRA(256)	41.3	71.7	20.8	46.7	20.2	56.7	**42.9**

In addition, we further conduct a sensitivity study using 10 (before warm-up), 30, and 50 (after warm-up) steps with LoRA (rank=256) under the same setting as Table 1. As shown in Table 12, 10 steps lead to noticeably worse performance, while the results remain stable after 30 steps, supporting our choice. Since the probing steps are included in the total training budget, using 50 steps delays the activation of NormTune by 20 updates compared to 30 steps, which may affect norm adaptation and lead to a slight performance drop. We also conduct an additional ablation on Qwen3-4B using NormTune+LoRA(256) on GPQA. Figure 6 show the same trend: performance is relatively insensitive to n, and $n = 16$ remains a strong default. This suggests that the optimal bucket number does not strongly depend on model size or depth in our tested settings.

A.3 ENVIRONMENT AND SETTINGS

All experiments are conducted on eight NVIDIA A800 80GB GPUs. Unless otherwise specified, we follow the original training settings of S1 Muennighoff et al. (2025), Z1 Yu et al. (2025b), and Data-Repetition Kopiczko et al. (2026) for the main hyperparameters, including the warmup schedule, the number of training steps, the optimizer, and the use of gradient checkpointing. We use Qwen3-4B, Qwen-2.5-Coder-7B-InstructYu et al. (2025b), OLMo-3-7BOlmo et al. (2025) and Qwen2.5-32B-Instruct as the base models for our PEFT experiments across different benchmarks. Unless otherwise specified, we compute layer-wise norm after training step 30, which is included in the full training steps, set the number of buckets to $n = 16$, set base learning rate to the $\eta_{\min}$ of each PEFT method, set the random seed to 42 during training, and use $\eta_{\max} = 1 \times 10^{-4}$. In addition, to ensure a fair comparison, Full-FT and PEFT use identical optimization hyperparameters, following the configurations adopted in the corresponding papers, which also include experiments on the GovReport Huang et al. (2021) and commonsense reasoning benchmarks. NormTune does not require a separate probing run. The first 30 updates are included in the normal training budget.

Table 8: Accuracy (%) of LoRA with different ranks on reasoning-intensive tasks.

MODEL	METHOD	RANK	AIME 2024 AVG@16	MATH 500 AVG@4	GPQA AVG@4	AIME 2025 AVG@16	AVG.
QWEN2.5-32B -INSTRUCT	FULL-FT	-	56.7	95.4	63.6	50	66.43
	LORA	8	10	78.8	47.47	13.33	37.4
	LORA	64	30	90.8	55.05	33.33	52.3
	LORA	128	26.67	90.2	58.59	40	53.87
	LORA	256	36.67	92.6	58.08	33.33	55.16
	LORA	512	40	91.4	59.09	53.33	60.96

Table 9: Accuracy (%) of full fine-tuning and rank-256 PEFT methods under two learning-rate settings on reasoning-intensive tasks.

METHOD	LEARNING-RATE	RANK	AIME 2024 AVG@16	MATH 500 AVG@4	GPQA AVG@4	AIME 2025 AVG@16	AVG.
S1 (FULL-FT)	1E-5	-	56.7	95.4	63.6	50	66.43
S1 (FULL-FT)	1E-4	-	10	5.9	3.0	0.7	4.9
LORA	1E-5	256	36.67	92.6	58.08	33.3	55.16
LORA	1E-4	256	36.67	89.6	52.53	20	49.7
BONE	1E-5	256	43.33	93.4	57.58	33.3	56.90
BONE	1E-4	256	33.3	90.6	52.02	46.67	55.65
MORA	1E-5	256	50	93.2	62.12	33.3	59.66
MORA	1E-4	256	26.67	89.4	48.48	33.33	49.47

The additional cost of the whole probing stage is negligible: on Qwen2.5-32B-Instruct, it requires only 0.018 GPU-hours on 8 A800 GPUs.

