SFT CONFLICTS, RL COEXISTS: A THEORETICAL AND EMPIRICAL ANALYSIS OF MULTI-TASK LEARNING FOR LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Supervised Fine-Tuning (SFT) and Reinforcement Learning (RL) exhibit fundamentally different behaviors in enhancing multi-task reasoning for large language models. Our preliminary experiments revealed a phenomenon: SFT suffers from task conflicts under multi-stage training, whereas RL enables coexistence across diverse tasks. Empirically, we trace this to parameter level, observing that RL induces sparse and approximately orthogonal updates across tasks. We provide a theoretical explanation for this mechanism by analyzing multi-task gradient interference. Our results reveal a distinction: interference in SFT is norm-limited, scaling with absolute gradient magnitude, whereas interference in RL is variance-limited, bounded by gradient variance induced by advantage normalization and on-policy optimization. This small variance bound yields near-orthogonal optimization directions across tasks. Leveraging this insight, we propose Parallel-RL, a paradigm that decouples multi-task training, improving efficiency and flexibility.

1 INTRODUCTION

Enhancing the reasoning capabilities of large language models (LLMs) is a central research focus, with Supervised Fine-Tuning (SFT) and Reinforcement Learning (RL) as the dominant paradigms (Jaech et al., 2024; Guo et al., 2025). Recent works have explored their differences in single-task training (Chu et al., 2025; Matsutani et al., 2025). But their behaviors in multi-task settings, which is also crucial for achieving Artificial General Intelligence (AGI), remain underexplored. We observe that in most existing works, researchers typically construct mixed datasets from multiple tasks for SFT (Dong et al., 2023; Park, 2025). In contrast, for RL, apart from using mixed-data training, many works commonly adopts multi-stage training, where each stage focuses on a single task (Narvekar et al., 2020; Cho et al., 2024). To explain this divergence, we conduct a comprehensive **empirical and theoretical analysis** comparing SFT and RL in multi-task training.

We first tested the performance differences between SFT and RL when using mixed-data and multi-stage strategy for multi-task training. Table 1 reveals a striking contrast: SFT suffers from performance collapses during multi-stage training, showing an average decline of 23.1% compared to base model. Conversely, RL demonstrates robustness improvement (+24.9%) in multi-stage settings. Therefore, one of the key questions of our research is this difference: **why SFT collapses under multi-stage training whereas RL achieves stable, cumulative performance growth on different tasks across multi-stage training?** To answer this question, we examined the impact of single-task training on untrained tasks. As shown in Table 2, while SFT improves target task, it causes substantial degradation in others. Conversely, RL enhances the target task while preserving performance on untrained tasks. We term this phenomenon **SFT Conflicts** and **RL Coexists**. This distinction in generalization directly answers our inquiry. As illustrated in Figure 1, although SFT parameters updates are toward improving performance of each single task during single-stage training, severe conflicts arise across multi-stages, leading to performance collapse in the final multi-stage SFT model. In contrast, parameter updates induced by RL across different tasks are approximately orthogonal and can therefore coexist, enabling multi-stage RL to stably improve performance across tasks.

To understand the mechanisms behind these phenomena, we conduct an **empirical analysis** of parameter updates ΔW (§3). As illustrated in Figure 1, we highlight two observations. First,

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.

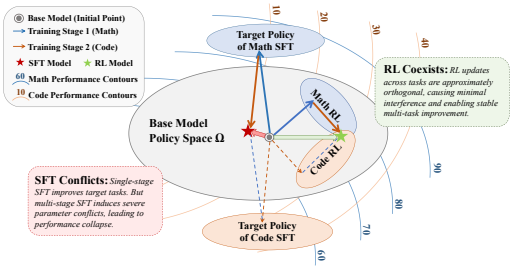

Figure 1: SFT and RL exhibit significant differences during **multi-stage** training. SFT shows task conflicts that impair performance, while RL can steadily improve performance across tasks.

Table 1: Accuracy (%) of SFT and RL with different strategies. Superscripts denote performance changes relative to the Base Model.

Strategy	Math	Science	Logic	Code
		Baseline		
Base Model	$83.1_{\pm 1.6}$	$34.9_{\pm 1.1}$	$31.0_{\pm 2.8}$	$15.0_{\pm 2.3}$
Mixed Data SFT	$84.6^{\uparrow 1.5}_{\pm 1.6}$	$38.9^{\uparrow 4.0}_{\pm 1.1}$	$34.0^{\uparrow 3.0}_{\pm 2.5}$	$16.0^{\uparrow 1.0}_{\pm 2.2}$
Multi Stage SFT	$78.2^{\downarrow 4.9}_{\pm 1.7}$	$31.1^{\downarrow 3.8}_{\pm 1.2}$	$9.0^{\downarrow 22.0}_{\pm 2.1}$	$14.3^{\downarrow 0.7}_{\pm 2.4}$
Mixed Data RL	$85.2^{\uparrow 2.1}_{\pm 1.5}$	$43.2^{\uparrow 8.3}_{\pm 1.1}$	$37.0^{\uparrow 6.0}_{\pm 2.4}$	$15.7^{\uparrow 0.7}_{\pm 2.3}$
Multi Stage RL	$86.6^{\uparrow 3.5}_{\pm 1.5}$	$49.3^{\uparrow 14.4}_{\pm 1.0}$	$43.0^{\uparrow 12.0}_{\pm 2.6}$	$17.3^{\uparrow 2.3}_{\pm 2.1}$

regarding the *magnitude*, RL induces significantly smaller updates, with an average magnitude over two orders of magnitude smaller than SFT. Second, regarding the interference between different ΔW, the cross-task update similarity is negligible for RL ($\sim 10^{-5}$) but substantial for SFT ($\sim 10^{-1}$).

We then provide a theoretical analysis in §4 to explain these observations by comparing the gradient directions of SFT and RL. Specifically, their two key differences between gradients lie in **Advantage Function** and **Policy Source**. Thus in §4.1 and §4.2, we qualitatively analyze how advantage function and on-policy optimization drive the observed sparsity of ΔW and small interference between different tasks. Furthermore, in §4.3, we derive the upper bound of gradient interference between tasks. Our proofs demonstrate that SFT interference is **norm-limited** and depends on absolute gradient magnitude. Conversely, RL interference is **variance-limited** and is bounded by variance of multiple rollouts (caused by both advantage and on-policy). This theorem clarifies the distinct interference levels in multi-task learning and reflects the fundamental objectives of SFT and RL.

Building on the insight that RL optimization directions across tasks exhibit approximate orthogonality, we propose **Parallel-RL**, a method for multi-task RL. In this framework, tasks are trained independently in parallel and their updates are subsequently merged. Experiments show that Parallel-RL matches single-task performance while improving training efficiency, with ablations confirming effective decoupling of task-specific capabilities. Guided by our analysis, we note that Parallel-RL goes beyond model merging and requires techniques for task selection and single-task training.

2 Preliminaries

In this section, we provide preliminary evidence of the fundamental differences between SFT and RL in multi-task training and examine their cross-task generalization after single-task training.

2.1 Experiment Setup

We employ DeepSeek-R1-Distill-Qwen-1.5B as base model. Our experiments cover four representative reasoning domains: math, science, coding, and logic. For RL, we adopt the GRPOGuo et al. (2025). We utilized LoRA to facilitate efficient fine-tuning and analysis for both RL and SFT. Here we select MATH500, MMLU, Knights & Knaves, and LiveCodeBench as benchmarks for Math, Science, Logic, and Code tasks, respectively. Implementation details are provided in Appendix A.

2.2 Differences of Training Paradigms

There are two paradigms for multi-task learning: **Mixed-Data**, which aggregates datasets from diverse tasks for joint training, and **Multi-Stage**, which trains tasks sequentially. For SFT, Mixed-Data is widely adopted to mitigate forgetting (Dong et al., 2023; Park, 2025). In contrast, while also utilizing Mixed-Data, RL often follows a Multi-Stage paradigm (Narvekar et al., 2020; Cho et al., 2024). To understand this difference, we conducted an experiment comparing two strategies, with results presented in Table 1. We observe that under multi-stage training, SFT suffers a significant decline across tasks, averaging **23.1% below** the base model, whereas mixed-data SFT yields a **7.4% gain**. Conversely, RL exhibits robust improvements in both multi-stage and mixed-data settings, achieving

Table 2: Accuracy (%) across all evaluation tasks after single-task training. Subscripts report uncertainty estimates; superscripts indicate changes from the base model in percentage points.

Training task	Math		Science		Logic		Code	
	SFT	RL	SFT	RL	SFT	RL	SFT	RL
Base model	$82.0_{\pm 1.6}$		$34.9_{\pm 1.1}$		$31.0_{\pm 2.8}$		$15.0_{\pm 2.3}$	
Math	$84.4^{\uparrow 2.4}_{\pm 1.5}$	$85.6^{\uparrow 3.6}_{\pm 1.5}$	$28.0^{\downarrow 6.9}_{\pm 1.1}$	$42.5^{\uparrow 7.6}_{\pm 1.1}$	$15.0^{\downarrow 16.0}_{\pm 2.3}$	$33.0^{\uparrow 2.0}_{\pm 2.1}$	$12.4^{\downarrow 2.6}_{\pm 2.4}$	$15.6^{\uparrow 0.6}_{\pm 2.4}$
Science	$74.6^{\downarrow 7.4}_{\pm 1.8}$	$83.2^{\uparrow 1.2}_{\pm 1.6}$	$43.5^{\uparrow 8.6}_{\pm 1.2}$	$48.5^{\uparrow 13.6}_{\pm 1.1}$	$10.0^{\downarrow 21.0}_{\pm 2.3}$	$31.0^{0.0}_{\pm 2.3}$	$10.5^{\downarrow 4.5}_{\pm 2.1}$	$15.9^{\uparrow 0.9}_{\pm 2.3}$
Logic	$78.4^{\downarrow 3.6}_{\pm 1.7}$	$83.8^{\uparrow 1.8}_{\pm 1.5}$	$36.0^{\uparrow 1.1}_{\pm 1.1}$	$41.0^{\uparrow 6.1}_{\pm 1.1}$	$35.0^{\uparrow 4.0}_{\pm 2.3}$	$38.0^{\uparrow 7.0}_{\pm 2.3}$	$14.1^{\downarrow 0.9}_{\pm 2.5}$	$16.1^{\uparrow 1.1}_{\pm 2.1}$
Code	$79.6^{\downarrow 2.4}_{\pm 1.6}$	$81.6^{\downarrow 0.4}_{\pm 1.7}$	$38.4^{\uparrow 3.5}_{\pm 1.1}$	$38.7^{\uparrow 3.8}_{\pm 1.1}$	$31.0^{0.0}_{\pm 3.3}$	$34.0^{\uparrow 3.0}_{\pm 3.1}$	$15.8^{\uparrow 0.8}_{\pm 2.2}$	$17.8^{\uparrow 2.8}_{\pm 2.0}$

average gains of **24.9% and 12.6%**, respectively. The relatively smaller gain in mixed-data RL may stem from factors such as gradient imbalance across tasks (Wu et al., 2025b), which is not the key focus of our work. We focus on the more striking disparity: **why multi-stage SFT leads to performance collapse while RL sustains stable and significant growth?**

Takeaway 2.2

Multi-stage SFT leads to performance collapse, whereas RL enables stable and cumulative performance growth across distinct tasks.

2.3 Generalization of Single-Task Training

To investigate the underlying reason for the question above, we analyze the cross-task generalization after single-task training. As shown in Table 2, while SFT improves the target task by an average of 4.0%, it comes at the cost of degradation in untrained tasks, resulting in an average decline of 5.1% across other tasks. In contrast, RL achieves a superior average gain of 6.8% on the target task while simultaneously exerting a positive influence on others, yielding an average improvement of 2.3% on untrained tasks. These observations indicate that SFT and RL differ fundamentally in how they affect other capabilities during learning on a single task. This phenomenon, which we term **SFT Conflicts** and **RL Coexists**, directly determines their suitability for different paradigms in multi-task settings.

3 Parameter-Level Empirical Analysis

To investigate the mechanism underlying the contrasting generalization behaviors during single-task training observed in Section 2.3, we analyze the parameter update dynamics. For each task T_i, we compute the parameter update vector ΔW_i and evaluate two key geometric properties: the **magnitude** (L_2 norm) and the **pairwise cosine similarity** between tasks. As visualized in Figure 2, we identify two fundamental distinctions between SFT and RL updates.

Observation 1: The magnitude of RL updates is minimal and sparse. The heatmaps reveal a stark difference between ΔW_{RL} and ΔW_{SFT} in scale: the average L_2 norm of ΔW is approximately 3×10^{-2} for RL, whereas it reaches 7.4 for SFT, showing a difference of over two orders of magnitude. Furthermore, RL updates exhibit high sparsity; only about 20% of parameters in RL have magnitudes exceeding 10^{-5}, compared to 93% in SFT. These results align with recent findings (Mukherjee et al., 2025; Shenfeld et al., 2025) that RL tends to make **minimal adjustments**. Such sparse, small updates in high-dimensional parameter space can reduce interference across task-specific RL training.

Observation 2: RL optimization directions across tasks are approximately orthogonal. We observe that the pairwise cosine similarity of ΔW between different RL tasks is negligible, averaging around 10^{-5}. In contrast, SFT exhibits much higher cross-task similarity (on the order of 10^{-1} to 1.0), with some updates even pointing in opposite directions (e.g., Math vs. Code). Consequently, RL task updates ΔW_i are practically orthogonal, resulting in minimal interference with the optimization landscapes of other tasks. Whereas the ΔW_i from SFT exhibit overlapping optimization directions, which leads to significant mutual interference and the observed catastrophic forgetting.

Overall, this empirical analysis shows that RL induces **sparser** and **nearly orthogonal** parameter updates across tasks, which are two key factors that explain its minimal interference with other tasks.

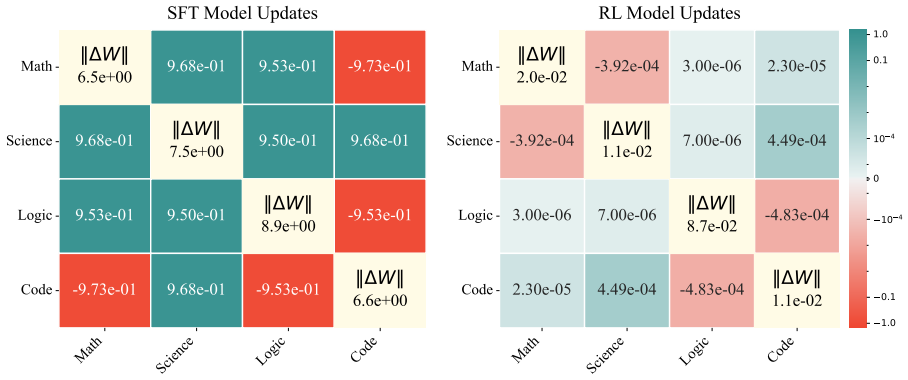

Figure 2: Analysis of Parameter Update Dynamics across SFT and RL. The heatmaps display the pairwise cosine similarity (off-diagonal) and the L_2 norm magnitude (diagonal) of parameter update vectors for different tasks. Left: SFT update has a larger absolute similarity value. (≈ 1.0). Right: RL updates show near-zero similarity, indicating orthogonal and task-specific parameter adjustments.

Takeaway 3

Unlike SFT, RL induces **sparse, minimal updates that are approximately orthogonal across tasks**, naturally decoupling the optimization of different tasks. This parametric mechanism fundamentally explains why RL supports robust multi-stage training.

4 Theoretical Analysis

Our empirical findings reveal two key properties of RL parameter updates which enable the coexistence of different task updates within a model: (1) **Sparsity and Small Magnitude**, where updates are confined to a small subset of parameters with diminishing norms compared to SFT; and (2) **Orthogonality**, where task-specific gradients occupy nearly disjoint subspaces. In this section, we will theoretically analyze these properties. To analyze the distinct parameter updates ΔW of SFT and RL, we examine their respective gradient descent directions. Let π_θ denote the policy parameterized by θ, π_{expert} be the supervision distribution. The expected gradients can be formalized as follows:

$$\begin{aligned} g_{\text{SFT}} &= \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_{\text{expert}}}\left[\nabla_\theta \log \pi_\theta(y|x)\right] \\ g_{\text{RL}} &= \mathbb{E}_{x\sim\mathcal{D},y\sim\pi_\theta}\left[A(x,y)\nabla_\theta \log \pi_\theta(y|x)\right] \end{aligned} \tag{1}$$

There are two fundamental distinctions: (1) Policy Source: SFT is off-policy, with the target response y sampled from a fixed expert distribution π_{expert}, whereas RL is on-policy, with y sampled from π_θ. (2) Advantage Function: The RL gradient is additionally weighted by a scalar advantage $A(x, y)$. In the following sections, We first qualitatively analyze how these two mechanisms shape gradient interference, and then quantify the key factors in §4.4.

4.1 Sparsity and Small Magnitude of RL Updates

Observation 1 of §3 highlights the conservative nature of RL updates. In the high-dimensional parameter space of LLMs, such sparse, small updates are likely to yield near-orthogonal directions across tasks, consistent with the principle that sparse high-dimensional vectors tend to be mutually orthogonal with high probability (Cai et al., 2013; Luisto, 2025). Recent works provide several explanations for this sparsity. In particular, RL's Razor (Shenfeld et al., 2025) shows that even without an explicit KL penalty, on-policy RL is implicitly biased toward solutions closest in KL divergence to the initial policy, which we restate as follows:

Proposition 4.1 (Convergence to KL-Minimal Solution, Shenfeld et al., 2025)**.** *Let $\mathcal{Y}$ be a finite set of outputs and let $\Pi \subseteq \Delta(\mathcal{Y})$ be a convex family of feasible policies. Let $R : \mathcal{Y} \to \{0, 1\}$ be a binary reward function and $\mathcal{P}^* = \{q : \mathbb{E}_q[R] = 1\}$ be the set of optimal policies. Then, under suitable regularity conditions, policy gradient converges to:*

$$\pi^{updated} = \arg\min_{\pi\in\mathcal{P}^*\cap\Pi} D_{KL}(\pi \,||\, \pi_0). \tag{2}$$

In other words, it selects the optimal policy closest in KL-divergence to the initialization π_0. This derivation shows that, under binary rewards, each RL update is equivalent to a two-stage alternating projection process: Information Projection (I-Projection) and Moment Projection (M-Projection), both implicitly minimizing KL divergence. These two stages correspond directly to the distinctions in Eq 1. I-Projection: Projects the current policy onto the set of all policies with $Reward = 1$, corresponding to reward function. M-Projection: Projects the intermediate policy derived from the I-Projection back to the policy space representable by the model's parameterization, reflecting the role of on-policy. In summary, unlike SFT which must bridge a potentially large gap to fit an extrinsic expert distribution, RL implicitly minimizes $D_{KL}(\pi^*||\pi_0)$. This restriction on deviation directly leads to the observed **sparsity** and **small magnitude** in parameter updates.

4.2 Qualitative Analysis of Multi-Task Gradients

In this section, we will analyze why gradients from different tasks in RL exhibit near-orthogonality. We also qualitatively examine this phenomenon from two perspectives: policy and advantage function. We consider two distinct tasks i and j with data distributions $\mathcal{D}_i$ and $\mathcal{D}_j$. Since recent works on reasoning enhancement use RL algorithms based on GRPO, we will use it as a representative.

Definition 4.2 (GRPO Gradient)**.** Given an input x from task i, we generate G group-wise rollouts $\{y_k\}_{k=1}^G$. The empirical gradient estimate is given by:

$$g_i(x) = \frac{1}{G}\sum_{k=1}^{G} \hat{A}_{i,k}(x)\nabla_\theta \log \pi_\theta(y_k|x), \tag{3}$$

where $\hat{A}_{i,k}(x)$ is the standardized advantage. Let $r_{i,k}(x)$ be the reward for the k-th sample, and let $\mu_{r_i}(x) = \frac{1}{G}\sum_{k=1}^{G} r_{i,k}(x)$ and $\sigma_{r_i}(x) = \sqrt{\frac{1}{G}\sum_{k=1}^{G}(r_{i,k}(x) - \mu_{r_i}(x))^2}$ be the group mean and standard deviation, respectively. The advantage is defined as $\hat{A}_{i,k}(x) = (r_{i,k} - \mu_{r_i}(x))/\sigma_{r_i}(x)$.

Definition of Score Function. For notational convenience, we will denote the log-likelihood gradient of the k-th rollout by the **score function** $S_{i,k}(x) = \nabla_\theta \log \pi_\theta(y_k|x)$ below.

According to the definition, the advantages for GRPO satisfy the **zero-sum property**.

$$\sum_{k=1}^{G} \hat{A}_{i,k}(x) = 0. \tag{4}$$

We contrast the inner products of gradient directions between tasks i and j in RL with those in SFT. We refer to this metric as the **gradient interference** term, which is formalized as follows:

Gradient Interference in SFT. Let π^* denote the expert policy (i.e., the supervision distribution). For a task i, the target response is $y_i^* \sim \pi^*(\cdot|x)$. Like RL, we define the **expert score function** as $S_i^*(x) \coloneqq \nabla_\theta \log \pi_\theta(y_i^*|x)$. Thus, the expected gradient interference in SFT is as follows:

$$\mathcal{I}_{\text{SFT}}(i,j) \coloneqq \mathbb{E}_{x,x'}\left[\langle S_i^*(x), S_j^*(x')\rangle\right]. \tag{5}$$

Gradient Interference in RL. For RL (GRPO), the expected inner product is given by:

$$\mathcal{I}_{\text{RL}}(i,j) \coloneqq \mathbb{E}_{x,x'}\left[\left\langle \frac{1}{G}\sum_{k=1}^{G}\hat{A}_{i,k}S_{i,k}(x), \frac{1}{G}\sum_{l=1}^{G}\hat{A}_{j,l}S_{j,l}(x')\right\rangle\right]. \tag{6}$$

Unlike SFT, we can decompose this raw form using the **zero-sum property** shown in Eq. (4).

Lemma 4.3 (Gradient Inner Product Decomposition)**.** *Let $\bar{S}_i(x) = \frac{1}{G}\sum_{k=1}^{G} S_{i,k}(x)$ be the mean score function for input x, and let $\delta S_{i,k}(x) = S_{i,k}(x) - \bar{S}_i(x)$ be the residual score function. The expected inner product between the gradients of task i and task j satisfies (proved in Appendix C):*

$$\mathcal{I}_{RL}(i,j) = \mathbb{E}_{x,x'}\left[\frac{1}{G^2}\sum_{k=1}^{G}\sum_{l=1}^{G}\hat{A}_{i,k}(x)\hat{A}_{j,l}(x')\langle \delta S_{i,k}(x), \delta S_{j,l}(x')\rangle\right], \tag{7}$$

Based on this derivations, we analyze the gradient interference in SFT and RL from two mechanisms.

Mechanism of Advantages. Based on Lemma 4.3, the **zero-sum property** of advantage function *algebraically removes the contribution of the mean gradient direction* $\bar{S}_i(x)$. This transforms the inner product of the score functions (i.e., log-likelihood gradients) $\langle S_{i,k}, S_{j,l}\rangle$ into the inner product of residuals $\langle \delta S_{i,k}(x), \delta S_{j,l}(x')\rangle$, diminishing the gradient interference in multi-task RL. Conversely, the SFT inner product in Eq. (5) depends directly on $\langle S_i^*(x), S_j^*(x')\rangle$. Consequently, the gradient interference in multi-task SFT is significantly larger than RL.

Mechanism of Policy Source. The second critical distinction is that SFT is **off-policy**, relying on an external expert distribution, whereas RL is **on-policy**.

First, according to Lemma 4.3, the RL inner product 7 is governed by residual score function, $\delta S_{i,k}(x)$. **Intuitively**, the residual $\delta S(x)$ captures the **intra-group policy divergence** across G rollouts for the same input x. Since both the input and the model parameters remain fixed during generation, this divergence is inherently limited. (We quantify the extent of this variation in §4.4.)

Second, for two distinct tasks i and j, the data distributions $\mathcal{D}_i$ and $\mathcal{D}_j$ are independent. Consequently, the small updates δS_i and δS_j from independent sampling processes can be viewed as **independent, zero-mean** vectors. In high-dimensional parameter space $\mathbb{R}^d$, concentration of measure Vershynin (2018) implies that such **independent, zero-mean, sparse** vectors are approximately orthogonal with high probability. Formally, the probability that their inner product deviates from zero decays exponentially with the dimension d: $\mathbb{P}\left(|\langle \delta S_i, \delta S_j\rangle| \geq t\right) \leq 2\exp(-ct^2 d)$ In contrast, off-policy SFT targets an external distribution, inducing **dense**, high-magnitude updates with greater cross-task **overlap**, and hence a larger interference term $\langle S_i^*(x), S_j^*(x')\rangle$. This intrinsic distinction between on-policy and off-policy directly accounts for the observed differences in gradient inner products. Quantitative analyses supporting are provided in §4.4.

Remark. The advantage function filters out the dense mean gradient $\bar{S}$, retaining only intra-group residuals, while on-policy optimization keeps these residuals δS small and thus less interfering.

4.3 Upper Bound of Gradient Orthogonality

We now derive an **upper bound** on the gradient interference in formula 5 and 7. This derivation allows us to identify the key factors contributing to the approximate orthogonality of RL observed in §3. To formally compare the interference magnitude, we introduce bounds characterizing the *absolute magnitude* of the score functions for SFT and the *intra-group variance* for RL.

Assumption 4.4 (Bounded Expected Norm and Variance)**.** For any task i with data distribution $\mathcal{D}_i$:

SFT Score Function Norm: The expected squared L_2 norm of the expert score function is bounded by a constant M_i^2. This characterizes the average energy of the updates:

$$\mathbb{E}_{x\sim\mathcal{D}_i}\left[\|S_i^*(x)\|_2^2\right] \leq M_i^2.$$

RL Score Function Variance: The expected intra-group variance of the residual score function (averaged over G rollouts) is bounded by a constant V_i^2:

$$\mathbb{E}_{x\sim\mathcal{D}_i}\left[\frac{1}{G}\sum_{k=1}^{G}\|\delta S_{i,k}(x)\|_2^2\right] \leq V_i^2.$$

Based on this assumption, we can give upper bounds for the gradient inner product for different tasks.

Theorem 4.5 (Upper Bound on Gradient Interference)**.** *Under Assumption 4.4, the expected gradient inner product for SFT and RL (GRPO) satisfies the following upper bounds:*

1. SFT Upper Bound (Norm-Limited):

$$|\mathcal{I}_{SFT}(i,j)| \leq M_i \cdot M_j. \tag{UB-SFT}$$

2. RL Upper Bound (Variance-Limited):

$$|\mathcal{I}_{RL}(i,j)| \leq V_i \cdot V_j. \tag{UB-RL}$$

Theorem 4.5 reveals a fundamental mechanistic difference between SFT and RL gradient interference upper bounds. The SFT bound depends on M_i, the absolute norm of the score function. As we have in our qualitative analysis in Section 4, the magnitude of SFT score function M_i is large. However, the situation is completely different for RL. The RL bound is controlled exclusively by V_i, the magnitude of the *residuals* δS. This is actually the variance with a group of rollouts generated by the model for the same input x. As we analyzed in Section 4.2, due to the on-policy nature of RL, the intra-group variation will be small. Furthermore, δS will continue to decay as training converges. The proof of this upper bound is provided in Appendix C.2. This theoretical distinction mirrors the fundamental essence of the two paradigms: SFT forces the model to fit fixed expert trajectories, whereas RL optimizes based on intra-group relative differences, reinforcing good trajectory while suppressing bad ones. We note that most properties also hold for other RL algorithms, as discussed in Appendix C.3.

Takeaway 4.3

Gradient interference in SFT is norm-limited, governed by the absolute magnitude of parameter updates. Conversely, RL interference is variance-limited, strictly bounded by the diversity of intra-group rollouts, which filters the common-mode conflicts.

4.4 Quantification and Visualization

To quantitatively validate theoretical analysis, we examine three key metrics related to score function S: the magnitude $||S||_2$, the residual δS (Lemma 4.3), and the cosine similarity. We follow the setup in §2.1, using 200 sampled instances per task. The comparative results are summarized in Table 3.

Verification of Interference Upper Bounds. We first evaluate the factors governing the gradient interference bounds derived in Theorem 4.5. For SFT, interference is *norm-limited*: Table 3 shows that $||S||_2$ is substantial (~ 7.1), indicating that fitting external experts can induce large gradient interference. In contrast, RL interference is *variance-limited*, governed by the intra-group residual δS. Empirically, $||\delta S||_2$ is observed to be small ($\sim 10^{-2}$). This confirms that filtering effect of the advantage function restricts interference to this minimal variance term, explaining the substantially lower interference in RL.

Table 3: Quantitative comparison of score function (S) indicators between SFT and RL. RL exhibits significantly smaller magnitude and similarity.

Indicator	SFT	RL
$\|\|S\|\|_2$	~ 7.1	$\sim 10^{-1}$
$\|\|\delta S\|\|_2$	–	$\sim 10^{-2}$
$\text{CosSim}(S_i, S_j)$	$\sim 10^{-1}$	$\sim 10^{-3}$

Verification of Task Separation. Next, we validate the mechanism discussed in Section 4.2, which posits that on-policy RL naturally leads to low overlap between updates from different tasks, while SFT updates show significant overlap. Table 3 reveals that the cosine similarity for RL ($\sim 10^{-3}$) is much lower than that of SFT ($\sim 10^{-1}$), indicating that RL updates occupy nearly orthogonal subspaces. We also employ t-SNE to plot the distribution of score functions. As shown in Figure 3, RL score functions form distinct, separable clusters for different tasks. Conversely, SFT distributions exhibit heavy overlap. Together, these results corroborate that SFT suffers from dense, conflicting updates, whereas RL naturally decouples task optimization. More visualizations are shown in Appendix E.3.

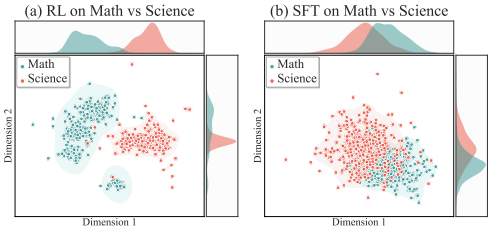

Figure 3: Distribution of the score function S sampled during training by RL and SFT on different tasks (Math and Science). We use t-SNE to reduce the dimensionality for visualization.

5 Application and Discussion: Parallel-RL Framework

5.1 Methodology

Our empirical and theoretical analyses reveal a key geometric property of multi-task RL: parameter updates for distinct reasoning tasks occupy **approximately orthogonal** subspaces (Section 4). This finding implies that the interference term $\langle \Delta W_i, \Delta W_j \rangle$ is negligible. Considering optimization

Table 4: The accuracy (%) on different tasks. Superscripts denote accuracy change vs. **corresponding single-task baseline**. Δ Base shows the average performance improvement over the Base Model. Retention (%) represents the **percentage of performance retained** compared to the Single-Task Training Model before merging for Parallel Training method. **Bold** represents the best performance.

Base Model	Method	Math		Science		Logic	Code	Average Statistic	
		MATH500	AIME2025	MMLU	GPQA	KK	LiveCodeBench	Δ Base	Retention (%)
	Baselines (Full Parameter GRPO)								
DeepSeek-R1-Distill-Qwen-1.5B	Base Model	82.0	26.9	34.9	32.3	31.0	15.0	-	-
	Single-Task SFT	85.8	31.5	49.1	35.4	37.0	16.3	+5.5	-
	Single-Task RL	87.4	32.7	51.8	39.9	44.0	21.6	+9.3	-
	Previous Multi Task Training Paradigms								
	Mixed Data SFT	$85.6^{-0.2}$	$30.1^{-1.4}$	$48.5^{-0.6}$	$34.3^{-1.1}$	$38.0^{+1.0}$	$17.0^{+0.7}$	+5.2	-
	Multi Stage SFT	$71.0^{-14.8}$	$21.8^{-9.7}$	$33.2^{-15.9}$	$23.2^{-12.2}$	$11.0^{-26.0}$	$12.1^{-4.2}$	-8.3	-
	Mixed Data RL	$87.0^{-0.4}$	$32.9^{+0.2}$	$50.9^{-0.9}$	$38.9^{-1.0}$	$45.0^{+1.0}$	$20.7^{-0.9}$	+8.9	-
	Multi Stage RL	$87.8^{+0.4}$	$33.1^{+0.4}$	$\mathbf{52.8}^{+1.0}$	$40.4^{+0.5}$	$48.0^{+4.0}$	$21.0^{-0.6}$	+10.2	-
	Parallel Training								
	Naive Parallel-SFT *(mean)*	$63.2^{-22.6}$	$19.2^{-12.3}$	$26.9^{-22.2}$	$25.3^{-10.1}$	$26.0^{-11.0}$	$9.3^{-7.0}$	-8.7	66.6
	Naive Parallel-SFT *(sum)*	$61.2^{-24.6}$	$20.0^{-11.5}$	$25.9^{-23.2}$	$26.8^{-8.6}$	$23.0^{-14.0}$	$10.1^{-6.2}$	-9.2	65.4
	Naive Parallel-RL *(mean)*	$85.6^{-1.8}$	$31.5^{-1.2}$	$47.3^{-4.5}$	$36.4^{-3.5}$	$39.0^{-5.0}$	$19.0^{-2.6}$	+6.1	93.3
	Naive Parallel-RL *(sum)*	$86.4^{-1.0}$	$32.3^{-0.4}$	$48.0^{-3.8}$	$37.4^{-2.5}$	$39.0^{-5.0}$	$18.4^{-3.2}$	+6.6	94.2
	TIES Parallel-RL	$87.6^{+0.2}$	$32.5^{-0.2}$	$49.2^{-2.6}$	$38.4^{-1.5}$	$43.0^{-1.0}$	$19.7^{-1.9}$	+8.0	97.4
	SVD Parallel-RL	$87.2^{-0.2}$	$31.9^{-0.8}$	$48.3^{-3.5}$	$36.9^{-3.0}$	$42.0^{-2.0}$	$15.9^{-5.7}$	+6.7	94.6
	Adapted Parallel-RL	$\mathbf{88.6}^{+1.2}$	$\mathbf{33.8}^{+1.1}$	$50.9^{-0.9}$	$\mathbf{41.4}^{+1.5}$	$\mathbf{49.0}^{+5.0}$	$\mathbf{22.5}^{+0.9}$	+10.7	**103.2**
	Baselines (Full Parameter GRPO)								
DeepSeek-R1-Distill-Qwen-7B	Base Model	91.8	49.2	51.8	48.5	62.0	36.5	-	-
	Single-Task SFT	93.8	54.4	59.1	51.0	65.0	39.1	+3.8	-
	Single-Task RL	94.6	57.1	63.5	53.0	68.0	42.3	+6.5	-
	Previous Multi Task Training Paradigms								
	Mixed Data SFT	$94.6^{+0.8}$	$53.1^{-1.3}$	$61.7^{+2.6}$	$52.5^{+1.5}$	$66.0^{+1.0}$	$40.0^{+0.9}$	+4.7	-
	Multi Stage SFT	$81.4^{-12.4}$	$38.6^{-15.8}$	$35.3^{-23.8}$	$39.1^{-11.9}$	$46.0^{-19.0}$	$29.5^{-9.6}$	-11.7	-
	Mixed Data RL	$95.0^{+0.4}$	$56.5^{-0.6}$	$62.2^{-1.3}$	$52.8^{-0.2}$	$67.0^{-1.0}$	$39.9^{-2.4}$	+5.6	-
	Multi Stage RL	$95.0^{+0.4}$	$58.2^{+1.1}$	$65.9^{+2.4}$	$\mathbf{55.6}^{+2.6}$	$66.0^{-2.0}$	$\mathbf{44.2}^{+1.9}$	+7.5	-
	Parallel Training								
	Naive Parallel-SFT *(mean)*	$61.0^{-32.8}$	$35.8^{-18.6}$	$33.1^{-26.0}$	$38.9^{-12.1}$	$48.0^{-17.0}$	$27.0^{-12.1}$	-16.0	67.3
	Naive Parallel-SFT *(sum)*	$59.2^{-34.6}$	$35.6^{-18.8}$	$32.4^{-26.7}$	$37.8^{-13.2}$	$50.0^{-15.0}$	$26.5^{-12.6}$	-16.4	66.6
	Naive Parallel-RL *(mean)*	$93.2^{-1.4}$	$52.3^{-4.8}$	$59.0^{-4.5}$	$51.0^{-2.0}$	$61.0^{-7.0}$	$38.3^{-4.0}$	+2.5	93.7
	Naive Parallel-RL *(sum)*	$93.6^{-1.0}$	$53.1^{-4.0}$	$59.6^{-3.9}$	$52.5^{-0.5}$	$63.0^{-5.0}$	$38.0^{-4.3}$	+3.3	95.1
	TIES Parallel-RL	$94.8^{+0.2}$	$56.5^{-0.6}$	$62.5^{-1.0}$	$52.0^{-1.0}$	$64.0^{-4.0}$	$40.5^{-1.8}$	+5.1	97.8
	SVD Parallel-RL	$93.2^{-1.4}$	$55.2^{-1.9}$	$61.4^{-2.1}$	$51.5^{-1.5}$	$65.0^{-3.0}$	$39.5^{-2.8}$	+4.3	96.7
	Adapted Parallel-RL	$\mathbf{96.2}^{+1.6}$	$\mathbf{59.0}^{+1.9}$	$\mathbf{66.8}^{+3.3}$	$54.0^{+1.0}$	$\mathbf{70.0}^{+2.0}$	$41.5^{-0.8}$	+8.0	**102.4**

directions are irrelevant, the sum of parameter updates obtained from parallel training of different tasks should theoretically approximate the result of multi-stage training of tasks.

Motivated by this insight, we preliminarily explore **Parallel-RL**, a decoupled post-training framework that optimizes tasks independently. Formally, let $\mathcal{T} = \{T_1, \ldots, T_N\}$ be a set of diverse tasks. We launch N parallel RL processes. Each process i produces a task-specific update ΔW_i. The final model W_{final} is obtained by merging these independent updates: $W_{final} = W_{base} + \mathcal{M}(\Delta W_1, \ldots, \Delta W_N)$. Here, $\mathcal{M}$ is a merging function (e.g., sum or SVD). We emphasize that Parallel-RL is not merely a model merging technique, but a complete post-training framework. Beyond merging stage, it also involves determining which tasks can be trained in parallel, as well as adopting appropriate tricks for each single task training, which are preliminarily discussed in Appendix D.

5.2 Main Experiment

Setup. We adopt DeepSeek-R1-Distill-Qwen-1.5B and 7B as the base models for full parameter training. The configuration follows the settings described in Appendix A. Here we choose GRPO as the RL algorithm. Results for other RL algorithms can be found in Appendix E.2.

Parallel-RL Method. We consider three implementations. **(1) Naive Parallel-RL.**We employ two strategies: (a) *sum*: directly summing the ΔW_i, (b) *mean*: averaging the ΔW_i. **(2) Sparse Parallel.** We consider two sparsification strategies. First, we adopt the standard TIES (Yadav et al., 2023). Second, motivated by evidence that the rank-1 component of ΔW_i encapsulates most RL effects (Cai et al., 2025), we perform SVD on updates and retain only rank-1 directions. **(3) Adapted Parallel-RL.** Since different RL models enhance the sampling probability of high-reward trajectory for different tasks, the merged model can benefit from light adaptation training. After Naive Parallel RL *(sum)*, we perform adaptation using a subset of samples (5% of original data) to refine the model.

5.3 Main Results

As shown in Table 4, we observe that even Naive Parallel-RL *(sum)* allows different task updates (ΔW_i) to coexist, retaining 95% of single-task RL performance and achieving an average improvement of 5.0% over base model. In contrast, Parallel-SFT (*sum*) retains only 66% of single-task SFT performance due to stronger task conflicts. Furthermore, incorporating sparsification strategies further

enhances the compatibility of various ΔW_i for RL. TIES and SVD Parallel-RL respectively retained 98% and 96% of the performance of the corresponding single-task RL. Notably, Adapted Parallel-RL achieves the best performance across most tasks. It delivers a 9.4% gain over the base model and even surpasses individual task-specific models, retaining an average of 102.8% of the Single-Task RL performance. Despite these gains, this paradigm only needs an additional 5% samples adaptation time compared to single-task training, demonstrating both the **efficiency and superior efficacy** of our approach. More experimental results are available in Appendix E.1. We also performed Pass@K analysis to further verify the stability of Parallel RL, as shown in Appendix E.4.