Beyond S1K-1.1 Muennighoff et al. (2025), Z1-Code-Reasoning107K Yu et al. (2025b), and Data-Repetition Kopiczko et al. (2026), we further fine-tune and evaluate on two additional benchmarks: commonsense reasoning and long-context understanding. For commonsense reasoning, we consider eight representative tasks, including HellaSwag Zellers et al. (2019), OpenBookQA (OBQA) Mihaylov et al. (2018), Social IQA (SIQA) Sap et al. (2019), WinoGrande Sakaguchi et al. (2020), BoolQ Clark et al. (2019), PIQA Bisk et al. (2020), ARC-Challenge (ARC-c), and ARC-Easy (ARC-e) Clark et al. (2018). For long-context understanding, we use GovReport Huang et al. (2021), a long-document summarization benchmark consisting of reports from U.S. government agencies and their corresponding summaries. We randomly sample 1,000 examples from the training split for fine-tuning and evaluate the resulting models on the official test split. These evaluations enable a comprehensive assessment of the effectiveness of our method. The detailed experimental configurations for full fine-tuning and PEFT methods, including LoRA, Bone, and MoRA, are reported in Tables 19, 20, 21, and 22. We set $a = 16,512$ for RsLoRA and $exp_temp = 10$ and $esd_update_steps = 2$ for TempBalance. All experiments are conducted on eight NVIDIA A800 GPUs and completed within 24 hours.

A.4 PROMPTS FOR EVALUATION

For the S1K-1.1 benchmark, we adopt the prompts provided in the lm-evaluation-harness GitHub repository. For Z1, we adopt the prompts from the Qwen2.5-Coder GitHub repository for LiveCodeBench and BigCodeBench, while using the following prompts for MATH-500 and GPQA Diamond:

Table 10: Accuracy (%) of different parameter-efficient fine-tuning methods under various adjustment styles and ranks on reasoning-intensive tasks.

METHOD	ADJUSTMENT STYLE	RANK	AIME 2024 AVG@16	MATH 500 AVG@4	GPQA AVG@4	AIME 2025 AVG@16	AVG.
FULL-FT	-	-	56.7	95.4	63.6	50	66.43
FULL-FT	STAGE	-	33.33	85.4	46.97	20	46.43
LORA	-	256	36.67	92.6	58.08	33.33	55.16
LORA	LINEAR	256	53.3	93.8	62.63	33.3	60.76
LORA	STAGE	256	46.67	93.8	65.66	46.67	63.2
LORA	EXPONENTIAL	256	63.33	91.4	63.13	33.33	62.79
LORA	NORMTUNE	256	63.33	93.6	64.14	53.33	**68.6**
BONE	-	256	43.33	93.4	57.58	33.3	56.90
BONE	LINEAR	256	50	93.6	55.56	40	59.79
BONE	STAGE	256	60	92	61.11	40	63.28
BONE	EXPONENTIAL	256	43.33	93.2	64.14	40	60.17
BONE	NORMTUNE	256	56.67	94.8	64.65	53.33	**67.36**
MORA	-	256	50	93.2	62.12	33.3	59.66
MORA	LINEAR	256	56.67	93	64.14	33.33	61.79
MORA	STAGE	256	50	94	59.09	40	60.78
MORA	EXPONENTIAL	256	56.67	92.8	62.63	40	63.03
MORA	NORMTUNE	256	60	94.6	63.13	60	**69.43**

Table 11: Accuracy (%) of different parameter-efficient fine-tuning methods on reasoning-intensive tasks.

METHOD	RANK	AIME 2024 AVG@16	MATH 500 AVG@4	GPQA AVG@4	AIME 2025 AVG@16	AVG.
FULL-FT	-	56.7	95.4	63.6	50	66.43
LORA	8	10	78.8	47.47	13.33	37.4
BONE	8	26.67	80.4	52.53	40	49.9
MORA	8	26.67	80.4	51.01	20	44.52

Prompts

```
<|im_start|>system
Please reason step by step, and put your final answer within \boxed{}.
<|im_end|>
<|im_start|>user
{problem}
<|im_end|>
<|im_start|>assistant
```

For Data-Repetition, we follow the evaluation protocol in the official implementation. AIME 2024 and AIME 2025 use the same prompt, which requires the final integer answer to be enclosed in `\boxed`. For GPQA Diamond, the model is instructed to place the letter corresponding to the selected answer in `\boxed`. Each query is formatted using the model's default chat template.