5.4 Ablation Study

To assess the orthogonality and decoupling of task-specific representations, we conduct an ablation study by selectively excluding each task-specific update (ΔW_i) during the merging process. As reported in Table 5, removing a specific ΔW_i substantially degrades its corresponding task, with an average drop Δ_{Target} of 7.1%, while performance on other tasks remains stable and even improves slightly on average ($\Delta_{\text{Others}} = +0.6\%$). This result demonstrates that Parallel-RL effectively decouples capabilities across tasks. Unlike multi-stage or mixed-data RL, this paradigm allows for the flexible composition of models, enabling modular adaptation to diverse task scenarios. More ablation results are shown in Appendix E.2

Table 5: Performance (%) ablation of Naive Parallel-RL. Δ_{Target} and Δ_{Others} denote accuracy changes on the target task and other tasks after removing each component.

Method	MATH500	MMLU	KK	LCB	Δ_{Target}	Δ_{Others}
Baselines						
Single-Task RL	87.4	51.8	44.0	21.6	–	–
Base Model	82.0	34.9	31.0	15.0	–	–
Ablation of Parallel-RL (1.5B GRPO)						
Parallel-RL	86.4	48.0	39.0	18.4	–	–
w/o Math	82.8	47.8	41.0	19.2	-3.6	+0.9
w/o Science	86.0	37.5	39.0	18.4	-10.5	-0.1
w/o Logic	85.6	49.6	30.0	20.3	-9.0	+0.9
w/o Code	88.2	46.9	40.0	13.1	-5.3	+0.6

6 Related Work

We summarize work most relevant to our study and provide a broader discussion in Appendix B. Recently, Enhancing LLM reasoning has become a major research focus (Guo et al., 2025; Jaech et al., 2024), with SFT (Xu et al., 2025; Hugging Face, 2025) and RL (Zheng et al., 2025; Ouyang et al., 2022) as the dominant post-training paradigms. Recent studies distinguish their mechanisms in *single-task* settings. Chu et al. (2025) argue that SFT memorizes, RL generalizes, suggesting SFT focuses on learning new paths, while RL enhances generalization. Similar works include (Matsutani et al., 2025; Jin et al., 2025a), etc. Mukherjee et al. (2025) further shows that RL updates only a small subnetwork in contrast to SFT. Other works have explored the combination of these two paradigms (Zhang et al., 2025; Fu et al., 2025). Multi-task learning is also essential for scaling towards AGI (Zeng et al., 2025a; Li et al., 2025a). However, the divergence between multi-task SFT and RL remains underexplored. Some works have explored related aspects. RL's Razor Shenfeld et al. (2025) theoretically explains that RL minimizes parameter deviation due to implicit KL constraints, thereby reducing forgetting. Similar analysis can be found in The Path Not Taken Zhu et al. (2025). However, their analysis targets forgetting rather than multi-task learning and the fundamental differences in the *training paradigms*. In contrast, our work is the first to systematically investigate the divergence between SFT and RL in multi-task training with theoretical and empirical analysis.

7 Conclusion

This work systematically analyzes the divergence between SFT and RL in multi-task training. We observe that in multi-stage training, SFT exhibits significant task conflicts, whereas RL demonstrates a coexistence property, allowing for stable improvements. Through mechanistic analysis, we find that the optimization directions of RL across different tasks share significantly lower similarity than SFT, appearing approximately orthogonal. We further substantiate these empirical findings with a in-depth theoretical analysis, elucidating how the advantage function and on-policy nature of RL drive this mechanism. Building on the findings, we introduce Parallel-RL to decouple multi-task training, enabling efficient, modular training that achieves superior performance with minimal adaptation.

AI USE STATEMENT

During the writing of this paper, we used LLMs to assist with grammar checking and text polishing. All AI-assisted revisions were carefully reviewed by the authors to ensure that they did not alter the original meaning, technical content, or conclusions. We also used AI assistants to improve the presentation of some tables and figures. All experimental data and numerical results were independently verified by the authors, and no experimental results were modified through AI assistance.

ETHICS STATEMENT

This paper aims to advance the development of multi-task learning for LLMs. By analyzing and improving the efficiency and effectiveness of model learning across diverse task scenarios, we hope to facilitate progress toward the vision of AGI. However, we emphasize that while striving for efficient multi-task training, it is also crucial to maintain the safety and reliability of each individual task to ensure responsible deployment. Finally, we believe that this methodological study does not introduce distinct ethical risks or societal impacts that warrant detailed discussion here.

REPRODUCIBILITY STATEMENT

We provide the implementation details required to reproduce our main results in both the main text and appendix. §2.1 summarizes the overall experimental setup, including the base models, tasks, and training algorithms. More complete reproducibility details are provided in the appendix. Specifically, Appendix A describes the training and evaluation datasets, base models, computational resources, and hyperparameter settings, while Appendix G provides the prompt templates used for training and evaluation. Together, these details are intended to improve the reproducibility of our work.

REFERENCES

Tony Cai, Jianqing Fan, and Tiefeng Jiang. Distributions of angles in random packing on spheres, 2013. URL https://arxiv.org/abs/1306.0256.

Yuchen Cai, Ding Cao, Xin Xu, Zijun Yao, Yuqing Huang, Zhenyu Tan, Benyi Zhang, Guiquan Liu, and Junfeng Fang. On predictability of reinforcement learning dynamics for large language models. *arXiv preprint arXiv:2510.00553*, 2025.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Zhoujun Cheng, Shibo Hao, Tianyang Liu, Fan Zhou, Yutao Xie, Feng Yao, Yuexin Bian, Yonghao Zhuang, Nilabjo Dey, Yuheng Zha, et al. Revisiting reinforcement learning for llm reasoning from a cross-domain perspective. *arXiv preprint arXiv:2506.14965*, 2025.

Myungsik Cho, Jongeui Park, Suyoung Lee, and Youngchul Sung. Hard tasks first: Multi-task reinforcement learning through task scheduling. In *Forty-first International Conference on Machine Learning*, 2024.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*, 2025.

Guanting Dong, Hongyi Yuan, Keming Lu, Chengpeng Li, Mingfeng Xue, Dayiheng Liu, Wei Wang, Zheng Yuan, Chang Zhou, and Jingren Zhou. How abilities in large language models are affected by supervised fine-tuning data composition. *arXiv preprint arXiv:2310.05492*, 2023.

Yuqian Fu, Tinghong Chen, Jiajun Chai, Xihuai Wang, Songjun Tu, Guojun Yin, Wei Lin, Qichao Zhang, Yuanheng Zhu, and Dongbin Zhao. Srft: A single-stage method with supervised and reinforcement fine-tuning for reasoning. *arXiv preprint arXiv:2506.19767*, 2025.

Google. Introducing gemini 3, 2025. URL https://blog.google/products/gemini/gemini-3-collection/.

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

Nathan Habib, Clémentine Fourrier, Hynek Kydlíček, Thomas Wolf, and Lewis Tunstall. Lighteval: A lightweight framework for llm evaluation, 2023. URL https://github.com/huggingface/lighteval.

Yupu Hao, Zhuoran Jin, Huanxuan Liao, Kang Liu, and Jun Zhao. Why multi-step tool-use reinforcement learning collapses and how supervisory signals fix it, 2026. URL https://arxiv.org/abs/2606.26027.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021. URL https://arxiv.org/abs/2009.03300.

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, 2021. URL https://arxiv.org/abs/2106.09685.

Maggie Huan, Yuetai Li, Tuney Zheng, Xiaoyu Xu, Seungone Kim, Minxin Du, Radha Poovendran, Graham Neubig, and Xiang Yue. Does math reasoning improve general llm capabilities? understanding transferability of llm reasoning. *arXiv preprint arXiv:2507.00432*, 2025.

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

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. *arXiv preprint arXiv:2212.04089*, 2022.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

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, 2024. URL https://arxiv.org/abs/2403.07974.

Hangzhan Jin, Sicheng Lv, Sifan Wu, and Mohammad Hamdaqa. Rl is neither a panacea nor a mirage: Understanding supervised vs. reinforcement learning fine-tuning for llms. *arXiv preprint arXiv:2508.16546*, 2025a.

Xisen Jin, Xiang Ren, Daniel Preotiuc-Pietro, and Pengxiang Cheng. Dataless knowledge fusion by merging weights of language models. *arXiv preprint arXiv:2212.09849*, 2022.

Zhuoran Jin, Hongbang Yuan, Kejian Zhu, Jiachun Li, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. Omni-reward: Towards generalist omni-modal reward modeling with free-form preferences. *arXiv preprint arXiv:2510.23451*, 2025b.

Zhuoran Jin, Kejian Zhu, Hongbang Yuan, Yupu Hao, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. Look light, think heavy: What multimodal chain-of-thought reasoning can and cannot do, 2026. URL https://arxiv.org/abs/2606.22565.

Song Lai, Haohan Zhao, Rong Feng, Changyi Ma, Wenzhuo Liu, Hongbo Zhao, Xi Lin, Dong Yi, Min Xie, Qingfu Zhang, Hongbin Liu, Gaofeng Meng, and Fei Zhu. Reinforcement fine-tuning naturally mitigates forgetting in continual post-training, 2025. URL https://arxiv.org/abs/2507.05386.

Derek Li, Jiaming Zhou, Leo Maxime Brunswic, Abbas Ghaddar, Qianyi Sun, Liheng Ma, Yu Luo, Dong Li, Mark Coates, Jianye Hao, et al. Omni-thinker: Scaling multi-task rl in llms with hybrid reward and task scheduling. *arXiv preprint arXiv:2507.14783*, 2025a.

Jiachun Li, Pengfei Cao, Zhuoran Jin, Yubo Chen, Jiexin Xu, Huaijun Li, Xiaojian Jiang, Kang Liu, and Jun Zhao. Fixing the broken compass: Diagnosing and improving inference-time reward modeling, 2026a. URL https://arxiv.org/abs/2503.05188.

Jiachun Li, Shaoping Huang, Zhuoran Jin, Chenlong Zhang, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. Mmr-life: Piecing together real-life scenes for multimodal multi-image reasoning, 2026b. URL https://arxiv.org/abs/2603.02024.

Jiachun Li, Zhuoran Jin, Tianyi Men, Yupu Hao, Kejian Zhu, Lingshuai Wang, Dongqi Huang, Longxiang Wang, Shengjia Hua, Lu Wang, Jinshan Gao, Hongbang Yuan, Ruilin Xu, Kang Liu, and Jun Zhao. Agentic environment engineering for large language models: A survey of environment modeling, synthesis, evaluation, and application, 2026c. URL https://arxiv.org/abs/2606.12191.

Xiaomin Li, Zhou Yu, Zhiwei Zhang, Xupeng Chen, Ziji Zhang, Yingying Zhuang, Narayanan Sadagopan, and Anurag Beniwal. When thinking fails: The pitfalls of reasoning for instruction-following in llms. *arXiv preprint arXiv:2505.11423*, 2025b.

Yu Li, Zhuoshi Pan, Honglin Lin, Mengyuan Sun, Conghui He, and Lijun Wu. Can one domain help others? a data-centric study on multi-domain reasoning via reinforcement learning. *arXiv preprint arXiv:2507.17512*, 2025c.

Xize Liang, Lin Yang, Jie Wang, Yiyang Lu, Runyu Wu, Hanzhu Chen, and Jianye Hao. Boosting multi-domain fine-tuning of large language models through evolving interactions between samples. In *Forty-second International Conference on Machine Learning*, 2025.

Xize Liang, Lin Yang, Jie Wang, Rui Liu, Yang Lu, Jinliang Zeng, Hanzhu Chen, Dong Li, and Jianye Hao. Boosting multi-domain reasoning of llms via curvature-guided policy optimization. In *The Fourteenth International Conference on Learning Representations*, 2026.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*, 2025a.

Junteng Liu, Yuanxiang Fan, Zhuo Jiang, Han Ding, Yongyi Hu, Chi Zhang, Yiqi Shi, Shitong Weng, Aili Chen, Shiqi Chen, et al. Synlogic: Synthesizing verifiable reasoning data at scale for learning logical reasoning and beyond. *arXiv preprint arXiv:2505.19641*, 2025b.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, et al. The flan collection: Designing data and methods for effective instruction tuning. In *International Conference on Machine Learning*, pp. 22631–22648. PMLR, 2023.

Rami Luisto. A short survey on almost orthogonal vectors in a few specific large dimensions, 2025. URL https://arxiv.org/abs/2510.23609.

Michael Luo, Sijun Tan, Roy Huang, Ameen Patel, Alpay Ariyak, Qingyang Wu, Xiaoxiang Shi, Rachel Xin, Colin Cai, Maurice Weber, et al. Deepcoder: A fully open-source 14b coder at o3-mini level. *Notion Blog*, 2025.

Michael S Matena and Colin A Raffel. Merging models with fisher-weighted averaging. *Advances in Neural Information Processing Systems*, 35:17703–17716, 2022.

Kohsei Matsutani, Shota Takashiro, Gouki Minegishi, Takeshi Kojima, Yusuke Iwasawa, and Yutaka Matsuo. Rl squeezes, sft expands: A comparative study of reasoning llms. *arXiv preprint arXiv:2509.21128*, 2025.

Tianyi Men, Zhuoran Jin, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. Agent-rewardbench: Towards a unified benchmark for reward modeling across perception, planning, and safety in real-world multimodal agents, 2025. URL https://arxiv.org/abs/2506.21252.

Tianyi Men, Zhuoran Jin, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. Empowering gui agents via autonomous experience exploration and hindsight experience utilization for task planning, 2026a. URL https://arxiv.org/abs/2606.27330.

Tianyi Men, Zhuoran Jin, Kang Liu, and Jun Zhao. The physics of multi-turn long-horizon planning: From pre-training to post-training via single- and multi-teacher on-policy agentic distillation, 2026b. URL https://arxiv.org/abs/2607.24720.

Chunyang Meng, Shijie Song, Haogang Tong, Maolin Pan, and Yang Yu. Deepscaler: Holistic autoscaling for microservices based on spatiotemporal gnn with adaptive graph learning. In *2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pp. 53–65. IEEE, 2023.

Sagnik Mukherjee, Lifan Yuan, Dilek Hakkani-Tur, and Hao Peng. Reinforcement learning finetunes small subnetworks in large language models. *arXiv preprint arXiv:2505.11711*, 2025.

Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research*, 21(181):1–50, 2020.

OpenAI. Gpt-5 is here, 2025. URL https://openai.com/gpt-5/.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Simon Park. Instruct-skillmix: A powerful pipeline for llm instruction tuning. Master's thesis, Princeton University, 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, 2023. URL https://arxiv.org/abs/2311.12022.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Idan Shenfeld, Jyothish Pari, and Pulkit Agrawal. Rl's razor: Why online reinforcement learning forgets less. *arXiv preprint arXiv:2509.04259*, 2025.

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

Yuqiao Tan, Minzheng Wang, Shizhu He, Huanxuan Liao, Chengfeng Zhao, Qiunan Lu, Tian Liang, Jun Zhao, and Kang Liu. Bottom-up policy optimization: Your language model policy secretly contains internal policies, 2026a. URL https://arxiv.org/abs/2512.19673.

Yuqiao Tan, Minzheng Wang, Bo Liu, Zichen Liu, Tian Liang, Shizhu He, Jun Zhao, and Kang Liu. From $p(y|x)$ to $p(y)$: Investigating reinforcement learning in pre-train space, 2026b. URL https://arxiv.org/abs/2604.14142.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

Xiaoyu Tian, Yunjie Ji, Haotian Wang, Shuaiting Chen, Sitong Zhao, Yiping Peng, Han Zhao, and Xiangang Li. Not all correct answers are equal: Why your distillation source matters, 2025a. URL https://arxiv.org/abs/2505.14464.

Xiaoyu Tian, Sitong Zhao, Haotian Wang, Shuaiting Chen, Yiping Peng, Yunjie Ji, Han Zhao, and Xiangang Li. Deepdistill: Enhancing llm reasoning capabilities via large-scale difficulty-graded data training. *arXiv preprint arXiv:2504.17565*, 2025b.

Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pp. 23965–23998. PMLR, 2022.

Minghao Wu, Thuy Vu, Lizhen Qu, and Reza Haf. Mixture-of-skills: Learning to optimize data usage for fine-tuning large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 14226–14240, 2024.

Runzhe Wu, Ankur Samanta, Ayush Jain, Scott Fujimoto, Jeongyeol Kwon, Ben Kretzu, Youliang Yu, Kaveh Hassani, Boris Vidolov, and Yonathan Efroni. Imbalanced gradients in rl post-training of multi-task llms. *arXiv preprint arXiv:2510.19178*, 2025a.

Runzhe Wu, Ankur Samanta, Ayush Jain, Scott Fujimoto, Jeongyeol Kwon, Ben Kretzu, Youliang Yu, Kaveh Hassani, Boris Vidolov, and Yonathan Efroni. Imbalanced gradients in rl post-training of multi-task llms, 2025b. URL https://arxiv.org/abs/2510.19178.

Chulin Xie, Yangsibo Huang, Chiyuan Zhang, Da Yu, Xinyun Chen, Bill Yuchen Lin, Bo Li, Badih Ghazi, and Ravi Kumar. On memorization of large language models in logical reasoning. 2024. URL https://arxiv.org/abs/2410.23123.

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

Haotian Xu, Xing Wu, Weinong Wang, Zhongzhi Li, Da Zheng, Boyuan Chen, Yi Hu, Shijia Kang, Jiaming Ji, Yingying Zhang, et al. Redstar: Does scaling long-cot data unlock better slow-reasoning systems? *arXiv preprint arXiv:2501.11284*, 2025.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36:7093–7115, 2023.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch. In *Forty-first International Conference on Machine Learning*, 2024.

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.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model?, 2025. URL https://arxiv.org/abs/2504.13837.

Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, et al. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*, 2025a.

Zhiyuan Zeng, Hamish Ivison, Yiping Wang, Lifan Yuan, Shuyue Stella Li, Zhuorui Ye, Siting Li, Jacqueline He, Runlong Zhou, Tong Chen, et al. Rlve: Scaling up reinforcement learning for language models with adaptive verifiable environments. *arXiv preprint arXiv:2511.07317*, 2025b.

Wenhao Zhang, Yuexiang Xie, Yuchang Sun, Yanxi Chen, Guoyin Wang, Yaliang Li, Bolin Ding, and Jingren Zhou. On-policy rl meets off-policy experts: Harmonizing supervised fine-tuning and reinforcement learning via dynamic weighting. *arXiv preprint arXiv:2508.11408*, 2025.

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.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. *arXiv preprint arXiv:2403.13372*, 2024.

Hanqing Zhu, Zhenyu Zhang, Hanxian Huang, DiJia Su, Zechun Liu, Jiawei Zhao, Igor Fedorov, Hamed Pirsiavash, Zhizhou Sha, Jinwon Lee, et al. The path not taken: Rlvr provably learns off the principals. *arXiv preprint arXiv:2511.08567*, 2025.

Kejian Zhu, Zhuoran Jin, Hongbang Yuan, Jiachun Li, Shangqing Tu, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. Mmr-v: What's left unsaid? a benchmark for multimodal deep reasoning in videos, 2026. URL https://arxiv.org/abs/2506.04141.

Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

A Experiment Details

A.1 Task and Dataset Description

In this study, we select four representative tasks widely utilized in reasoning enhancement research for multi-task learning: Math, Science, Code, and Logic. In the following sections, we detail the training and evaluation datasets employed for each of these tasks.

Training Datasets. We use different datasets that are more suitable for SFT and RL training, respectively. **For SFT**, we utilize a subset of *OpenR1-Math-220k (Hugging Face, 2025)* as the math training set, a subset of *AM-DeepSeek-Distilled-40M (Tian et al., 2025b)* for code generation, and a subset of *AM-Thinking-v1-Distilled (Tian et al., 2025a)* for science. For the logical reasoning task, we employ the *knights-and-knaves (Xie et al., 2024)* dataset. Since the original data lacks CoTs, we leverage the DeepSeek-R1 API to distill LongCoT data and ensure the correctness of the distilled answers using the rule-based verification logic from *Logic-RL (Xie et al., 2025)*. **Regarding RL**, our training data consists of a subset of the *DeepScaleR-Preview-Dataset (Meng et al., 2023)* for math and a subset of the *DeepCoder-Preview-Dataset (Luo et al., 2025)* for code. For science, we use

a subset of the science portion from *AM-Thinking-v1-Distilled*. Finally, following the protocol of *Logic-RL*, a subset of the *knights-and-knaves* dataset is used for the logical task.

Evaluation Benchmarks. We evaluate our models using six mainstream and widely recognized benchmarks. For Mathematics, we select *MATH500 (Lightman et al., 2023)* and *AIME2025*. It is worth noting that the number of questions for AIME2025 is relatively small. To avoid randomness, we report the evaluation metric as avg@16, which is the average accuracy of 16 tests on the dataset. For Science, we employ *MMLU (Hendrycks et al., 2021)* and *GPQA-Diamond (Rein et al., 2023)*. Considering the massive scale of MMLU and the fact that some of its tasks are not directly related to scientific reasoning, we specifically focus on the following subjects: *high_school_physics, high_school_chemistry, college_chemistry, college_biology, astronomy,* and *professional_medicine*. For Code Generation, we primarily use *LiveCodeBench (Jain et al., 2024)* for evaluation. Lastly, for Logical Reasoning, the *knights-and-knaves* dataset is used as the evaluation benchmark.

A.2 Training Details

We employ *DeepSeek-R1-Distill-Qwen-1.5B* and *7B* as base models. All experiments are conducted on a machine of eight NVIDIA A100 (80GB) GPUs. To facilitate efficient parameter analysis, some experiments are performed using LoRA (Hu et al., 2021). For all LoRA experiments, we set the rank to $r = 64$ and the scaling factor to $\alpha = 32$. For multi-stage training, we chose the following task order: Math, Science, Code, Logic. For TIES merging (Yadav et al., 2023), we follow the default settings and set the sensitivity parameter to 0.8.

SFT. SFT is implemented using the *LLaMA-Factory (Zheng et al., 2024)* framework. We fine-tune the models on teacher CoT trajectories. The learning rate is set to 1×10^{-5}, and the sequence cutoff length is configured at 8K tokens.

RL. We utilize the *VeRL (Sheng et al., 2024)* framework for reinforcement learning training, specifically employing the GRPO algorithm. The RL learning rate is fixed at 3×10^{-6}. Following the official DeepSeek configuration, we set the sampling temperature to 0.6 and *top-p* to 0.95. For each prompt, we generate $G = 16$ rollouts to perform group-based relative reward estimation. The maximum output length is set to 8K tokens. Notably, we disable the KL divergence penalty in the loss function for our specific analysis.

We acknowledge the exceptional contributions of *LLaMA-Factory* and *VeRL* teams. Their extensible and user-friendly frameworks, which support diverse architectures and advanced algorithms, significantly facilitated our training implementation.

A.3 Evaluation Details

For the majority of the tasks, we conduct evaluation using the *lighteval (Habib et al., 2023)* toolkit provided by Hugging Face. Consistent with the training configuration described above, we employ a sampling temperature of 0.6 and a *top-p* value of 0.95 for all model generations. For the logical reasoning tasks, we utilize the specialized evaluation suite from *Logic-RL*. We express our gratitude to the developers of these tools for their significant contributions to the open-source community, which ensured the reliability and comparability of our evaluation results. All evaluation experiments are performed on NVIDIA A100 (80GB) GPUs to maintain consistency in the computational environment.

B More Related Works

In this section, we provide a comprehensive review of the literature concerning post-training paradigms, the comparative analysis of SFT (Wei et al., 2021) and RL (Ziegler et al., 2019), and multi-task learning.

Post-Training for Large Language Models. The mainstream post-training paradigms mainly include SFT and RL. In the era of reasoning enhancement, SFT has evolved beyond simple instruction following to incorporate Chain-of-Thought (CoT) methodologies (Wei et al., 2022; Jin et al., 2026; Zhu et al., 2026; Li et al., 2026b), which encourage models to generate step-by-step reasoning paths (Li et al., 2025b; Longpre et al., 2023). Recent advancements have further refined this approach

through specialized reasoning fine-tuning, significantly boosting capabilities in mathematical and logical tasks (Xu et al., 2025; Hugging Face, 2025; Tan et al., 2026b;a). At the same time RL has shifted focus from traditional human preference alignment (Ouyang et al., 2022; Jin et al., 2025b; Li et al., 2026a; Men et al., 2025) to outcome-driven reasoning optimization. By leveraging verifiable rewards in domains like mathematics and coding (Hao et al., 2026), methods exemplified by DeepSeek-R1 (Guo et al., 2025) and other recent frameworks (Yu et al., 2025; Team et al., 2025; Zeng et al., 2025a) have greatly improved the model reasoning capabilities (Li et al., 2026c; Men et al., 2026b).

Comparative Analysis of SFT and RL in Single-Task Learning. A representative comparative study (Chu et al., 2025) characterize the distinction as "SFT memorizes, RL generalizes", observing that while SFT is responsible for fitting supervised training paths, RL focuses on generalization. Mechanistically, this divergence is often attributed to search space dynamics. Other work (Matsutani et al., 2025) propose that "RL squeezes, SFT expands", indicating that RL contracts the policy towards high-reward regions, whereas SFT broadens the sampling space. Furthermore, Jin et al. (2025a) find that RL can recover generalization capabilities lost during aggressive SFT. Other research has also compared the differences between the two in terms of out-of-distribution (OOD) task generalization (Huan et al., 2025). There is also workMukherjee et al. (2025) pointing out that compared to SFT, RL training only fine-tunes the small subnetwork. Recognizing the distinct properties of RL and SFT, recent hybrid approaches integrate on-policy RL with off-policy SFT to leverage the benefits of both paradigms (Fu et al., 2025; Zhang et al., 2025).

Multi-Task Learning. To enable deployment in complex real-world scenarios, enhancing the multi-task reasoning capabilities of LLMs has become a priority (Google, 2025; OpenAI, 2025; Yang et al., 2025). For SFT, the common practice is to train on mixed datasets spanning diverse tasks (Hugging Face, 2025). Existing studies suggest that conflicts in multi-task SFT are often severe and evolve throughout training, motivating adaptive data selection, curriculum design, and conflict-aware reweighting instead of static data mixing alone (Wu et al., 2024; Liang et al., 2025). In contrast, RL paradigms exhibit more diverse strategies. Many works explore multi-stage training (Zeng et al., 2025a), or employ curriculum learning strategies (Cho et al., 2024; Li et al., 2025a) to progressively build capabilities. Other studies investigate mixed-data RL by alleviating gradient imbalance (Wu et al., 2025a) or scaling RL on heterogeneous reasoning data (Liu et al., 2025a). Recent multi-task RL studies further show that cross-domain transfer is highly dependent on tasks: while some domains are mutually beneficial, others induce negative transfer or capacity competition (Li et al., 2025c; Cheng et al., 2025). Complementary optimization-based methods further seek to mitigate such conflicts by improving gradient compatibility across domains (Liang et al., 2026). Despite these advances, the paradigmatic divergence between SFT and RL in multi-task settings remains under-analyzed. Recent studies demonstrate that RL exhibits significantly less catastrophic forgetting than SFT (Shenfeld et al., 2025; Lai et al., 2025; Zhu et al., 2025). In contrast, our work is the first to systematically investigate this divergence through both theoretical and empirical analysis. As the field evolves towards Agentic RL and the number of reasoning tasks continues to grow (Zeng et al., 2025b; Men et al., 2026a), understanding these fundamental differences is increasingly important for scalable AGI development.

Model Merging Techniques. The paradigm of model merging has emerged as a highly efficient strategy to integrate the capabilities of multiple expert models without the prohibitive costs of re-training. Foundational works in this domain demonstrated that models can be fused via simple weight averaging (Wortsman et al., 2022) or through Task Arithmetic (Ilharco et al., 2022), which operates on task vectors to add or subtract specific model behaviors. However, directly combining multiple task vectors frequently leads to severe performance degradation due to task conflicts. To mitigate this, subsequent research introduced subspace and sparsification strategies. Most notably, TIES-Merging (Yadav et al., 2023) resolves interference by pruning task vectors based on parameter magnitudes and electing a dominant sign direction to eliminate sign conflicts during aggregation. Following a similar sparsification philosophy, DARE (Yu et al., 2024) randomly drops parameters and rescales the remaining weights to seamlessly assimilate homologous models. Beyond sparsification, other advanced methodologies calculate precise merging coefficients by evaluating parameter importance, utilizing tools such as the Fisher Information Matrix (Matena & Raffel, 2022) or closed-form statistical solutions for linear layers (Jin et al., 2022).

C Theory and Proof

C.1 RL Inner Product Decomposition

We present the proof of Lemma 4.3 in the main text. This lemma states that for GRPO, the inner product of training gradients for different tasks actually depends on the difference between rollouts within a group rather than the rollout itself.

Proof. We expand the inner product of the gradients defined in Definition 4.2:

$$\langle g_i(x), g_j(x') \rangle = \left\langle \frac{1}{G} \sum_{k=1}^{G} \hat{A}_{i,k} S_{i,k}, \frac{1}{G} \sum_{l=1}^{G} \hat{A}_{j,l} S_{j,l} \right\rangle.$$

Substituting the decomposition $S_{i,k} = \bar{S}_i + \delta S_{i,k}$, the term for task i becomes:

$$\sum_{k=1}^{G} \hat{A}_{i,k}(\bar{S}_i + \delta S_{i,k}) = \bar{S}_i \underbrace{\sum_{k=1}^{G} \hat{A}_{i,k}}_{=0} + \sum_{k=1}^{G} \hat{A}_{i,k} \delta S_{i,k}$$

$$= \sum_{k=1}^{G} \hat{A}_{i,k} \delta S_{i,k}.$$

The mean score direction $\bar{S}_i$ is eliminated due to the zero-sum property of the standardized advantages. Substituting this back into the inner product yields:

$$\langle g_i(x), g_j(x') \rangle = \frac{1}{G^2} \sum_{k,l} \hat{A}_{i,k} \hat{A}_{j,l} \langle \delta S_{i,k}, \delta S_{j,l} \rangle.$$

Taking the expectation over $x \sim \mathcal{D}_i$ and $x' \sim \mathcal{D}_j$ completes the proof. □

C.2 Proof of Theorem 4.5 (Upper Bound on Gradient Interference)

In this section, we derive the upper bounds for the gradient inner product for SFT and RL, as presented in Theorem 4.5.

C.2.1 Proof for SFT (Norm-Limited)

Proof. As defined in Eq 5, the gradient interference term is defined as the expected inner product of the expert score functions over independent data distributions $\mathcal{D}_i$ and $\mathcal{D}_j$ of different task i and j respectively:

$$\mathcal{I}_{\text{SFT}}(i,j) = \mathbb{E}_{x \sim \mathcal{D}_i, x' \sim \mathcal{D}_j} \left[\langle S_i^*(x), S_j^*(x') \rangle \right].$$

First, we apply the property $|\mathbb{E}[Z]| \leq \mathbb{E}[|Z|]$ (which follows from **Jensen's inequality**) together with the **Cauchy-Schwarz inequality** for the inner product ($|\langle \mathbf{u}, \mathbf{v} \rangle| \leq \|\mathbf{u}\|_2 \|\mathbf{v}\|_2$):

$$\begin{aligned} |\mathcal{I}_{\text{SFT}}(i,j)| &= \left| \mathbb{E}_{x,x'} \left[\langle S_i^*(x), S_j^*(x') \rangle \right] \right| \\ &\leq \mathbb{E}_{x,x'} \left[|\langle S_i^*(x), S_j^*(x') \rangle| \right] \\ &\leq \mathbb{E}_{x,x'} \left[\|S_i^*(x)\|_2 \cdot \|S_j^*(x')\|_2 \right]. \end{aligned}$$

Since the samples x and x' are drawn independently from $\mathcal{D}_i$ and $\mathcal{D}_j$, the expectation of the product can be factored into the product of expectations:

$$\mathbb{E}_{x,x'} \left[\|S_i^*(x)\|_2 \cdot \|S_j^*(x')\|_2 \right] = \mathbb{E}_x \left[\|S_i^*(x)\|_2 \right] \cdot \mathbb{E}_{x'} \left[\|S_j^*(x')\|_2 \right].$$

Next, we apply the property: $(\mathbb{E}[Z])^2 \leq \mathbb{E}[Z^2]$, implying $\mathbb{E}[Z] \leq \sqrt{\mathbb{E}[Z^2]}$. This can also be derived from **Jensen's inequality**.

$$\mathbb{E}_x \left[\|S_i^*(x)\|_2 \right] \leq \sqrt{\mathbb{E}_x \left[\|S_i^*(x)\|_2^2 \right]}$$

$$\mathbb{E}_{x'} \left[\|S_j^*(x')\|_2 \right] \leq \sqrt{\mathbb{E}_{x'} \left[\|S_j^*(x')\|_2^2 \right]}.$$

Finally, substituting this back into the expression and applying the bounds M_i and M_j defined in Assumption 4.4 (where $\mathbb{E}_{x\sim\mathcal{D}_i}\left[\|S_i^*(x)\|_2^2\right] \le M_i^2$), we obtain:

$$|\mathcal{I}_{\text{SFT}}(i,j)| \le M_i \cdot M_j.$$

This confirms the bound in Eq.equation UB-SFT. □

C.2.2 Proof for RL (Variance-Limited)

Proof. For RL (GRPO series, i.e., RL algorithms that normalize the advantage function within group), we start with the decomposition of the expected gradient inner product derived in Lemma 4.3:

$$\mathcal{I}_{\text{RL}}(i,j) = \mathbb{E}_{x,x'}\left[\frac{1}{G^2}\sum_{k=1}^{G}\sum_{l=1}^{G}\hat{A}_{i,k}\hat{A}_{j,l}\langle \delta S_{i,k}, \delta S_{j,l}\rangle\right].$$

To derive the upper bound, we first apply the property $|\mathbb{E}[Z]| \le \mathbb{E}[|Z|]$ (**Jensen's inequality**) and the Triangle Inequality to the sum inside the expectation:

$$\begin{aligned}|\mathcal{I}_{\text{RL}}(i,j)| &\le \mathbb{E}_{x,x'}\left[\left|\frac{1}{G^2}\sum_{k=1}^{G}\sum_{l=1}^{G}\hat{A}_{i,k}\hat{A}_{j,l}\langle \delta S_{i,k}, \delta S_{j,l}\rangle\right|\right] \\ &\le \mathbb{E}_{x,x'}\left[\frac{1}{G^2}\sum_{k=1}^{G}\sum_{l=1}^{G}|\hat{A}_{i,k}||\hat{A}_{j,l}||\langle \delta S_{i,k}, \delta S_{j,l}\rangle|\right].\end{aligned}$$

Next, we apply the **Cauchy-Schwarz inequality** ($|\langle \mathbf{u}, \mathbf{v}\rangle| \le \|\mathbf{u}\|_2\|\mathbf{v}\|_2$) to the inner product: $|\langle \delta S_{i,k}, \delta S_{j,l}\rangle| \le \|\delta S_{i,k}\|_2\|\delta S_{j,l}\|_2$. Thus the expression can be factored into the product of two independent sums:

$$|\mathcal{I}_{\text{RL}}(i,j)| \le \mathbb{E}_{x,x'}\left[\frac{1}{G^2}\left(\sum_{k=1}^{G}|\hat{A}_{i,k}|\|\delta S_{i,k}\|_2\right)\left(\sum_{l=1}^{G}|\hat{A}_{j,l}|\|\delta S_{j,l}\|_2\right)\right].$$

Since we consider two distinct tasks, the data sampling and rollout generation processes for task i and task j are independent. Consequently, the expectation of the product factors into the product of expectations:

$$|\mathcal{I}_{\text{RL}}(i,j)| \le \frac{1}{G^2}\underbrace{\mathbb{E}_x\left[\sum_{k=1}^{G}|\hat{A}_{i,k}|\|\delta S_{i,k}\|_2\right]}_{Term_i}\cdot\underbrace{\mathbb{E}_{x'}\left[\sum_{l=1}^{G}|\hat{A}_{j,l}|\|\delta S_{j,l}\|_2\right]}_{Term_j}.$$

We now bound the term for task i ($Term_i$). First, we apply the **Cauchy-Schwarz inequality** (for scalars) to the sum inside the expectation:

$$\sum_{k=1}^{G}|\hat{A}_{i,k}|\|\delta S_{i,k}\|_2 \le \sqrt{\sum_{k=1}^{G}\hat{A}_{i,k}^2}\cdot\sqrt{\sum_{k=1}^{G}\|\delta S_{i,k}\|_2^2}. \tag{8}$$

For RL algorithms of GRPO series, the advantage estimates are standardized within each group to stabilize training. Specifically, for a group of rewards $\{r_{i,1},\ldots,r_{i,G}\}$, the standardized advantage is computed as:

$$\hat{A}_{i,k} = \frac{r_{i,k} - \text{mean}(\{r_i\})}{\text{std}(\{r_i\})}.$$

By the definition of standardization, the mean of their squares is exactly 1:

$$\frac{1}{G}\sum_{k=1}^{G}\hat{A}_{i,k}^2 = 1 \quad\Longrightarrow\quad \sum_{k=1}^{G}\hat{A}_{i,k}^2 = G.$$

Substituting this property into the inequality 8 simplifies the bound to:

$$\sum_{k=1}^{G} |\hat{A}_{i,k}| \|\delta S_{i,k}\|_2 \leq \sqrt{G} \cdot \sqrt{\sum_{k=1}^{G} \|\delta S_{i,k}\|_2^2}.$$

Taking the expectation on both sides, we use **Jensen's inequality** for the concave function $f(z) = \sqrt{z}$ (implying $\mathbb{E}[\sqrt{Z}] \leq \sqrt{\mathbb{E}[Z]}$):

$$\begin{aligned} Term_i = \mathbb{E}_x \left[\sum_{k=1}^{G} |\hat{A}_{i,k}| \|\delta S_{i,k}\|_2 \right] &\leq \mathbb{E}_x \left[\sqrt{G} \cdot \sqrt{\sum_{k=1}^{G} \|\delta S_{i,k}\|_2^2} \right] \\ &\leq \sqrt{G} \cdot \sqrt{\mathbb{E}_x \left[\sum_{k=1}^{G} \|\delta S_{i,k}\|_2^2 \right]}. \end{aligned}$$

According to Assumption 4.4, the expected intra-group variance is bounded: $\mathbb{E}_x \left[\frac{1}{G} \sum_{k=1}^{G} \|\delta S_{i,k}\|_2^2 \right] \leq V_i^2$, which implies $\mathbb{E}_x \left[\sum_{k=1}^{G} \|\delta S_{i,k}\|_2^2 \right] \leq G V_i^2$. Substituting this into the inequality:

$$Term_i \leq \sqrt{G} \cdot \sqrt{G \cdot V_i^2} = G V_i.$$

Applying the same bound for task j ($Term_j \leq G V_j$) and substituting back into the expression C.2.2:

$$|\mathcal{I}_{\text{RL}}(i,j)| \leq \frac{1}{G^2} \cdot (G V_i) \cdot (G V_j) = V_i \cdot V_j.$$

This confirms the bound in Eq.equation UB-RL. □

C.3 Generalization to Other RL Algorithm

The variance-limited interference mechanism is not exclusive to GRPO. First, in the context of the reasoning enhancement or RLVR Guo et al. (2025) focused on in this paper, most recent reasoning-oriented RL algorithms (e.g., GSPO (Zheng et al., 2025), DAPO (Yu et al., 2025)) generally follow the GRPO paradigm by generating a group of rollouts for a single prompt and performing group-wise advantage normalization. Consequently, these algorithms also satisfy the derivation presented above. Second, regarding classical RL algorithms, this property also holds approximately. In fact, as long as there is normalization for the advantage and the on-policy property, our derivation can hold. In this section, we provide a derivation using PPO (Schulman et al., 2017) as an example.