For commonsense reasoning, we evaluate on BoolQ, PIQA, SIQA, HellaSwag, WinoGrande, ARC-Easy, ARC-Challenge, and OpenBookQA. For each benchmark, the task-specific instruction is wrapped with the Alpaca-style prompt template used in the official DoRA implementation:

Prompts

Below is an instruction that describes a task. Write a response that appropriately completes the request.

```
### Instruction:
```

Table 12: Sensitivity analysis of the number of probing steps with NormTune+LoRA (rank=256).

METHOD	RANK	PROBING STEPS	AIME 2024	MATH 500	GPQA	AIME 2025	AVG.
NORMTUNE+LORA (OURS)	256	10	53.33	93.6	64.14	20	57.77
NORMTUNE+LORA (OURS)	256	30	63.33	93.6	64.14	53.33	68.6
NORMTUNE+LORA (OURS)	256	50	63.33	93.4	62.12	53.33	68.05

Table 13: Sensitivity analysis of the bucket number n on Qwen3-4B using NormTune+LoRA (rank=256) on GPQA.

METHOD	RANK	n	AVG@4
NORMTUNE+LORA (OURS)	256	4	41.3
NORMTUNE+LORA (OURS)	256	8	40.9
NORMTUNE+LORA (OURS)	256	16	42.6
NORMTUNE+LORA (OURS)	256	32	41.7

```
{instruction}
### Response:
```

For GovReport, we adopt the following prompt:

Prompts

Read the following government report and write a faithful, comprehensive executive summary. Focus on the key findings, background, and conclusions. Do not invent information.

```
### Report
{report}
### Executive Summary
```

Table 14: Delete? Performance of LoRA and NormTune+LoRA (rank=8) on commonsense reasoning and long-context understanding benchmarks.

BENCHMARK	METHOD	AVG.
COMMONSENSE REASONING	LORA	90.66
COMMONSENSE REASONING	NORMTUNE+LORA (OURS)	90.98
GOVREPORT	LORA	32.44
GOVREPORT	NORMTUNE+LORA (OURS)	34.21

Table 15: Learning rate sensitivity analysis of LoRA with rank 256.

METHOD	RANK	LEARNING RATE	AVG.
LORA	256	6E-6	52.1
LORA	256	8E-6	52.9
LORA	256	1E-5	55.16
LORA	256	3E-5	53.36
LORA	256	5E-5	52.0
LORA	256	1E-4	49.7

Table 16: Accuracy comparison of different parameter-efficient fine-tuning methods across multiple commonsense reasoning benchmarks.

METHOD	RANK	BOOLQ	PIQA	SIQA	WINOGRANDE	ARC-E	OBQA	HELLASWAG	ARC-C	AVG.
FULL-FT	-	77.46	93.96	81.99	92.34	97.73	97.2	96.24	92.32	91.16
LORA	8	76.23	92.6	81.06	89.02	98.56	95.8	96.99	95.05	90.66
BONE	8	73.37	92.93	82.44	89.5	98.44	95.6	96.56	94.96	90.48
MORA	8	72.75	91.45	80.91	85.48	98.27	94.8	96.22	94.37	89.28

Table 17: ROUGE results (%) on the GovReport dataset.