Unlike GRPO which normalizes advantages within a group of outputs for the same prompt, PPO (as implemented in VeRL (Sheng et al., 2024)) typically employs **token-level GAE** and performs **Batch-Level Advantage Normalization**. We show that our theoretical framework seamlessly generalizes to this setting.

Definition (PPO Gradient with Batch Normalization). Consider a training batch $\mathcal{B}$ consisting of multiple trajectories. Let $\mathcal{K}$ be the set of indices for all valid tokens in this batch (flattening batch and time dimensions), with total count $N = |\mathcal{K}|$. For each token $k \in \mathcal{K}$, let $S_k = \nabla_\theta \log \pi_\theta(a_k|s_k)$ be the score function and $\hat{A}_k$ be the normalized advantage. The empirical gradient estimate for task i is given by:

$$g_i^{\text{PPO}} = \frac{1}{N} \sum_{k=1}^{N} \hat{A}_k S_k.$$

Crucially, the **Batch-Level Advantage Normalization** ensures that the advantages also satisfy the property that the sum of advantage values is 0 and the variance is 1 over the batch:

$$\sum_{k=1}^{N} \hat{A}_k = 0, \quad \text{and} \quad \sum_{k=1}^{N} \hat{A}_k^2 = N.$$

Lemma (PPO Gradient Decomposition). Let $\bar{S}_{\mathcal{B}} = \frac{1}{N}\sum_{k=1}^{N} S_k$ be the **batch mean score function**, representing the common optimization direction of the entire batch. Let $\delta S_k = S_k - \bar{S}_{\mathcal{B}}$ be the **residual score** for the k-th token. The PPO gradient can be decomposed as:

$$g_i^{\text{PPO}} = \frac{1}{N}\sum_{k=1}^{N} \hat{A}_k \delta S_k.$$

Proof. Substituting $S_k = \bar{S}_{\mathcal{B}} + \delta S_k$ into the gradient definition:

$$g_i^{\text{PPO}} = \frac{1}{N}\sum_{k=1}^{N} \hat{A}_k(\bar{S}_{\mathcal{B}} + \delta S_k) = \frac{1}{N}\underbrace{\left(\sum_{k=1}^{N} \hat{A}_k\right)}_{=0}\bar{S}_{\mathcal{B}} + \frac{1}{N}\sum_{k=1}^{N} \hat{A}_k \delta S_k = \frac{1}{N}\sum_{k=1}^{N} \hat{A}_k \delta S_k.$$

This confirms that PPO also filters out the common mean component $\bar{S}_{\mathcal{B}}$, relying solely on the residuals over batch.

Theorem (PPO Interference Upper Bound). Let Assumption 4.4 be adapted to PPO such that the expected squared norm of the residual scores over a batch of size N is bounded:

$$\mathbb{E}_{\mathcal{B}\sim\mathcal{D}_i}\left[\frac{1}{N}\sum_{k=1}^{N} \|\delta S_{i,k}\|_2^2\right] \leq V_i^2.$$

Then, the expected gradient inner product between task i and task j (trained with PPO) satisfies:

$$|\mathcal{I}_{\text{PPO}}(i,j)| \leq V_i \cdot V_j.$$

The derivation steps are the same as in Section C.2.

Discussion on the Differences with GRPO Series. While the upper bound for PPO is algebraically identical to GRPO in Eq UB-RL, the interpretation of δS differs slightly. In RL algorithms utilizing group-wise advantage normalization (e.g. GRPO), δS captures the intra-group policy divergence corresponding to a single prompt. In contrast, within PPO, this term represents the deviation across the prompts of entire batch. Nevertheless, the core mechanism remains the same: the advantage term $\hat{A}$ acts as a high-pass filter that removes high-magnitude common direction ($\bar{S}$). In GRPO, the filtered $\bar{S}$ represents the common characteristics of the model's responses to a specific input. This common direction exhibits a large magnitude because, given a fixed model and input, variance in model activations is inherently limited. Conversely, in PPO, the filtering applies to the common direction across all samples in a batch. Intuitively, this common direction is weaker than that derived from a single input. This results in PPO being more perturbative than GRPO during parallel training. Supporting evidence can be found in Appendix E.1. However, since samples within a batch typically belong to same task domain (e.g., mathematics or coding), this process still effectively filters out significant commonalities associated with task-specific features and linguistic priors. Consequently, the upper bound of interference in PPO is determined by the policy fluctuations within specific task. Drawing from this analysis, we propose a potential strategy to optimize PPO: by generating a group of rollouts for a single input and constraining scope of advantage normalization to these intra-group trajectories, PPO may potentially achieve orthogonality effects comparable to those of GRPO.

C.4 Supplementary Analysis: Cosine Similarity of Multi-Task Gradients

This subsection provides a complementary analysis to the derivation in the main text. In Section 4.3, we upper bounded the *gradient inner product* between tasks, which characterizes practical interference. Here, we further study the problem from a directional perspective by factoring out scale and directly examining the interference induced by different tasks under SFT and RL. Specifically, we analyze the *cosine similarity* between task gradients for the two training paradigms.

To make the directional comparison explicit, we introduce an additional high-dimensional decomposition assumption on the score function. Under this assumption, we show that SFT admits a positive lower bound on cross-task cosine similarity due to the persistence of a shared dominant direction, whereas RL admits an upper bound controlled only by residual coupling, because the shared direction is exactly canceled by the zero-sum advantage weights.

Setup. Consider two independent tasks i and j. For task i, given prompt x_i, the model generates G rollouts $\{y_{i,k}\}_{k=1}^{G}$; similarly, task j generates $\{y_{j,m}\}_{m=1}^{G}$.

We assume that the score function of each rollout admits the decomposition

$$S_{i,k}(x_i) = \mu + \epsilon_{i,k}, \qquad S_{j,m}(x_j) = \mu + \epsilon_{j,m},$$

where μ denotes a task-shared dominant direction, and ϵ is a task- and sample-specific residual component.

High-dimensional geometric condition. We assume that, on a high-probability event $\mathcal{E}$,

$$\|\mu\| = M_\mu, \qquad \|\epsilon_{i,k}\| = \|\epsilon_{j,m}\| = \sigma = \eta M_\mu, \qquad \eta \ll 1.$$

That is, the residual norm is concentrated and much smaller than the shared dominant direction.

We further define the *task residual coupling factor* $\gamma_{i,j} \in [0, 1]$ as

$$\max_{k,m} \frac{|\langle \epsilon_{i,k}, \epsilon_{j,m} \rangle|}{\sigma^2} \le \gamma_{i,j}.$$

Similarly, for distinct rollouts within the same task, we assume an intra-task coupling factor $\gamma_{\text{in}} \ll 1$ such that

$$\max_{k \neq m} \frac{|\langle \epsilon_{i,k}, \epsilon_{i,m} \rangle|}{\sigma^2} \le \gamma_{\text{in}}.$$

SFT gradients preserve the shared dominant direction. Under SFT, the gradient is the uniform average of score functions:

$$g_i^{\text{SFT}} = \frac{1}{G} \sum_{k=1}^{G} S_{i,k}(x_i) = \mu + \bar{\epsilon}_i, \qquad \bar{\epsilon}_i := \frac{1}{G} \sum_{k=1}^{G} \epsilon_{i,k}.$$

By the triangle inequality,

$$\|\bar{\epsilon}_i\| \le \sigma = \eta M_\mu.$$

We first lower bound the numerator:

$$\begin{aligned} \langle g_i^{\text{SFT}}, g_j^{\text{SFT}} \rangle &= \langle \mu + \bar{\epsilon}_i, \mu + \bar{\epsilon}_j \rangle \\ &= \|\mu\|^2 + \langle \mu, \bar{\epsilon}_i \rangle + \langle \mu, \bar{\epsilon}_j \rangle + \langle \bar{\epsilon}_i, \bar{\epsilon}_j \rangle \\ &\ge M_\mu^2 - M_\mu \|\bar{\epsilon}_i\| - M_\mu \|\bar{\epsilon}_j\| - \|\bar{\epsilon}_i\| \, \|\bar{\epsilon}_j\| \\ &\ge M_\mu^2 (1 - 2\eta - \eta^2). \end{aligned}$$

Next, we upper bound the denominator:

$$\begin{aligned} \|g_i^{\text{SFT}}\| \, \|g_j^{\text{SFT}}\| &\le (\|\mu\| + \|\bar{\epsilon}_i\|)(\|\mu\| + \|\bar{\epsilon}_j\|) \\ &\le M_\mu^2 (1 + \eta)^2. \end{aligned}$$

Therefore,

$$|\cos(g_i^{\text{SFT}}, g_j^{\text{SFT}})| \ge \frac{1 - 2\eta - \eta^2}{(1 + \eta)^2}. \tag{9}$$

When $\eta \ll 1$, the right-hand side remains strictly positive and close to 1. Hence, even for distinct tasks, SFT gradients remain directionally aligned because the shared component μ cannot be removed by uniform averaging.

RL gradients cancel the shared dominant direction. For RL algorithms such as GRPO, the gradient is weighted by normalized advantages $\hat{A}_{i,k}$ satisfying

$$\sum_{k=1}^{G} \hat{A}_{i,k} = 0, \qquad \sum_{k=1}^{G} \hat{A}_{i,k}^2 = G.$$

Thus,

$$\begin{aligned} g_i^{\mathrm{RL}} &= \frac{1}{G}\sum_{k=1}^{G}\hat{A}_{i,k}(\mu+\epsilon_{i,k}) \\ &= \mu\left(\frac{1}{G}\sum_{k=1}^{G}\hat{A}_{i,k}\right)+\frac{1}{G}\sum_{k=1}^{G}\hat{A}_{i,k}\epsilon_{i,k} \\ &= \frac{1}{G}\sum_{k=1}^{G}\hat{A}_{i,k}\epsilon_{i,k}. \end{aligned}$$

Similarly,

$$g_j^{\mathrm{RL}} = \frac{1}{G}\sum_{m=1}^{G}\hat{A}_{j,m}\epsilon_{j,m}.$$

We first upper bound the numerator:

$$\begin{aligned} |\langle g_i^{\mathrm{RL}}, g_j^{\mathrm{RL}}\rangle| &= \left|\frac{1}{G^2}\sum_{k=1}^{G}\sum_{m=1}^{G}\hat{A}_{i,k}\hat{A}_{j,m}\langle\epsilon_{i,k},\epsilon_{j,m}\rangle\right| \\ &\le \frac{1}{G^2}\sum_{k=1}^{G}\sum_{m=1}^{G}|\hat{A}_{i,k}|\,|\hat{A}_{j,m}|\,|\langle\epsilon_{i,k},\epsilon_{j,m}\rangle| \\ &\le \frac{\gamma_{i,j}\sigma^2}{G^2}\left(\sum_{k=1}^{G}|\hat{A}_{i,k}|\right)\left(\sum_{m=1}^{G}|\hat{A}_{j,m}|\right). \end{aligned}$$

By Cauchy–Schwarz,

$$\sum_{k=1}^{G}|\hat{A}_{i,k}| \le \sqrt{G}\left(\sum_{k=1}^{G}\hat{A}_{i,k}^2\right)^{1/2} = G,$$

and similarly for task j, yielding

$$|\langle g_i^{\mathrm{RL}}, g_j^{\mathrm{RL}}\rangle| \le \gamma_{i,j}\sigma^2.$$

We now lower bound the denominator. For one task,

$$\begin{aligned} \|g_i^{\mathrm{RL}}\|^2 &= \frac{1}{G^2}\sum_{k=1}^{G}\sum_{m=1}^{G}\hat{A}_{i,k}\hat{A}_{i,m}\langle\epsilon_{i,k},\epsilon_{i,m}\rangle \\ &= \frac{1}{G^2}\left(\sum_{k=1}^{G}\hat{A}_{i,k}^2\|\epsilon_{i,k}\|^2+\sum_{k\neq m}\hat{A}_{i,k}\hat{A}_{i,m}\langle\epsilon_{i,k},\epsilon_{i,m}\rangle\right). \end{aligned}$$

Using $\|\epsilon_{i,k}\|^2=\sigma^2$ and the intra-task coupling bound,

$$\|g_i^{\mathrm{RL}}\|^2 \ge \frac{1}{G^2}\left(G\sigma^2-\gamma_{\mathrm{in}}\sigma^2\sum_{k\neq m}|\hat{A}_{i,k}|\,|\hat{A}_{i,m}|\right).$$

Since

$$\sum_{k\neq m}|\hat{A}_{i,k}|\,|\hat{A}_{i,m}| \le \left(\sum_{k=1}^{G}|\hat{A}_{i,k}|\right)^2 \le G^2,$$

we obtain

$$\|g_i^{\mathrm{RL}}\|^2 \ge \frac{\sigma^2}{G}(1-\gamma_{\mathrm{in}}G).$$

The same bound holds for task j. Combining the bounds above,

$$|\cos(g_i^{\mathrm{RL}}, g_j^{\mathrm{RL}})| \le \frac{\gamma_{i,j}G}{1-\gamma_{\mathrm{in}}G}. \tag{10}$$

In the high-dimensional regime where $\gamma_{\mathrm{in}}\to 0$,

$$|\cos(g_i^{\mathrm{RL}}, g_j^{\mathrm{RL}})| \lesssim \mathcal{O}(\gamma_{i,j}G).$$

Interpretation. Eq. equation 9 and Eq. equation 10 reveal a directional contrast between SFT and RL. In SFT, averaging preserves the shared dominant direction μ, which induces a strictly positive lower bound on cross-task cosine similarity. In RL, the zero-sum advantage weighting removes the shared direction algebraically, so the cosine similarity is controlled only by the residual coupling factor $\gamma_{i,j}$. Therefore, under weak residual coupling, RL gradients from different tasks become nearly orthogonal, which provides an additional explanation for the observed task coexistence. We emphasize, however, that this conclusion does not imply that all task pairs in RL are non-interfering. In particular, the residual coupling factor $\gamma_{i,j}$ need not always be very small. We discuss how to assess such cases in detail in Appendix D.1.

D PARALLEL RL

D.1 METHOD FOR JUDGING DIFFERENT TASKS

As discussed in §5.4 of main text, the efficacy of Parallel-RL is sensitive to task selection. Not all task pairs qualify as "coexist"; certain tasks may exhibit great interference and are thus unsuitable for parallel training. In this section, building upon the analysis in §4, we conduct a preliminary exploration into defining whether two tasks are independent and can be trained in parallel.

Based on the qualitative analysis in §4.2, we found that in two different tasks, high-reward trajectories activate disparate neural circuits and patterns. Thus an intuitive method to verify this is to analyze the distributions of the score functions S for the two tasks. Specifically, we visualize these distributions using t-SNE. Given that t-SNE is a dimensionality reduction technique, the separation of score function distributions in the t-SNE latent space serves as a practical heuristic for determining task coexistence. As illustrated in Figure 3 of the main text and Figure 6 in the Appendix E.3, the training data for the selected tasks in main experiment exhibit relative separation for RL, suggesting their suitability for parallel training. Next, we present a counter-example involving a task where the training data likely interferes with others. We selected SynLogic (Liu et al., 2025b), a recent influential work that synthesizes data for logical reasoning games (e.g., Sudoku, Mazes), which we refer to as the **Game** task. We trained DeepSeek-R1-Distill-Qwen-1.5B on this game task following the same protocol as the main experiments and subsequently merged it with the **Math** and **Code** models trained in §5.2, respectively. The results reveal significant interference between the Game task and both the Math and Code tasks. The results are shown in Figure 4 (c) and (d). Consistent with Table 4 in the main text, we primarily compare the performance of the merged Parallel-RL model against the corresponding task-specific model. For instance, on the math task, we evaluate the performance change of Parallel-RL relative to the single-task math model. We found that Parallel-RL of Game and Math resulted in a performance decline of 4.0% on math and 6.0% on game. Similarly, a decline of 3.3% and 4.9% was observed in Parallel-RL of Game and Code. These results demonstrate significant interference. Consistent with our methodology, we generated score functions for the SynLogic data and visualized them using t-SNE. As depicted in Figure 4 (a) and (b), there is a marked overlap between the distributions of the Game task and those of the Math and Code tasks. This observation preliminarily validates the effectiveness of our proposed criterion for task compatibility.

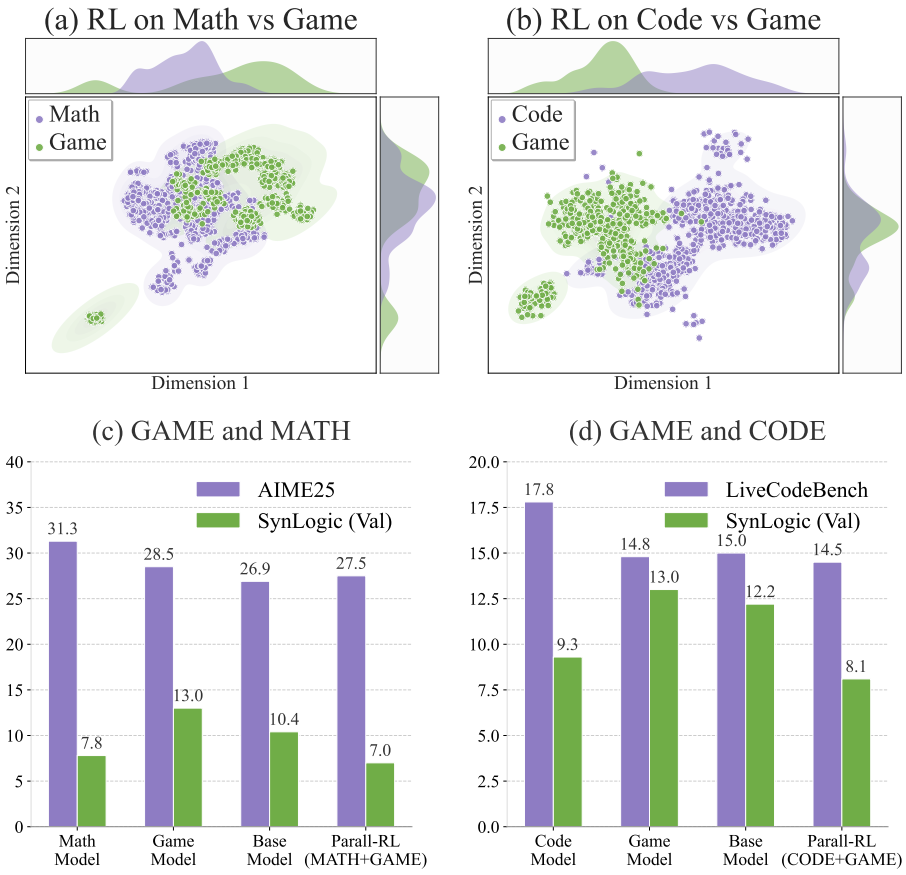

Figure 4: **Analysis of RL training on interfering tasks. Top:** Distribution of the score function S sampled during RL training on math, code and game. **Bottom:** Corresponding performance comparison on Math (AIME25) and Code (LiveCodeBench) tasks. Training game models in parallel with math or code models can lead to significant performance losses.

D.2 Single-Task Training Techniques: Trade-off in Parallel RL

Based on the upper bound derived in Section 4.3, the gradient interference of RL between different tasks is limited by the intra-group variance of rollouts, specifically $|\mathcal{I}_{\text{RL}}(i,j)| \leq V_i \cdot V_j$. A deeper analysis of this theory reveals an interesting trade-off in Parallel-RL training: *the trade-off between single-task exploration and multi-task interference.*

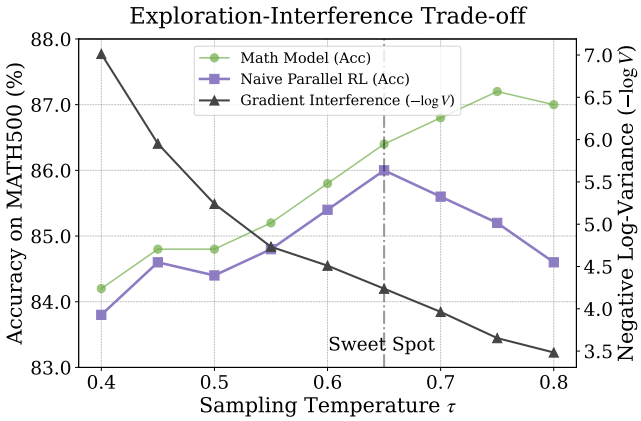

Figure 5: The Trade-off between Single-Task Exploration and Multi-Task Coexistence. When rollout temperature τ grows, we calculates the performance of Math Model, intra-group gradient variance (measured by $-\log V$), and Naive Parallel-RL accuracy on MATH500. Increasing τ improves single-task performance but also increases gradient variance, leading to higher gradient interference. As a result, the non-monotonic trend of Parallel-RL performance reveals a sweet spot that balances exploration benefits and interference costs.

1. Exploration Benefit: In single-task RL, increasing exploration and output diversity helps the model discover low-probability but high-reward reasoning trajectory (Guo et al., 2025). As a result, higher output diversity often leads to improved convergence and performance in single-task RL.

2. Interference Cost: However, increased exploration also amplifies the diversity among the G rollout trajectories generated for the same prompt, thereby enlarging the intra-group variance V. According to Theorem 4.5, the interference between tasks rises accordingly. While high exploration benefits single-task training, it increases the gradient interference. This may cause greater conflicts and performance loss during the merging phase.

Empirical Analysis.

We conducted an ablation study using sampling temperature τ to control exploration. Since τ directly affects the entropy of the policy π_θ, it serves as an effective control variable. Based on preliminary tests, we set the temperature range to $\tau \in [0.4, 0.8]$. We varied the rollout temperature for the Math Model training while keeping other tasks constant. We evaluated the accuracy of the Naive Parallel RL model on MATH500 and measured the negative log-variance ($-\log V$) of the intra-group variance V. Due to resource constraints, we performed this analysis using LoRA on a data subset.

As shown in Figure 5, the performance of Math Model improves as τ increases from 0.4 to 0.8. At the same time $-\log V$ consistently decreases. This indicates a rise in intra-group variance and a corresponding increase in the interference upper bound. Crucially, the Naive Parallel-RL performance exhibits a non-monotonic trend: accuracy first increases and then degrades as τ grows. At low temperatures, interference is small but single-task learning is insufficient; at high temperatures, single-task performance improves but excessive inter-task interference degrades the merged model. This curve clearly illustrates the exploration–interference trade-off.

Notably, around $\tau \approx 0.65$, Parallel-RL achieves the best performance, achieving an optimal balance between single-task capability and multi-task interference. This result highlights that, under Parallel-RL, optimizing single-task exploration alone is insufficient; controlling multi-task interference is equally crucial for effective task merging with low loss.

E MORE EXPERIMENTAL RESULTS

In §5.2 of the main text, building upon our comprehensive empirical and theoretical analyses, we proposed the **Parallel-RL** training paradigm. In this appendix, we present an extensive set of additional experiments to further substantiate the efficacy and robustness of this approach. Specifically, §E.1 provides the complete results corresponding to the main experiments discussed in Table 4. §E.2 presents expanded ablation studies (complementing Table 5) to demonstrate how Parallel-RL effectively decouples capabilities across diverse tasks. §E.3 visualizes the distributions of score functions across tasks as referenced in §4.2. Finally, in §E.4, we conduct a Pass@k evaluation of the Parallel-RL model to provide a more rigorous assessment of its reasoning boundaries.

E.1 MORE MAIN EXPERIMENT RESULTS

In this section, we present the results of Parallel-RL derived from two additional training settings on the 1.5B base model: (1) GRPO training using LoRA, and (2) full-parameter training using PPO.

We found that when using LoRA for training, the performance loss caused by parameter interference is minimal. Even direct summation of ΔW_i (Naive Parallel-RL (sum)) maintains 98% of the accuracy of the corresponding single-task models. Adapted Parallel-RL even outperforms Single Task RL, achieving an average performance of 103% of Single Task RL. This indicates that the reasoning capability of task-specific LoRA modules is largely preserved during the merging process.

For PPO, Parallel-RL is equally effective. Even Naive Parallel-RL model achieves an average performance improvement of 5.3% compared to the base model. Adapted Parallel-RL showed an average improvement of 9.1%. This demonstrates the effectiveness of the Parallel RL paradigm. However, the performance of the single-task model that Naive Parallel (sum) of PPO retained (92.9%) is lower than that of GRPO shown in Table 4 (94.2%). This aligns with our theoretical analysis in Appendix C.3. GRPO-based parallel training exhibits lower interference compared to PPO.

Table 6: The accuracy (%) of several multi task training methods. Superscripts denote the performance gap relative to the **corresponding Single-Task Baseline**. Δ Base shows the average performance improvement over the Base Model. Retention (%) represents the **percentage of performance retained** compared to the Single-Task Baseline.

Base Model	Method	Math		Science		Logic	Code	Average Statistic	
		MATH500	AIME2025	MMLU	GPQA	KK	LiveCodeBench	Δ Base	Retention (%)
	Baselines (LoRA GRPO)								
DeepSeek-R1-Distill-Qwen-1.5B (LoRA)	Base Model	82.0	26.9	34.9	32.3	31.0	15.0	-	-
	Single-Task SFT	84.4	28.3	43.5	34.8	35.0	15.8	+3.3	-
	Single-Task RL	85.6	31.3	48.5	35.9	38.0	17.8	+5.8	-
	Previous Multi Task Training Paradigms								
	Mixed Data SFT	$84.6^{+0.2}$	$27.1^{-1.2}$	$38.9^{-4.6}$	$32.3^{-2.5}$	$34.0^{-1.0}$	$16.0^{+0.2}$	+1.8	-
	Multi Stage SFT	$78.2^{-6.2}$	$19.2^{-9.1}$	$31.1^{-12.4}$	$25.3^{-9.5}$	$9.0^{-26.0}$	$14.3^{-1.5}$	-7.5	-
	Mixed Data RL	$85.2^{-0.4}$	$30.5^{-0.8}$	$43.2^{-5.3}$	$34.8^{-1.1}$	$37.0^{-1.0}$	$15.7^{-2.1}$	+4.1	-
	Multi Stage RL	$86.6^{+1.0}$	$31.5^{+0.2}$	$49.3^{+0.8}$	**36.4**$^{+0.5}$	$43.0^{+5.0}$	**17.3**$^{-0.5}$	+7.0	-
	Parallel Training								
	Naive Parallel-SFT *(mean)*	$51.0^{-33.4}$	$17.3^{-11.0}$	$26.7^{-16.8}$	$25.8^{-9.0}$	$20.0^{-15.0}$	$9.7^{-6.1}$	-11.9	62.2
	Naive Parallel-SFT *(sum)*	$52.6^{-31.8}$	$16.9^{-11.4}$	$27.1^{-16.4}$	$24.7^{-10.1}$	$25.0^{-10.0}$	$8.8^{-7.0}$	-11.2	64.1
	Naive Parallel-RL *(mean)*	$85.4^{-0.2}$	$28.5^{-2.8}$	$45.9^{-2.6}$	$31.8^{-4.1}$	$39.0^{+1.0}$	$16.9^{-0.9}$	+4.2	96.3
	Naive Parallel-RL *(sum)*	$85.2^{-0.4}$	$29.2^{-2.1}$	$46.5^{-2.0}$	$32.8^{-3.1}$	$40.0^{+2.0}$	$16.9^{-0.9}$	+4.8	97.5
	TIES Parallel-RL	$86.0^{+0.4}$	$31.0^{-0.3}$	$47.0^{-1.5}$	$34.8^{-1.1}$	$42.0^{+4.0}$	$16.3^{-1.5}$	+5.8	100.0
	SVD Parallel-RL	$85.4^{-0.2}$	$31.5^{+0.2}$	$45.8^{-2.7}$	$32.8^{-3.1}$	$37.0^{-1.0}$	$15.6^{-2.2}$	+4.3	96.5
	Adapted Parallel-RL	**86.6**$^{+1.0}$	**32.1**$^{+0.8}$	**48.3**$^{-0.2}$	$35.4^{-0.5}$	**45.0**$^{+7.0}$	**17.3**$^{-0.5}$	+7.1	**102.9**
	Baselines (Full Parameter PPO)								
DeepSeek-R1-Distill-Qwen-1.5B (PPO)	Base Model (1.5B)	82.0	26.9	34.9	32.3	31.0	15.0	-	-
	Single-Task SFT	85.8	31.5	49.1	35.4	37.0	16.3	+5.5	-
	Single-Task RL	86.6	31.5	49.8	37.9	46.0	20.3	+8.3	-
	Previous Multi Task Training Paradigms								
	Mixed Data SFT	$85.6^{-0.2}$	$30.1^{-1.4}$	$48.5^{-0.6}$	$34.3^{-1.1}$	$38.0^{+1.0}$	$17.0^{+0.7}$	+5.2	-
	Multi Stage SFT	$71.0^{-14.8}$	$21.8^{-9.7}$	$33.2^{-15.9}$	$23.4^{-12.0}$	$11.0^{-26.0}$	$12.1^{-4.2}$	-8.3	-
	Mixed Data RL	$85.0^{-1.6}$	$30.0^{-1.5}$	$48.2^{-1.6}$	$37.4^{-0.5}$	$41.0^{-5.0}$	$16.0^{-4.3}$	+5.9	-
	Multi Stage RL	$87.8^{+1.2}$	$33.5^{+2.0}$	$53.5^{+3.7}$	$39.4^{+1.5}$	$43.0^{-3.0}$	**21.8**$^{+1.5}$	+9.5	-
	Parallel Training								
	Naive Parallel-SFT *(mean)*	$63.2^{-22.6}$	$19.2^{-12.3}$	$26.9^{-22.2}$	$25.3^{-10.1}$	$26.0^{-11.0}$	$9.3^{-7.0}$	-8.7	66.6
	Naive Parallel-SFT *(sum)*	$61.2^{-24.6}$	$20.0^{-11.5}$	$25.9^{-23.2}$	$26.8^{-8.6}$	$23.0^{-14.0}$	$10.1^{-6.2}$	-9.2	65.5
	Naive Parallel-RL *(mean)*	$85.4^{-1.2}$	$27.3^{-4.2}$	$49.1^{-0.7}$	$33.8^{-4.1}$	$38.0^{-8.0}$	$16.1^{-4.2}$	+4.6	91.7
	Naive Parallel-RL *(sum)*	$84.8^{-1.8}$	$28.3^{-3.2}$	$48.9^{-0.9}$	$34.3^{-3.6}$	$40.0^{-6.0}$	$16.5^{-3.8}$	+5.1	92.9
	TIES Parallel-RL	$86.0^{-0.6}$	$30.4^{-1.1}$	$51.4^{+1.6}$	$36.9^{-1.0}$	$44.0^{-2.0}$	$18.5^{-1.8}$	+7.5	98.2
	SVD Parallel-RL	$85.4^{-1.2}$	$29.6^{-1.9}$	$49.9^{+0.1}$	$35.4^{-2.5}$	$39.0^{-7.0}$	$17.5^{-2.8}$	+5.8	94.4
	Adapted Parallel-RL	**87.0**$^{+0.4}$	**32.1**$^{+0.6}$	**54.4**$^{+4.6}$	**38.4**$^{+0.5}$	**45.0**$^{-1.0}$	$19.5^{-0.8}$	+9.1	**101.6**

E.2 More Ablation Study Results

In this section, we present more ablation studies on the trained Naive Parallel-RL model to evaluate the impact of excluding specific task updates on the overall multi-task performance. As shown in Table 7, removing the update ΔW_i of specific task from the Naive Parallel RL model results in a sharp performance drop on that specific task, with an average change of -6.0% (Δ_{Target}). However, the performance on the remaining tasks remains robust or even slightly improves, with an average change of +0.9% (Δ_{Others}). This further shows that parameter updates ΔW_i derived from RL on different tasks exhibit low coherence and high decoupling. The update vector for a given task primarily affects the performance of that task with minimal impact on others. This suggests that beyond improving multi-task training efficiency, the Parallel-RL paradigm enhances the **modularity and decoupling of task capabilities in the final model**. We envision that future advancements in Parallel-RL algorithms could enable the on-demand composition of model capabilities, allowing for flexible adaptation to specialized real-world environments with specific requirements.

Table 7: Comprehensive ablation study across different settings. Superscripts denote the performance gap (Δ) relative to the full **Parallel-RL** model. Red indicates drop, Blue indicates improvement.

Model Setting	Method	MATH500 (Math)	MMLU (Science)	KK (Logic)	LCB (Code)	Δ_{Target} (Impact)	Δ_{Others} (Side-effect)
				Naive Parallel RL			
1.5B LoRA	Parallel-RL	85.2	46.5	40.0	16.9	-	-
	- w/o Math	$82.4^{-2.8}$	$47.3^{+0.8}$	$38.0^{-2.0}$	$17.1^{+0.2}$	-2.8	-0.3
	- w/o Science	$85.0^{-0.2}$	$38.7^{-7.8}$	$37.0^{-3.0}$	$17.5^{+0.6}$	-7.8	-0.9
	- w/o Logic	$84.6^{-0.6}$	$47.1^{+0.6}$	$32.0^{-8.0}$	$16.3^{-0.6}$	-8.0	-0.2
	- w/o Code	$85.8^{+0.6}$	$46.3^{-0.2}$	$39.0^{-1.0}$	$14.7^{-2.2}$	-2.2	-0.2
				Baselines			
	Single-Task RL	85.6	48.5	38.0	17.8	-	-
	Base Model	82.0	34.9	31.0	15.0	-	-
				Naive Parallel RL			
1.5B PPO	Parallel-RL	84.8	48.9	36.0	16.5	-	-
	- w/o Math	$82.8^{-2.0}$	$48.5^{-0.4}$	$38.0^{+2.0}$	$18.6^{+2.1}$	-2.0	+1.2
	- w/o Science	$85.8^{+1.0}$	$35.6^{-13.3}$	$44.0^{+8.0}$	$18.1^{+1.6}$	-13.3	+3.5
	- w/o Logic	$84.6^{-0.2}$	$48.3^{-0.6}$	$30.0^{-6.0}$	$16.8^{+0.3}$	-6.0	-0.2
	- w/o Code	$87.4^{+2.6}$	$50.3^{+1.4}$	$45.0^{+9.0}$	$12.5^{-4.0}$	-4.0	+4.3
				Baselines			
	Single-Task RL	86.6	49.8	46.0	20.3	-	-
	Base Model	82.0	34.9	31.0	15.0	-	-
				Naive Parallel RL			
7B GRPO	Parallel-RL	93.6	59.6	68.0	38.0	-	-
	- w/o Math	$90.2^{-3.4}$	$59.1^{-0.5}$	$67.0^{-1.0}$	$41.0^{+3.0}$	-3.4	+0.5
	- w/o Science	$94.2^{+0.6}$	$49.7^{-9.9}$	$68.0^{+0.0}$	$39.5^{+1.5}$	-9.9	+0.7
	- w/o Logic	$93.8^{+0.2}$	$60.2^{+0.6}$	$60.0^{-8.0}$	$38.3^{+0.3}$	-8.0	+0.4
	- w/o Code	$95.2^{+1.6}$	$61.8^{+2.2}$	$69.0^{+1.0}$	$33.9^{-4.1}$	-4.1	+1.6
				Baselines			
	Single-Task RL	94.6	63.5	68.0	42.3	-	-
	Base Model	91.8	51.8	62.0	36.5	-	-

E.3 More Visualizations

In §4.4, we presented a comparative visualization of the score function S distributions for Math and Science tasks trained via RL and SFT, respectively. In this section, we provide additional t-SNE visualizations of score function distributions covering a broader range of tasks. As illustrated in Figure 6, the top row displays the distributions for different tasks trained using RL. We observe a clear separation between task clusters with minimal overlap. Conversely, the bottom row shows the score functions S for SFT, where we observe a significant degree of overlap across different tasks. This contrast serves as empirical evidence reflecting the high interference and conflicts of parameter updates inherent to SFT training, as opposed to the incoherence and coexistence nature of RL updates.

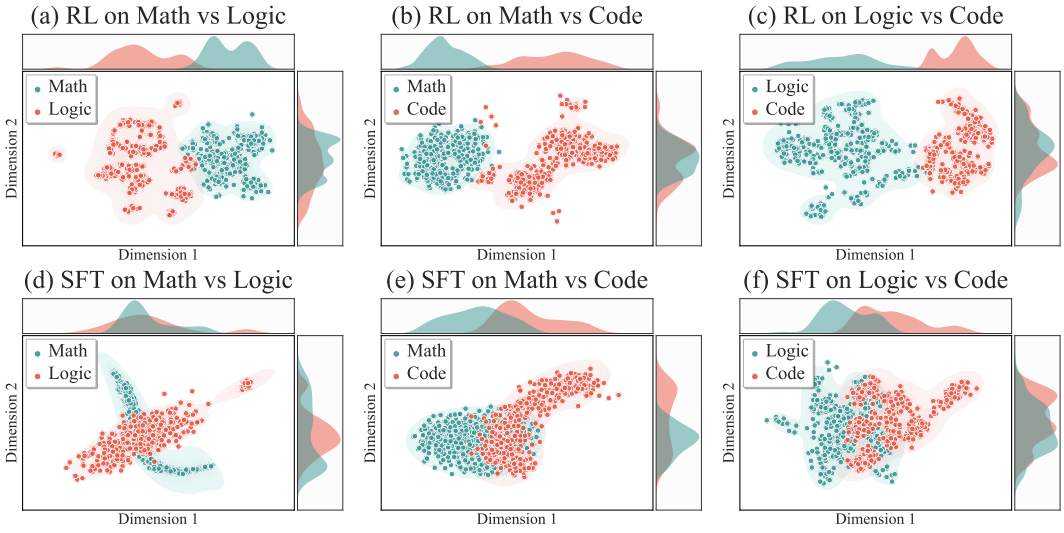

Figure 6: Distribution of the score function S sampled during training by RL and SFT on different tasks (Math, Code and Logic). We use t-SNE to reduce the dimensionality of the high-dimensional score function vectors for visualization.

E.4 Pass@K Analysis

Accuracy evaluation can be sensitive to the decoding strategy. To provide a more robust evaluation of the stability and reasoning capability boundaries of the Parallel-RL paradigm, we adopt the **pass@k** metric. This metric measures the probability that at least one out of k generated samples for a given problem is correct, serving as a proxy for the model's potential reasoning upper bound. In this section, we conduct a pass@k analysis on the **Naive Parallel-RL** described in Section 5.2. Specifically, we select four representative benchmarks for our four tasks: AIME2025 (Math), GPQA Diamond (Science), Knights & Knaves (Logic), and LiveCodeBench (Code). For each task, we compare three models: the **Base Model**, the task-specific **Single-Task RL** model trained on corresponding task, and the **Naive Parallel-RL** model.

To reduce variance associated with small sample sizes, we employ an unbiased estimation method (Chen et al., 2021; Yue et al., 2025). We do not strictly sample k times. Instead, we generate a larger pool of n samples ($n \geq k$) and compute the unbiased estimator:

$$\text{pass@}k = 1 - \frac{\binom{n-c}{k}}{\binom{n}{k}}$$

where c denotes the number of correct samples among the n generations. In our experiments, we set the total number of samples n to 64, which covers the largest k value evaluated in our benchmarks.

We primarily analyze the pass@k performance for the *DeepSeek-R1-Distill-Qwen-1.5B* and *7B* models trained via GRPO (full-parameter), as well as the 1.5B model trained via LoRA. The results are presented in Figure 7.

As observed in the figures, although the Parallel-RL model may exhibit a performance drop at Pass@1 compared to the task-specific RL models, the Pass@k curves of the two models tend to **converge** as the number of samples increases. This convergence indicates that their reasoning boundaries are nearly identical. This phenomenon provides strong evidence for the **low interference** between RL tasks and the inherent **stability** of the Parallel-RL paradigm; even simple parameter averaging does not disrupt the underlying reasoning mechanisms learned by individual RL modules.

Furthermore, it is notable that for most tasks, the performance gap between Naive Parallel-RL and Single-Task RL becomes negligible at Pass@16. This observation indirectly validates the feasibility of **Adapted Parallel-RL**. Since our adaptation phase utilizes $G = 16$ rollouts per prompt, the minimal gap at Pass@16 suggests that the model's capability is preserved in the merged parameters.