METHOD	RANK	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-LSUM	AVERAGE
FULL-FT	-	55.11	27.39	28.95	28.95	35.10
LORA	8	52.6	20.63	24.67	31.84	32.44
BONE	8	60.64	30.45	31.94	31.93	38.74
MORA	8	58.8	30	31.29	31.08	37.79

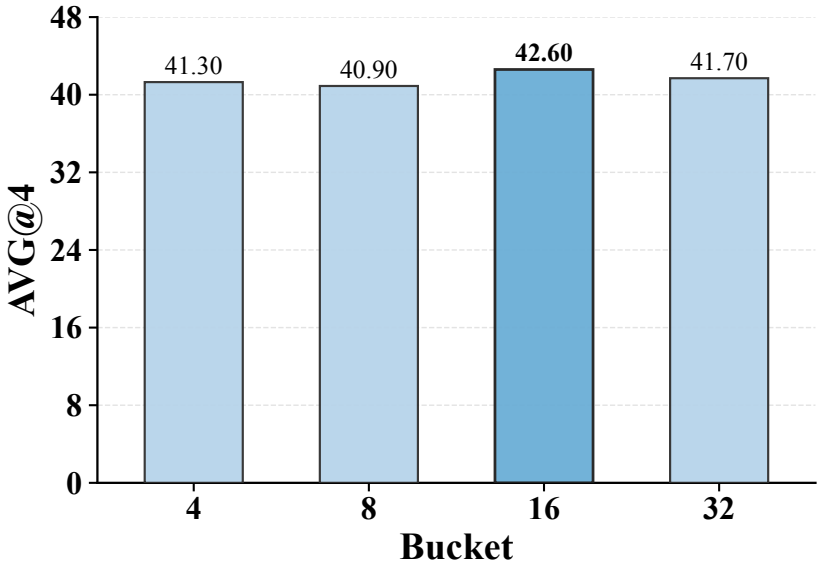


Figure 6: Effect of the bucket number n in NormTune on GPQA after training on Qwen3-4B.

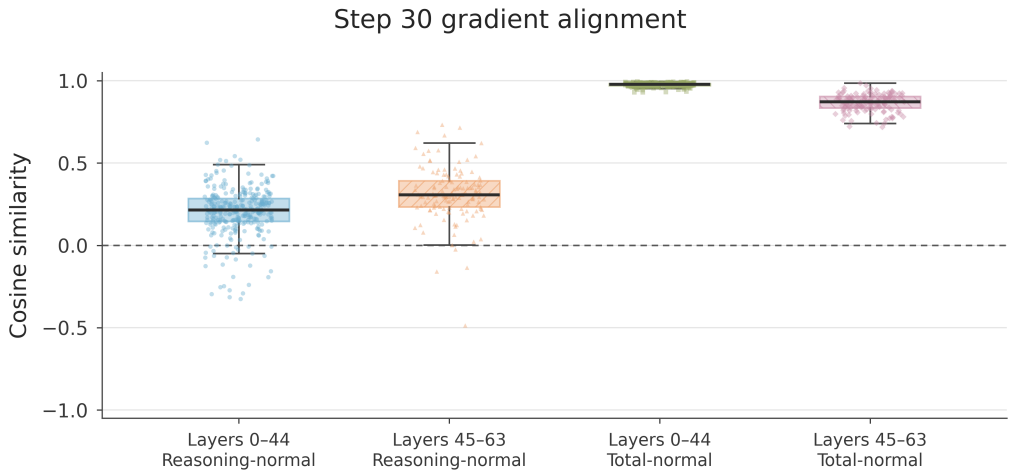


Figure 7: Gradient alignment across early and late layers, measured by $\mathrm{CosSim}(\nabla_\theta \mathcal{L}_{\text{reasoning}}, \nabla_\theta \mathcal{L}_{\text{normal}})$ and $\mathrm{CosSim}(\nabla_\theta \mathcal{L}_{\text{total}}, \nabla_\theta \mathcal{L}_{\text{normal}})$.

Table 18: Hyperparameter settings for GovReport and commonsense reasoning training configurations.

Hyperparameter	GovReport				commonsense reasoning			
	Full-FT	LoRA	Bone	MoRA	Full-FT	LoRA	Bone	MoRA
Batch size	16	16	16	16	128	128	128	128
Optimizer	Adam	Adam	Adam	Adam	Adam	Adam	Adam	Adam
LR	1e-5	1e-5	1e-5	1e-5	3e-5	3e-5	3e-5	3e-5
Warmup Ratio	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
Epoch	5	5	5	5	3	3	3	3
Weight Decay	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4
Rank	–	8	8	8	–	8	8	8
LoRA α	–	16	–	–	–	16	–	–
MoRA Type	–	–	–	6	–	–	–	6
Block Size	32768	32768	32768	32768	256	256	256	256

Table 19: NormTune-specific configuration for different training benchmark. _ provides the optimal choice.

Hyperparameter	S1K-1.1	Z1-Code-Reasoning107K	Data-repetition	OLMo-3-7B
Batch size	16	128	1	1
Optimizer	Adam	Adam	8-bit Adam	8-bit Adam
$\eta_{\min}$	1e-5	1e-5	1e-5, 3e-5	2e-5
$\eta_{\max}$	1e-4,1e-3	2e-5,1e-4	2e-5,3e-5,1e-4	1e-4
n buckets	16	16	16	16
Probing Steps	30	30	30	30
Warmup Ratio	0.05	0.06	0.1	0.1
Epoch	5	2	16	16
Weight Decay	1e-4	0.0	0.0	0.0
Target Module	Q, K, V, O, Gate, Up, Down			
Block Size	32768	8192	32768	32768

Table 20: Hyperparameter settings for different PEFT training configurations.

Configuration	Method	Batch Size	Optimizer	LR	Warmup Ratio	Epoch	Weight Decay	Rank	LoRA α	MoRA Type	AdaLoRA_init_rank	Block Size
S1K-1.1	Full-FT	16	Adam	1e-5	0.05	5	1e-4	–	–	–	–	32768
	LoRA	16	Adam	1e-5	0.05	5	1e-4	8,256	16,512	–	–	32768
	Bone	16	Adam	1e-5	0.05	5	1e-4	8,256	–	–	–	32768
	MoRA	16	Adam	1e-5	0.05	5	1e-4	8,256	–	6,1	–	32768
	AdaLoRA	16	Adam	1e-5	0.05	5	1e-4	8,256	–	–	12,384	32768
	DoRA	16	Adam	1e-5	0.05	5	1e-4	8,256	–	–	–	32768
	RSLoRA	16	Adam	1e-5	0.05	5	1e-4	8,256	–	–	–	32768
	LoRA+	16	Adam	1e-5	0.05	5	1e-4	8,256	–	–	–	32768
	TempBalance	16	Adam	1e-5	0.05	5	1e-4	8,256	–	–	–	32768
Z1-Code-Reasoning107K	Full-FT	128	Adam	1e-5	0.06	2	0.0	256	–	–	–	8192
	LoRA	128	Adam	1e-5	0.06	2	0.0	256	512	–	–	8192
	Bone	128	Adam	1e-5	0.06	2	0.0	256	–	–	–	8192
	MoRA	128	Adam	1e-5	0.06	2	0.0	256	–	1	–	8192
	AdaLoRA	128	Adam	1e-5	0.06	2	0.0	256	–	–	384	8192
	DoRA	128	Adam	1e-5	0.06	2	0.0	256	–	–	–	8192
	RSLoRA	128	Adam	1e-5	0.06	2	0.0	256	–	–	–	8192
	LoRA+	128	Adam	1e-5	0.06	2	0.0	256	–	–	–	8192
	TempBalance	128	Adam	1e-5	0.06	2	0.0	256	–	–	–	8192
Data-repetition	Full-FT	1	8-bit Adam	3e-5	0.1	16	0.0	256	–	–	–	32768
	LoRA	1	8-bit Adam	3e-5	0.1	16	0.0	256	512	–	–	32768
	Bone	1	8-bit Adam	3e-5	0.1	16	0.0	256	–	–	–	32768
	MoRA	1	8-bit Adam	3e-5	0.1	16	0.0	256	–	1	–	32768
	AdaLoRA	1	8-bit Adam	3e-5	0.1	16	0.0	256	–	–	384	32768
	DoRA	1	8-bit Adam	3e-5	0.1	16	0.0	256	–	–	–	32768
	RSLoRA	1	8-bit Adam	3e-5	0.1	16	0.0	256	–	–	–	32768
	LoRA+	1	8-bit Adam	3e-5	0.1	16	0.0	256	–	–	–	32768
	TempBalance	1	8-bit Adam	3e-5	0.1	16	0.0	256	–	–	–	32768
OLMo-3-7B	Full-FT	1	8-bit Adam	2e-5	0.1	16	0.0	256	–	–	–	32768
	LoRA	1	8-bit Adam	2e-5	0.1	16	0.0	256	512	–	–	32768
	Bone	1	8-bit Adam	2e-5	0.1	16	0.0	256	–	–	–	32768
	MoRA	1	8-bit Adam	2e-5	0.1	16	0.0	256	–	1	–	32768
	AdaLoRA	1	8-bit Adam	2e-5	0.1	16	0.0	256	–	–	384	32768
	DoRA	1	8-bit Adam	2e-5	0.1	16	0.0	256	–	–	–	32768
	RSLoRA	1	8-bit Adam	2e-5	0.1	16	0.0	256	–	–	–	32768
	LoRA+	1	8-bit Adam	2e-5	0.1	16	0.0	256	–	–	–	32768
	TempBalance	1	8-bit Adam	2e-5	0.1	16	0.0	256	–	–	–	32768