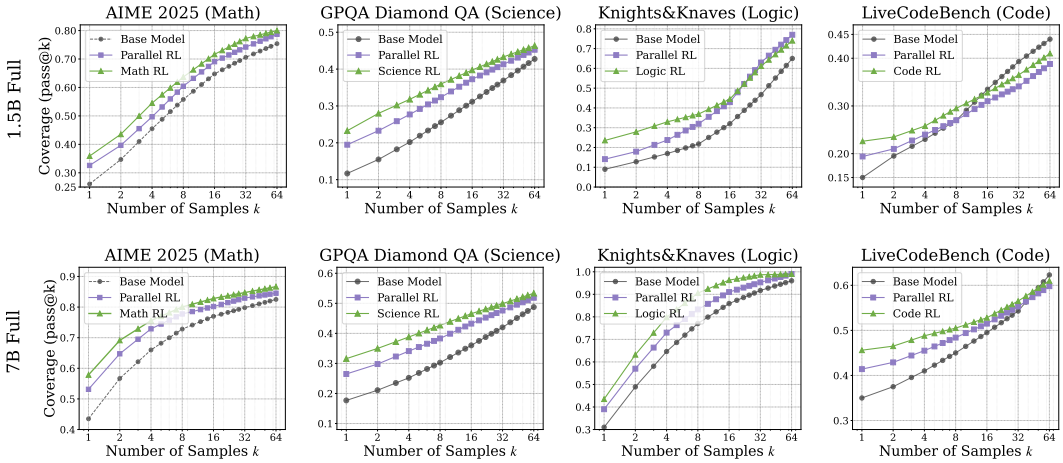

Figure 7: Pass@K Analysis of Naive Parallel RL with full parameter GRPO

Consequently, a small amount of adaptation data (e.g., 5%) is sufficient to realign the policy and fully restore performance, as confirmed in our main experiments (Section 5.2).

F Case Study

In this section, we present several response examples from models trained on different tasks. We also showcase the performance of a multi-task model integrated using a naive parallel RL approach, which involves directly averaging the parameters of individual task models. Specifically, we focus on responses within the math and logic tasks. Our case study includes the base model (DeepSeek-R1-Distill-Qwen-1.5B), the task-specific model (math model and logic model), and the multi-task model (Naive Parallel-RL as described in Section 5.2).

Figure 8,9, 10 and 11 display the responses of these four models to a test sample from MATH500. Figure 12, 13 and 14 illustrate their performance on a "Knights and Knaves" logic question. As observed, the original base model fails to solve either task. While the math and logic models succeed in their respective domains, they produce errors when applied to untrained tasks. In contrast, the Parallel-RL model demonstrates the capability to provide correct answers for both tasks.

Question of MATH500 (Math): Solve the following math problem efficiently and clearly. The last line of your response should be of the following format: 'Therefore, the final answer is: $\boxed{ANSWER}$. I hope it is correct' (without quotes) where ANSWER is just the final number or expression that solves the problem. Think step by step before answering. What is the smallest positive integer n such that all the roots of $z^4 + z^2 + 1 = 0$ are n^{th} roots of unity?

DeepSeek-R1-Distill-Qwen-1.5B (Base Model): **Incorrect**

<think>Okay, so I need to find the smallest positive integer n such that all the roots of the equation $z^4 + z^2 + 1 = 0$ are n-th roots of unity. Hmm, let me think about how to approach this. First, I remember that the roots of unity are complex numbers that satisfy $z^n = 1$ for some integer n. So, each root lies on the unit circle in the complex plane and are equally spaced around it. The equation given is a quartic, which means there are four roots. So, all these roots should lie on the unit circle as well. Let me rewrite the equation: $z^4 + z^2 + 1 = 0$. Hmm, maybe I can factor this equation somehow. Let me see if I can factor it as a quadratic in terms of z^2. Let's set $w = z^2$. Then the equation becomes $w^2 + w + 1 = 0$. That's a quadratic equation in w. Let me solve for w. Using the quadratic formula: $w = [-1 \pm \sqrt{1-4}]/2 = [-1 \pm \sqrt{-3}]/2$. So, w is $(-1 \pm i\sqrt{3})/2$. These are complex numbers, and they are the primitive 3rd roots of unity because they satisfy $w^3 = 1$. Wait, let me check that. Let me compute w^3: Take $w = e^{i2\pi/3}$, which is $(-1 + i\sqrt{3})/2$. Then $w^3 = e^{i2\pi} = 1$. Similarly, the other root is $e^{i4\pi/3}$, so it's also a primitive 3rd root of unity. So, $w = z^2 = e^{i2\pi/3}$ or $e^{i4\pi/3}$. Therefore, to find z, I need to take square roots of these. So, $z^2 = e^{i2\pi/3}$ implies $z = e^{i\pi/3}$ or $z = e^{i5\pi/3}$, because when you square $e^{i\pi/3}$, you get $e^{i2\pi/3}$, and squaring $e^{i5\pi/3}$ gives $e^{i10\pi/3}$ which is the same as $e^{i4\pi/3}$ because $10\pi/3$ is $3\pi + \pi/3$, which is equivalent to $\pi/3$ in terms of the unit circle (since 3π is equivalent to $\pi \pmod{2\pi}$). Wait, no, that's not right. Let me compute $5\pi/3$ squared: $(5\pi/3)^2 = 25\pi^2/9$. That's more than 2π, so I should subtract 2π to get it within 0 to 2π. $25\pi/9$ is about 2.777π, so subtract 2π (which is $18\pi/9$) to get $7\pi/9$. Wait, that can't be right because $5\pi/3$ is 300 degrees, and squaring it would be 600 degrees, which is $600 - 360 = 240$ degrees, which is $4\pi/3$. So, $z = e^{i4\pi/3}$. Similarly, $z^2 = e^{i4\pi/3}$ implies $z = e^{i2\pi/3}$ or $e^{i8\pi/3}$, but $8\pi/3$ is equivalent to $8\pi/3 - 2\pi = 2\pi/3$. So, the roots are $z = e^{i\pi/3}, e^{i5\pi/3}, e^{i4\pi/3}, e^{i2\pi/3}$. Wait, **[...](omit 3006 tokens)** For example, the 3rd roots of unity have order 3, but perhaps in this case, the equation might have roots that are of higher order, but in this case, they are 3rd roots. So, the minimal n is 3. So, the roots of the original equation are the square roots of w, so four roots in total. **Final Answer** The smallest positive integer n is $\boxed{3}$. </think> To solve the problem of finding the smallest positive integer n such that all the roots of the equation $z^4 + z^2 + 1 = 0$ are n-th roots of unity, we start by factoring the given polynomial. First, we recognize that the equation can be rewritten by substituting $w = z^2$, transforming it into a quadratic equation:

$$w^2 + w + 1 = 0$$

Solving this quadratic equation using the quadratic formula, we find the roots:

$$w = \frac{-1 \pm \sqrt{-3}}{2} = \frac{-1 \pm i\sqrt{3}}{2}$$

These roots are the primitive 3rd roots of unity, specifically $e^{i2\pi/3}$ and $e^{i4\pi/3}$. Next, we find the roots of the original equation by solving $z^2 = e^{i2\pi/3}$ and $z^2 = e^{i4\pi/3}$. The solutions are:

$$z = e^{i\pi/3}, e^{i5\pi/3}, e^{i2\pi/3}, e^{i4\pi/3}$$

These roots are all 3rd roots of unity, meaning they satisfy $z^3 = 1$. Since none of these roots are 1st or 2nd roots of unity, the smallest positive integer n such that all roots are n-th roots of unity is 3. Thus, the smallest positive integer n is $\boxed{3}$. **(3707 tokens)**

Figure 8: Response from the base model (DeepSeek-R1-Distill-Qwen-1.5B) on a sample problem from MATH500. As shown, the base model fails to solve the problem.

Question of MATH500 (Math): Solve the following math problem efficiently and clearly. The last line of your response should be of the following format: 'Therefore, the final answer is: $\boxed{ANSWER}$. I hope it is correct' (without quotes) where ANSWER is just the final number or expression that solves the problem. Think step by step before answering. What is the smallest positive integer n such that all the roots of $z^4 + z^2 + 1 = 0$ are n^{th} roots of unity?

Logic Model: **Incorrect**

<think>Okay, so I've got this problem here: I need to find the smallest positive integer n such that all the roots of the equation $z^4 + z^2 + 1 = 0$ are n-th roots of unity. Hmm, I'm a bit rusty on roots of unity, but let me try to recall. First, I remember that the n-th roots of unity are the solutions to the equation $z^n = 1$. They are evenly spaced around the unit circle in the complex plane. The general form is $z = e^{2\pi ik/n}$ where k is an integer from 0 to $n-1$. The equation given is $z^4 + z^2 + 1 = 0$. That's a quartic equation, and I think it can be factored somehow. Maybe it's a quadratic in terms of z^2. Let me try substituting $w = z^2$. Then the equation becomes $w^2 + w + 1 = 0$. Oh, right! That's a quadratic equation. Let me solve for w. Using the quadratic formula: $w = [-1 \pm \sqrt{1 - 4 \cdot 1 \cdot 1}]/2$. That gives me $w = [-1 \pm \sqrt{-3}]/2$. So, the solutions are $w = (-1 + i\sqrt{3})/2$ and $w = (-1 - i\sqrt{3})/2$. These are complex numbers, and I think they can be expressed in polar form. Let me calculate their modulus and argument. The modulus for both is $\sqrt{(-1/2)^2 + (\sqrt{3}/2)^2} = \sqrt{1/4 + 3/4} = \sqrt{1} = 1$. So, both are on the unit circle. What about their arguments? For $w = (-1 + i\sqrt{3})/2$, I think that's a complex number in the second quadrant. The angle is $120°$ because $\cos(120°) = -1/2$ and $\sin(120°) = \sqrt{3}/2$. Similarly, the other solution is $240°$ because $\cos(240°) = -1/2$ and $\sin(240°) = -\sqrt{3}/2$. So, in exponential form, these are $e^{2\pi i \cdot 2/3}$ and $e^{2\pi i \cdot 4/3}$. So $w = e^{2\pi i \cdot 2/3}$ and $w = e^{2\pi i \cdot 4/3}$. But since $w = z^2$, we can write $z^2 = e^{2\pi i \cdot 2/3}$ and $z^2 = e^{2\pi i \cdot 4/3}$. To solve for z, we take the square roots of each. So for each w, z will be the square roots. Let's compute them. Starting with $w_1 = e^{2\pi i \cdot 2/3}$. The square roots would be $e^{\pi i \cdot 2/3 + \pi i k}$ for $k = 0, 1$. So, for $k = 0$: $e^{2\pi i \cdot 2/3 \cdot 1/2} = e^{\pi i \cdot 2/3} = e^{i \cdot 120°}$. Similarly, for $k = 1$: $e^{\pi i \cdot 2/3 + \pi i} = e^{i \cdot 2\pi/3 + i\pi} = e^{i(2\pi/3 + \pi)} = e^{i \cdot 5\pi/3}$. Similarly, for $w_2 = e^{2\pi i \cdot 4/3}$. The square roots would be $e^{\pi i \cdot 4/3 + \pi i k}$. For $k = 0$: $e^{4\pi i/3}$ which is $e^{i \cdot 240°}$. For $k = 1$: $e^{4\pi i/3 + \pi i} = e^{7\pi i/3}$ which is equivalent to $e^{i\pi/3}$ because $7\pi/3$ is more than 2π, so subtract 2π gives $\pi/3$. So, the four roots of the equation $z^4 + z^2 + 1 = 0$ are $z = e^{i\pi/3}$, $e^{i \cdot 2\pi/3}$, $e^{i \cdot 4\pi/3}$, and $e^{i \cdot 5\pi/3}$. Wait, let me make sure. So each of these roots is a 12th root of unity? Because $e^{i\pi/3}$ is the same as $e^{i \cdot 4\pi/3 \cdot 2}$, right? Hmm, no, wait. Let me think. Each of these roots is of the form $e^{i \cdot 2\pi k/12}$ because $2\pi/12 = \pi/6$. So, $e^{i \cdot 2\pi k/12} = e^{i \cdot \pi k/6}$. Let me check: for $k = 1$, it's $\pi/6$; $k = 2$, $2\pi/6 = \pi/3$; $k = 3$, $\pi/2$; $k = 4$, $2\pi/3$; $k = 5$, $5\pi/6$; $k = 6$, π; $k = 7$, $7\pi/6$; $k = 8$, $4\pi/3$; etc. Looking at our roots, they are $e^{i\pi/3}$ ($k = 2$), $e^{i2\pi/3}$ ($k = 4$), $e^{i4\pi/3}$ ($k = 8$), and $e^{i5\pi/3}$ ($k = 10$). **[...](omit 2458 tokens)** Therefore, the smallest positive integer n is 12. </think> To solve the problem, we start by recognizing that the roots of the equation $z^4 + z^2 + 1 = 0$ can be found by substituting $w = z^2$, transforming the quartic into a quadratic equation $w^2 + w + 1 = 0$. Solving this quadratic using the quadratic formula gives the roots $w = e^{2\pi i \cdot 2/3}$ and $w = e^{2\pi i \cdot 4/3}$. Since $w = z^2$, we solve for z by taking the square roots of each w. This yields four roots: $z = e^{\pm \pi i/3}$ and $z = e^{\pm 5\pi i/3}$. These roots are $e^{\pm \pi i/3}$, $e^{\pm 5\pi i/3}$, which correspond to the 12th roots of unity. To confirm, we factor the quartic as $(z^2 + z + 1)(z^2 - z + 1)$. The roots of $z^2 + z + 1$ are the primitive 3rd roots of unity, and the roots of $z^2 - z + 1$ are the primitive 12th roots of unity. Thus, all roots of the original equation are indeed 12th roots of unity. Since the roots are not all 6th roots of unity (as $e^{\pi i/6}$ is not a root of the quartic), the smallest positive integer n for which all roots are n-th roots of unity is $\boxed{12}$. **(3178 tokens)**

Figure 9: Response from the logic model on a sample problem from MATH500. The logic model is derived from the base model via GRPO training on a logic dataset.

Question of MATH500 (Math): Solve the following math problem efficiently and clearly. The last line of your response should be of the following format: 'Therefore, the final answer is: $\boxed{ANSWER}$. I hope it is correct' (without quotes) where ANSWER is just the final number or expression that solves the problem. Think step by step before answering. What is the smallest positive integer n such that all the roots of $z^4 + z^2 + 1 = 0$ are n^{th} roots of unity?

Math Model: Correct

<think>Okay, so I have this problem: I need to find the smallest positive integer n such that all the roots of the equation $z^4 + z^2 + 1 = 0$ are n-th roots of unity. Hmm, let me try to figure this out step by step. First, I remember that roots of unity are complex numbers that satisfy $z^n = 1$ for some integer n. So, each root is $e^{2\pi ik/n}$ where k is an integer from 0 to $n-1$. These roots are equally spaced around the unit circle in the complex plane. Now, the equation given is $z^4 + z^2 + 1 = 0$. I wonder if I can factor this or simplify it somehow. Let me see. Maybe substituting $w = z^2$ would make it easier. Let's try that. If I let $w = z^2$, then the equation becomes $w^2 + w + 1 = 0$. That's a quadratic equation in terms of w. I can solve this using the quadratic formula. The solutions for w would be: $w = [-1 \pm \sqrt{1 - 4 \cdot 1 \cdot 1}]/2 = [-1 \pm \sqrt{-3}]/2 = (-1 \pm i\sqrt{3})/2$. So, the solutions for w are $(-1 + i\sqrt{3})/2$ and $(-1 - i\sqrt{3})/2$. These are complex numbers. I recognize them as the complex cube roots of unity, except that the cube roots of unity are $(1, \omega, \omega^2)$ where $\omega = e^{2\pi i/3}$. Wait, but here the solutions are actually the primitive cube roots of unity, because $(-1 \pm i\sqrt{3})/2$ are the non-real cube roots. So, they satisfy $w^3 = 1$, but aren't equal to 1 themselves. Wait, actually, let me check. Let me compute $(-1 + i\sqrt{3})/2$. Let me denote this as ω. Let's compute ω^3: $\omega = e^{2\pi i/3} = \cos(2\pi/3) + i\sin(2\pi/3) = -1/2 + i\sqrt{3}/2$. So, yes, ω is a primitive cube root of unity, meaning $\omega^3 = 1$. So, $w = \omega$ and $w = \omega^2$, where ω^2 is the other cube root. So, $w^2 + w + 1 = 0$, so the solutions are the primitive cube roots of unity. Therefore, $w = e^{2\pi i/3}$ and $w = e^{4\pi i/3}$. But since $w = z^2$, that means $z^2 = e^{2\pi i/3}$ or $z^2 = e^{4\pi i/3}$. So, to find z, we take the square roots of these. Let me solve $z^2 = e^{2\pi i/3}$. The solutions are $z = e^{\pi i/3}$ and $z = e^{\pi i/3 + \pi i} = e^{4\pi i/3}$. Similarly, for $z^2 = e^{4\pi i/3}$, the solutions are $z = e^{2\pi i/3}$ and $z = e^{2\pi i/3 + \pi i} = e^{5\pi i/3}$. So, the roots of the equation $z^4 + z^2 + 1 = 0$ are $z = e^{\pi i/3}, e^{4\pi i/3}, e^{2\pi i/3}$, and $e^{5\pi i/3}$. **[...](omit 10051 tokens)** They are 6th roots, so the minimal n is 6. So, the final answer is 6. </think> **Final Answer**

The smallest positive integer n is $\boxed{6}$. To solve the problem of finding the smallest positive integer n such that all the roots of $z^4 + z^2 + 1 = 0$ are n-th roots of unity, we start by solving the given equation. First, we substitute $w = z^2$ to transform the equation into a quadratic equation:

$$w^2 + w + 1 = 0$$

Solving this quadratic equation using the quadratic formula, we get:

$$w = \frac{-1 \pm \sqrt{-3}}{2} = \frac{-1 \pm i\sqrt{3}}{2}$$

These solutions are the primitive cube roots of unity, denoted as ω and ω^2, where $\omega = e^{2\pi i/3}$ and $\omega^2 = e^{4\pi i/3}$. Next, we solve for z by taking the square roots of ω and ω^2. The square roots of ω are $e^{\pi i/3}$ and $e^{4\pi i/3}$, while the square roots of ω^2 are $e^{2\pi i/3}$ and $e^{5\pi i/3}$. Thus, the roots of the equation $z^4 + z^2 + 1 = 0$ are:

$$z = e^{\pi i/3}, e^{2\pi i/3}, e^{4\pi i/3}, e^{5\pi i/3}$$

These roots are all 6th roots of unity because each root can be expressed as $e^{k\pi i/3}$ where k is an integer, and the smallest n such that all these roots are n-th roots of unity is 6. Therefore, the smallest positive integer n such that all the roots of $z^4 + z^2 + 1 = 0$ are n-th roots of unity is: $\boxed{6}$ **(10667 tokens)**

Figure 10: Response from the math model on a sample problem from MATH500. The math model is derived from the base model via GRPO training on a math dataset.

Question of MATH500 (Math): Solve the following math problem efficiently and clearly. The last line of your response should be of the following format: 'Therefore, the final answer is: $\boxed{ANSWER}$. I hope it is correct' (without quotes) where ANSWER is just the final number or expression that solves the problem. Think step by step before answering. What is the smallest positive integer n such that all the roots of $z^4 + z^2 + 1 = 0$ are n^{th} roots of unity?