Table 21: Hyperparameter settings for TempBalance configurations.

Configuration	Method	Batch Size	Optimizer	LR	Warmup Ratio	Epoch	Weight Decay	Rank	LoRA α	MoRA Type	AdaLoRA_init_rank	Block Size
S1K-1.1	Full-FT	16	Adam	1e-5	0.05	5	1e-4	–	–	–	–	32768
	LoRA	16	Adam	1e-5	0.05	5	1e-4	8,256	16,512	–	–	32768
	Bone	16	Adam	1e-5	0.05	5	1e-4	8,256	–	–	–	32768
	MoRA	16	Adam	1e-5	0.05	5	1e-4	8,256	–	6,1	–	32768
	AdaLoRA	16	Adam	1e-5	0.05	5	1e-4	8,256	–	–	12,384	32768
	DoRA	16	Adam	1e-5	0.05	5	1e-4	8,256	–	–	–	32768
	RSLoRA	16	Adam	1e-5	0.05	5	1e-4	8,256	–	–	–	32768
	LoRA+	16	Adam	1e-5	0.05	5	1e-4	8,256	–	–	–	32768
	TempBalance	16	Adam	1e-5	0.05	5	1e-4	8,256	–	–	–	32768
Z1-Code-Reasoning107K	Full-FT	128	Adam	1e-5	0.06	2	0.0	256	–	–	–	8192
	LoRA	128	Adam	1e-5	0.06	2	0.0	256	512	–	–	8192
	Bone	128	Adam	1e-5	0.06	2	0.0	256	–	–	–	8192
	MoRA	128	Adam	1e-5	0.06	2	0.0	256	–	1	–	8192
	AdaLoRA	128	Adam	1e-5	0.06	2	0.0	256	–	–	384	8192
	DoRA	128	Adam	1e-5	0.06	2	0.0	256	–	–	–	8192
	RSLoRA	128	Adam	1e-5	0.06	2	0.0	256	–	–	–	8192
	LoRA+	128	Adam	1e-5	0.06	2	0.0	256	–	–	–	8192
	TempBalance	128	Adam	1e-5	0.06	2	0.0	256	–	–	–	8192
Data-repetition	Full-FT	1	8-bit Adam	3e-5	0.1	16	0.0	256	–	–	–	32768
	LoRA	1	8-bit Adam	3e-5	0.1	16	0.0	256	512	–	–	32768
	Bone	1	8-bit Adam	3e-5	0.1	16	0.0	256	–	–	–	32768
	MoRA	1	8-bit Adam	3e-5	0.1	16	0.0	256	–	1	–	32768
	AdaLoRA	1	8-bit Adam	3e-5	0.1	16	0.0	256	–	–	384	32768
	DoRA	1	8-bit Adam	3e-5	0.1	16	0.0	256	–	–	–	32768
	RSLoRA	1	8-bit Adam	3e-5	0.1	16	0.0	256	–	–	–	32768
	LoRA+	1	8-bit Adam	3e-5	0.1	16	0.0	256	–	–	–	32768
	TempBalance	1	8-bit Adam	3e-5	0.1	16	0.0	256	–	–	–	32768
OLMo-3-7B	Full-FT	1	8-bit Adam	2e-5	0.1	16	0.0	256	–	–	–	32768
	LoRA	1	8-bit Adam	2e-5	0.1	16	0.0	256	512	–	–	32768
	Bone	1	8-bit Adam	2e-5	0.1	16	0.0	256	–	–	–	32768
	MoRA	1	8-bit Adam	2e-5	0.1	16	0.0	256	–	1	–	32768
	AdaLoRA	1	8-bit Adam	2e-5	0.1	16	0.0	256	–	–	384	32768
	DoRA	1	8-bit Adam	2e-5	0.1	16	0.0	256	–	–	–	32768
	RSLoRA	1	8-bit Adam	2e-5	0.1	16	0.0	256	–	–	–	32768
	LoRA+	1	8-bit Adam	2e-5	0.1	16	0.0	256	–	–	–	32768
	TempBalance	1	8-bit Adam	2e-5	0.1	16	0.0	256	–	–	–	32768

Table 22: Evaluation settings for different training configurations. For S1K-1.1 and Z1-Code-Reasoning107K, mean greedy accuracy is averaged over repeated evaluation runs of the same checkpoint.

Configuration	Evaluation suites	Metric / evaluator	Decoding	Max tokens	Completions per problem per run	Greedy evaluation runs (R)
S1K-1.1	AIME 2024, AIME 2025, MATH 500, GPQA	Mean greedy accuracy (GPT-4o-mini)	Greedy, $T = 0$	32768	1	4: MATH 500, GPQA; 16: AIME 2024, AIME 2025
Z1-Code-Reasoning107K	LiveCodeBench, MATH 500, BigCodeBench-Hard, GPQA	Mean greedy accuracy (Qwen2.5-32B-Instruct)	Greedy, $T = 0$	4096	1	4: MATH 500, GPQA; 16: LiveCodeBench, BigCodeBench-Hard
Data-repetition	GPQA, AIME 2024, AIME 2025	Avg@k and Pass@k	Sampling, $T = 0.6$, top-$p = 0.95$, top-$k = 20$	32768	4 (GPQA) 16 (AIME)	—
OLMo-3-7B	GPQA, AIME 2024, AIME 2025	Avg@k and Pass@k	Sampling, $T = 0.6$, top-$p = 0.95$, top-$k = 50$	32768	4 (GPQA) 16 (AIME)	—
GovReport	GovReport test split	ROUGE-1, ROUGE-2, ROUGE-L, ROUGE-Lsum	Sampling, $T = 0.1$, top-$p = 0.9$	32768	1	—
Commonsense reasoning	BoolQ, PIQA, SIQA, WinoGrande, ARC-e/c, OBQA, HellaSwag	Accuracy	4-beam search; $T = 0.1$, top-$p = 0.75$, top-$k = 40$ in the generation configuration	32	1	—

Note. For S1K-1.1 and Z1-Code-Reasoning107K, each greedy evaluation run generates one completion per problem. The reported accuracy is the arithmetic mean of the accuracies from R evaluation runs using the same checkpoint and benchmark questions. These runs are not independent training runs or stochastic decoding samples. For Data-repetition and OLMo-3-7B, k denotes the number of sampled completions per problem used to compute Avg@k and Pass@k. A dash in the last column indicates that the repeated-greedy evaluation protocol does not apply.