Parallel-RL Model: Correct

<think>Okay, so I've got this problem here: I need to find the smallest positive integer n such that all the roots of the equation $z^4 + z^2 + 1 = 0$ are n-th roots of unity. Hmm, okay. Let's see. First, I remember that roots of unity are complex numbers that satisfy the equation $z^n = 1$. So, they lie on the unit circle in the complex plane and are equally spaced around the circle. The roots of unity are also the solutions to $z^n = 1$. The given equation is $z^4 + z^2 + 1 = 0$. I need to figure out the roots of this equation and then determine the smallest n for which all these roots are n-th roots of unity. Let me try to factor the equation or find its roots. Maybe I can factor it as a quadratic in terms of z^2. Let me set $w = z^2$, so the equation becomes $w^2 + w + 1 = 0$. That's a quadratic equation, which I can solve using the quadratic formula. The quadratic formula says that for $ax^2 + bx + c = 0$, the roots are $(-b \pm \sqrt{b^2 - 4ac})/(2a)$. Applying that here, $a = 1$, $b = 1$, $c = 1$. So the roots are $(-1 \pm \sqrt{1 - 4})/2 = (-1 \pm \sqrt{-3})/2$. So, $w = (-1 \pm i\sqrt{3})/2$. Therefore, $z^2 = (-1 \pm i\sqrt{3})/2$. So, to find z, I need to take the square roots of these two complex numbers. Wait, but I remember that $(-1 + i\sqrt{3})/2$ is actually a complex number on the unit circle. Let me check its modulus. The modulus of a complex number $a + ib$ is $\sqrt{a^2 + b^2}$. So here, $a = -1/2$ and $b = (\sqrt{3})/2$. So modulus squared is $(1/4) + (3/4) = 1$, so modulus is 1. So, it's indeed on the unit circle. Similarly, $(-1 - i\sqrt{3})/2$ also has modulus 1. So, these are both complex numbers with modulus 1, which suggests they are points on the unit circle at angles of 120 degrees and 240 degrees, or $2\pi/3$ and $4\pi/3$ radians, respectively. So, if I can express these roots in exponential form, that might help. Since they're on the unit circle, they can be written as $e^{i\theta}$, where θ is the angle. So, for $(-1 + i\sqrt{3})/2$, θ is $2\pi/3$, and for $(-1 - i\sqrt{3})/2$, θ is $4\pi/3$. Therefore, the solutions for z^2 are $e^{i2\pi/3}$ and $e^{i4\pi/3}$. So, taking square roots, the solutions for z would be the square roots of these. Wait, but when you take the square root of $e^{i\theta}$, you get $e^{i\theta/2}$ and $e^{i(\theta/2+\pi)}$. So, let's compute that. For the first root, $z^2 = e^{i2\pi/3}$, so $z = e^{i\pi/3}$ and $e^{i(\pi/3+\pi)} = e^{i4\pi/3}$. Similarly, for the second root, $z^2 = e^{i4\pi/3}$, so $z = e^{i2\pi/3}$ and $e^{i(2\pi/3+\pi)} = e^{i5\pi/3}$. Wait, **[...](omit 5291 tokens)** Therefore, the smallest positive integer n is 6. **Final Answer** The smallest positive integer n is $\boxed{6}$.</think> To find the smallest positive integer n such that all the roots of $z^4 + z^2 + 1 = 0$ are n-th roots of unity, we start by factoring the given polynomial. First, we substitute $w = z^2$ to transform the equation into a quadratic:

$$w^2 + w + 1 = 0$$

Solving this quadratic equation using the quadratic formula, we get:

$$w = \frac{-1 \pm \sqrt{-3}}{2} = \frac{-1 \pm i\sqrt{3}}{2}$$

These roots are complex numbers on the unit circle, specifically: $w = e^{i2\pi/3}$ and $w = e^{i4\pi/3}$
Next, we find the roots of the original equation by solving for z:
1. For $z^2 = e^{i2\pi/3}$, the roots are: $z = e^{i\pi/3}$ and $z = e^{i4\pi/3}$
2. For $z^2 = e^{i4\pi/3}$, the roots are: $z = e^{i2\pi/3}$ and $z = e^{i5\pi/3}$
These roots are the 6th roots of unity, as they are at angles $\pi/3, 2\pi/3, 4\pi/3$, and $5\pi/3$. These angles are all multiples of $\pi/3$, which are angles of the 6th roots of unity. The 6th roots of unity are $e^{i2\pi k/6}$ for $k = 0, 1, 2, 3, 4, 5$. The roots of the given polynomial are a subset of these, specifically excluding 1 and -1. Therefore, all roots are 6th roots of unity. Thus, the smallest positive integer n such that all the roots are n-th roots of unity is: $\boxed{6}$ **(6080 tokens)**

Figure 11: Response from the Parallel-RL model on a sample problem from MATH500. This model uses **Naive Parallel-RL** (shown in Section 5.2), which is obtained by directly averaging the models trained on **four tasks**.

Question of KK (Logic): A very special island is inhabited only by knights and knaves. Knights always tell the truth, and knaves always lie. You meet 2 inhabitants: Owen, and Penelope. Owen asserted: "Penelope is a knight if and only if Penelope is a knave". Penelope noted, "Owen is not a knave". So who is a knight and who is a knave?

DeepSeek-R1-Distill-Qwen-1.5B (Base Model): **Incorrect**

<think>Okay, so I have this logic puzzle about knights and knaves. Knights always tell the truth, and knaves always lie. I need to figure out who is who between Owen and Penelope based on their statements.First, let's break down Owen's statement: "Penelope is a knight if and only if Penelope is a knave." The "if and only if" part is a bit tricky because it means both parts of the statement need to be true for the whole statement to be true. So, either Penelope is a knight and also a knave, or she's neither. But wait, knights are always truthful, and knaves always lie. So Penelope can't be both a knight and a knave at the same time. That makes Penelope's statement a bit confusing.Now, Penelope's statement is: "Owen is not a knave." Since knights tell the truth and knaves lie, if Penelope is a knight, her statement must be true, meaning Owen is not a knave, so Owen must be a knight too. But if Penelope is a knave, her statement is false, which would mean that Owen is a knave. However, if Owen is a knave, then Penelope's statement is a lie, which aligns with Penelope being a knave. Wait, but if Owen is a knight, then his statement must be true. So if Penelope is also a knight, her statement "Owen is not a knave" would be true because Owen is indeed a knight. That seems consistent. Alternatively, if Penelope is a knave, her statement would be false, meaning Owen is a knave. But if Owen is a knave, then his statement "Penelope is a knight if and only if Penelope is a knave" would be false because Penelope is a knave and a knight at the same time, which isn't possible. So that would lead to a contradiction.Therefore, the only consistent scenario is that both Owen and Penelope are knights. That way, Penelope's statement is true, and Owen's statement holds because he's a knight.</think> Penelope and Owen are both knights. This is because Penelope's statement that Owen is not a knave is true, and Owen's statement that Penelope is a knight if and only if Penelope is a knave is also true since both are knights. <answer> (1) Owen is a knight. (2) Penelope is a knight. </answer> **(523 tokens)**

Logic Model: **Correct**

<think> Okay, so I'm trying to solve this logic puzzle where there are two inhabitants on an island: Owen and Penelope. They're either knights, who always tell the truth, or knaves, who always lie. I need to figure out who is who based on what they said. First, let's break down what each statement means. Owen said, "Penelope is a knight if and only if Penelope is a knave." That's a bit of a mouthful, but let's parse it. The "if and only if" means that both parts of the statement are connected by an equivalence—so if one is true, the other must be true, and if one is false, the other must be false. So, if Penelope is a knight, she tells the truth, so her statement must be true. That would mean Penelope is both a knight and a knave, which doesn't make sense because knights and knaves are distinct categories. Wait, that's a problem. So if Penelope is a knight, her statement that "Penelope is a knight if and only if Penelope is a knave" would have to be true. But that would require Penelope to be both a knight and a knave, which is impossible. Therefore, Penelope cannot be a knight. That means Penelope must be a knave, and since knaves lie, her statement is false. Now, let's look at Penelope's statement: "Owen is not a knave." Since Penelope is a knave, her statement is false. So the opposite of "Owen is not a knave" must be true. The opposite of "not a knave" is "knave," so Penelope's statement being false means "Owen is a knave." Therefore, Owen is a knave. To recap: - If Penelope is a knave, her statement "Owen is not a knave" is false, meaning Owen is a knave. - Since Owen is a knave, his statement "Penelope is a knight if and only if Penelope is a knave" is false, which fits because knaves always lie. So, both Penelope and Owen are knaves. </think> Penelope is a knave, and Owen is also a knave. <answer> (1)Penelope is a knave. (2) Owen is a knave. </answer> **(505 tokens)**

Figure 12: Comparison of responses from the base model (DeepSeek-R1-Distill-Qwen-1.5B) and the logic model on a sample problem from Knights and Knaves. The logic model is derived from the base model via GRPO training on a logic dataset. As shown, the base model fails to solve the problem, while the logic model provides the correct answer.

Question of KK (Logic): A very special island is inhabited only by knights and knaves. Knights always tell the truth, and knaves always lie. You meet 2 inhabitants: Owen, and Penelope. Owen asserted: "Penelope is a knight if and only if Penelope is a knave". Penelope noted, "Owen is not a knave". So who is a knight and who is a knave?

Math Model: **Incorrect**

<think>Okay, so I've got this logic puzzle about knights and knaves. Knights always tell the truth, and knaves always lie. There are two people involved: Owen and Penelope. Owen says, "Penelope is a knight if and only if Penelope is a knave." Penelope responds, "Owen is not a knave." I need to figure out who is a knight and who is a knave. First, I'll try to understand what each statement means. The first statement by Owen uses a biconditional ("if and only if"), which means both parts must be true or both must be false. So, if Penelope is a knight, then Penelope is a knight, and Penelope is a knave. Wait, that can't be right because Penelope can't be both a knight and a knave at the same time. That seems contradictory. On the other hand, if Penelope is a knave, then Penelope is not a knight, and Penelope is a knight. Again, that's a contradiction because a knave can't be both not a knight and a knight simultaneously. Hmm, this is confusing. Maybe I made a mistake in interpreting the statements. Let me break it down step by step. Let's consider the two possibilities: Penelope is a knight or Penelope is a knave. 1. **Case 1: Penelope is a knight.** - Since Penelope is a knight, she tells the truth. Her statement, "Owen is not a knave," means Owen is a knight. But wait, if Owen is a knight, then his statement must be true. His statement is a biconditional: Penelope is a knight if and only if Penelope is a knave. If Penelope is a knight, then the statement would be "Knight if and only if Knave," which is a contradiction because a knight cannot be a knave. Therefore, this case leads to a contradiction. 2. **[...](omit 257 tokens)** So, there's a problem here. I think I messed up the logical analysis. Let me try again. If Penelope is a knave, her statement is false. Her statement is "Owen is not a knave." The negation of "Owen is not a knave" is "Owen is a knave." So, Owen is a knave. Now, Owen's statement is "Penelope is a knight if and only if Penelope is a knave." Both are knaves, so the biconditional is true. But since Owen is a knave, his statement must be false. That can't happen because a true statement can't be a lie. Therefore, the only consistent scenario is that Penelope is a knight and Owen is a knave. Wait, but earlier that led to a contradiction. Let me think again. If Penelope is a knight, her statement is true. Her statement is "Owen is a knight if and only if Penelope is a knave." Since Penelope is a knight, the statement is "Owen is a knight if and only if Penelope is a knave," which is "Owen is a knight if and only if not a knight." That's a contradiction because a knight can't be a knave. So Penelope can't be a knight. Therefore, Penelope must be a knave, and Owen must be a knight. But then Owen, being a knight, tells the truth about Penelope's status. His statement is "Penelope is a knight if and only if Penelope is a knave," which is a biconditional. Since Penelope is a knave, the statement is false because Penelope is a knave but not a knight. But as a knight, Owen must tell the truth, so his statement must be true. This is a contradiction. This is confusing. Maybe there's no solution, but logically, Penelope must be a knave because her statement leads to a contradiction if she were a knight. So the only consistent solution is that Penelope is a knave, and Owen is also a knave, but that leads to a contradiction in Owen's statement. Therefore, the only solution is Penelope is a knave, and Owen is a knight, but that contradicts Owen's truthful statement. I think I need to conclude that Penelope is a knave, and Owen is a knight, but that leads to a contradiction. Alternatively, Penelope is a knight, but that leads to a contradiction. Therefore, there must be a mistake in my reasoning. Wait, perhaps if Penelope is a knight, her statement is true, which would mean both Penelope and Owen are knaves, but that contradicts. If Penelope is a knave, her statement is false, meaning Penelope is a knight, which is also a contradiction. Therefore, there must be an error in the problem setup, but logically, Penelope must be a knave, and Owen is a knight, but that contradicts. Alternatively, Penelope is a knight, and Owen is a knave, but that also contradicts. This is perplexing.</think> <answer> Penelope is a knave, and Owen is a knight. </answer> **(1399 tokens)**

Figure 13: Response from the math model on a sample problem from Knights and Knaves. The math model is derived from the base model via GRPO training on a math dataset.

Question of KK (Logic): A very special island is inhabited only by knights and knaves. Knights always tell the truth, and knaves always lie. You meet 2 inhabitants: Owen, and Penelope. Owen asserted: "Penelope is a knight if and only if Penelope is a knave". Penelope noted, "Owen is not a knave". So who is a knight and who is a knave?

Parallel-RL Model: Correct

<think>Okay, so I've got this logic puzzle involving knights and knaves. Knights always tell the truth, and knaves always lie. I need to figure out who is who between Owen and Penelope. Let's break it down step by step. First, Owen made a statement: "Penelope is a knight if and only if Penelope is a knave." That's a bit of a mouthful. The "if and only if" part means that both parts of the statement are connected by a biconditional, so they must be both true or both false for the statement to hold. Penelope then said, "Owen is not a knave." So Owen is claiming that Owen isn't a knave. Since knaves always lie, if Penelope is a knave, her statement would be false, meaning Owen is a knave. But wait, if Penelope is a knave, that would mean Owen is a knave, which would make Penelope's statement false, which aligns with knaves always lying. Hmm, maybe I should approach this more systematically. Let's consider the two possibilities: Penelope is a knight or a knave. 1. **Case 1: Penelope is a knight.** - Since knights tell the truth, Penelope's statement "Owen is not a knave" must be true. So, Owen is not a knave. That means Owen is a knight. - Now, let's look at Owen's statement. He said "Penelope is a knight if and only if Penelope is a knave." If both parts are true, that would mean Penelope is both a knight and a knave, which isn't possible because each inhabitant is either a knight or a knave, not both. - Alternatively, if both parts are false, then Penelope is neither a knight nor a knave, which also contradicts the fact that each inhabitant is either one or the other. - Therefore, Penelope being a knight leads to a contradiction because Owen's statement can't be satisfied without a contradiction. So this case can't be true. 2. **Case 2: Penelope is a knave.** - As knaves always lie, Penelope's statement "Owen is not a knave" is false. So, the opposite is true: Owen is a knave. - Now, let's check Owen's statement again. He said "Penelope is a knight if and only if Penelope is a knave." If both parts are true, Penelope would have to be both a knight and a knave, which isn't allowed. But since Penelope is a knave, her statement is false, so the biconditional is false. That means either Penelope is a knight and a knave (which is impossible), or Penelope is neither a knight nor a knave, which also isn't possible. Wait, that doesn't seem right. Let me clarify: For a biconditional to be false, at least one part must be false. So, either Penelope is not a knight or Penelope is not a knave. Since we've established Penelope is a knave, the second part is false. Therefore, the biconditional is false, which is consistent with Owen being a knave, who always lies. So this case holds without contradiction. So, putting it all together: - Penelope is a knave.- Owen is also a knave because Penelope's lie makes him one. </think> <answer> (1) Penelope is a knave. (2) Owen is a knave. </answer> **(756 tokens)**

Figure 14: Response from the Parallel-RL model on a sample problem from Knights and Knaves. This model uses **Naive Parallel-RL** (shown in Section 5.2), which is obtained by directly averaging the models trained on **four tasks**.

G PROMPT TEMPLATES

In this section, we provide the specific prompt templates used for both training and evaluation across the four representative tasks discussed in the main text.

Training Prompts. The prompt templates used for training the Math and Science tasks are presented in Figure 15. For the Logic task, we follow the configuration established in Logic-RL, with the specific prompt shown in Figure 20. Finally, for the Code task, we adopt the prompt template displayed in Figure 21, which aligns with the settings used in DeepCoder.

Evaluation Prompts. For evaluation, we utilize the Lighteval framework and largely adhere to the prompt templates provided in its official repository. For Math benchmarks, Figure 16 and Figure 17 illustrate the templates used for evaluating AIME and MATH500, respectively, following the Lighteval standard. For Science benchmarks, the prompt templates for the MMLU and GPQA are shown in Figure 19 and Figure 18, also consistent with the Lighteval repository. For Logic and Code tasks, the evaluation prompts are the same as those used during training.

Prompt for Math and Science Training

<|im_start|>system
You are a helpful assistant.<|im_end|>
<|im_start|>user
{**question**}
Let's think step by step and output the final answer within \\boxed{{}}.<|im_end|>
<|im_start|>assistant

Figure 15: Prompt for **Math and Science Task**. We follow the prompt template settings in deepscaler.

Prompt for AIME Evaluation

<|im_start|>system
You are a helpful assistant.<|im_end|>
<|im_start|>user
{**question**}
Solve the following math problem efficiently and clearly. The last line of your response should be of the following format: 'Therefore, the final answer is: $\\boxed{{ANSWER}}$. I hope it is correct' (without quotes) where ANSWER is just the final number or expression that solves the problem.
Think step by step before answering.<|im_end|>
<|im_start|>assistant

Figure 16: Prompt for **AIME Evaluation**. We follow the prompt template settings in Lighteval.

Prompt for MATH500 Evaluation

<|im_start|>system
You are a helpful assistant.<|im_end|>
<|im_start|>user
{**question**}
Solve the following problem. The final line of your response MUST be of the following format:
"ANSWER: $ANSWER" (without quotes) where $ANSWER is the final answer. Think step by step before answering.<|im_end|>
<|im_start|>assistant

Figure 17: Prompt for **MATH500 Evaluation**. We follow the prompt template settings in Lighteval.

Prompt for GPQA Evaluation

<|im_start|>system
You are a helpful assistant.<|im_end|>
<|im_start|>user
Answer the following multiple choice question. The last line of your response should be of the following format: 'Answer: $LETTER' (without quotes) where LETTER is one of ABCD. Think step by step before answering.
{**Question**}
A) {A}
B) {B}
C) {C}
D) {D}. <|im_end|>
<|im_start|>assistant

Figure 18: Prompt for **GPQA Evaluation**. We follow the prompt template settings in Lighteval.

Prompt for MMLU Evaluation

<|im_start|>system
You are a helpful assistant.<|im_end|>
<|im_start|>user
The following are multiple choice questions (with answers) about {subject}.
Question: {**question**}
A. {choice_A}
B. {choice_B}
C. {choice_C}
D. {choice_D}
Answer: <|im_end|>
<|im_start|>assistant

Figure 19: Prompt for **MMLU Evaluation**. We follow the prompt template settings in Lighteval.

Prompt for Logic Task (Knights&Knaves)

<|im_start|>system
You are a helpful assistant. The assistant first thinks about the reasoning process in the mind and then provides the user with the answer. The reasoning process and answer are enclosed within `` and<answer> </answer> tags, respectively, i.e., ``<answer> answer here </answer>. Now the user asks you to solve a logical reasoning problem. After thinking, when you finally reach a conclusion, clearly state the identity of each character within <answer> </answer> tags. i.e., <answer> (1) Zoey is a knight
(2) ... </answer>.<|im_end|>
<|im_start|>user
{**question**}
<|im_end|>
<|im_start|>assistant

Figure 20: Prompt for **Logic Task**. We follow the prompt template settings in LogicRL.

Prompt for Code Task (LiveCodeBench)

<|im_start|>system
You are a helpful assistant.<|im_end|>
<|im_start|>user
You will be given a question (problem specification) and will generate a correct Python program that matches the specification and passes all tests.

Question: {**question_content**}

{ if question.starter_code }
You will use the following starter code to write the solution to the problem and enclose your code within delimiters.
```python
{**starter_code**}
```

{ else }
Format:
Read the inputs from stdin solve the problem and write the answer to stdout (do not directly test on the sample inputs). Enclose your code within delimiters as follows. Ensure that when the python program runs, it reads the inputs, runs the algorithm and writes output to STDOUT.
```python
# YOUR CODE HERE
```

Answer: (use the provided format with backticks)<|im_end|>

<|im_start|>assistant
```python

Figure 21: Prompt for **Code Task**. We follow the prompt template settings in LiveCodeBench.
```