Understanding and Mitigating Cross-Domain Interference in RL Post-Training

Anonymous ACL submission

Abstract

Reinforcement learning (RL) has played a central role in recent advances in large reasoning models (LRMs), yielding strong domain-specific gains in both verifiable and open-ended reasoning. However, training a single LRM across diverse domains is challenging due to pronounced domain heterogeneity. Through a systematic study of two widely used strategies, SEQUENTIAL RL and MIXED RL, we find that both suffer from substantial cross-domain interference at the behavioral and gradient levels, resulting in limited overall gains. To address these challenges, we introduce **M**odular **G**radient **S**urgery (**MGS**), which resolves gradient conflicts at the module level within the transformer. When applied to Llama and Qwen models, MGS achieves average improvements of 4.3 (16.6%) and 4.5 (11.1%) points, respectively, over standard multi-task RL across three representative domains (math, general chat, and instruction following). Further analysis demonstrates that MGS remains effective under prolonged training. Overall, our study clarifies the sources of cross-domain interference in multi-domain RL and provides an effective module-wise approach for mitigating it.

1 Introduction

Building large reasoning models (LRMs) with long chain-of-thought capabilities has become a central goal in recent language model development (Liu et al., 2025a; Jaech et al., 2024; Zhang et al., 2025c). Reinforcement learning (RL) has emerged as a key driver of this progress. In particular, reinforcement learning with verifiable rewards (RLVR) has demonstrated substantial gains in domains with objective correctness signals, such as mathematics (Yu et al., 2025; Liu et al., 2025e) and code (Wang et al., 2024; Zhoubian et al., 2025), where RL not only improves final accuracy but also incentivizes advanced reasoning such as branching and reflection (Mondorf and Plank, 2024; Zeng et al., 2025a; Xu et al., 2025). Motivated by these successes, recent efforts have extended RL beyond strictly verifiable settings, applying RL to more general tasks such as instruction following (Gunjal et al., 2025; Chang et al., 2025) and open-ended chat (Bhaskar et al., 2025; Li et al., 2025a) using model-based rewards.

To train a reasoning model that performs well across diverse domains, one needs to combine heterogeneous rewards and domains (e.g., rule-based rewards for math, preference-based rewards for chat), which is challenging. Practitioners consistently observe difficulties when mixing multiple domains these objectives: learning on one domain often degrades performance on another, transfer across domains is weak, and naively mixing rewards can lead to unstable or brittle training dynamics (Li et al., 2025a; Bhaskar et al., 2025; Wang et al., 2025a).

In this work, we systematically study how to perform RL on diverse domains to build broadly capable reasoning models. As illustrated in Figure 1, we begin by analyzing two natural and widely used strategies: (1) SEQUENTIAL RL, where a model is trained on different domains one after another; and (2) MIXED (multi-task) RL, where data from multiple domains are combined within each training batch. Through controlled experiments on representative domains—math reasoning, general chat, and instruction following—we show that both strategies suffer from limitations in multi-domain settings. SEQUENTIAL RL exhibits mode interference, where optimizing strongly for one domain erases or constrains behaviors needed for others, while MIXED RL suffers from gradient conflicts between domains. As a result, both approaches exhibit a substantial gap compared to single-domain experts, as shown in Figure 1.

To address these limitations, we propose **M**odular **G**radient **S**urgery (**MGS**), an effective method for mitigating cross-domain interference

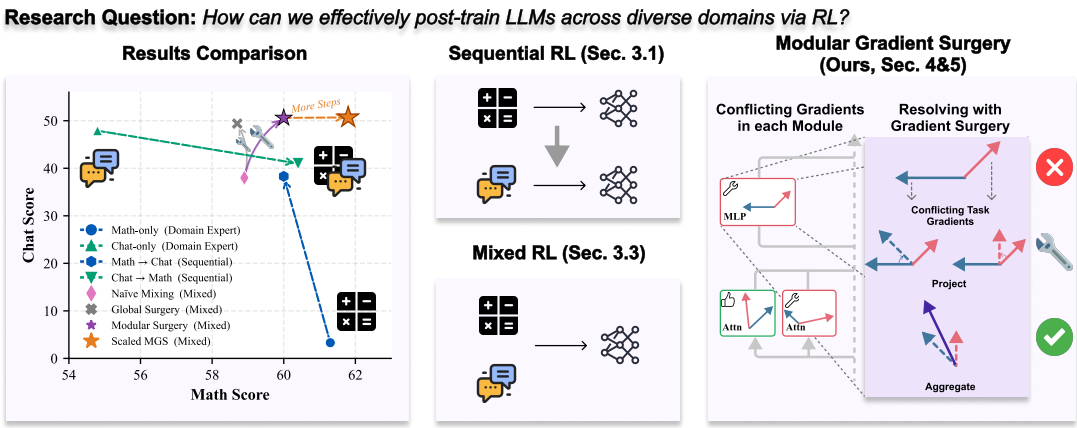

Figure 1: Effectiveness of different ways for training reasoning models on multiple domains. Naive strategies, such as sequential RL training (SEQUENTIAL RL), or mixing different domains in the same batch (MIXED RL), often result in limited performance across domains. We propose Modular Gradient Surgery (MGS), which resolves conflicting gradients at the module level and achieves a more balanced multi-domain performance.

in MIXED RL training. Pretrained transformers are highly modular, with different components specializing in distinct functionalities, and gradient conflicts are often localized to specific modules rather than uniformly distributed across the model. MGS resolves conflicting gradients at the module level using gradient surgery (Yu et al., 2020), allowing compatible updates to be preserved while selectively mitigating destructive interference.

Empirically, we show that MGS consistently improves MIXED RL training across model families and task combinations. Applied to Llama and Qwen models over *Math*, *Chat*, and Instruction-Following (*IF*) domains, MGS substantially expands the capability frontier, achieving stronger and more balanced performance than SEQUENTIAL training, Naïve Mixing, and Global Gradient Surgery. Further analysis demonstrates that MGS scales with extended training, yielding an additional 3% relative improvement on the Math benchmark. Furthermore, MGS generalizes well to additional tasks, achieving a 19.4% relative average improvement on `Llama-3.1-8B`. These gains are achieved with negligible overhead with parallel training frameworks like FSDP.

Overall, our findings characterize two key sources of cross-domain interference in multi-domain RL post-training: mode interference in SEQUENTIAL training and gradient conflicts in MIXED training. Based on this analysis, we introduce a module-wise gradient surgery approach to mitigate such interference. To summarize, our contributions include: (1) A systematic study of multi-domain RL for reasoning, revealing the limitations of SEQUENTIAL and MIXED training strategies; (2) A new approach, Modular Gradient Surgery, which resolves cross-domain interference at modular levels; and (3) An extensive evaluation across models and domains, establishing strong and consistent empirical gains of MGS.

2 Background and Experimental Setup

Research question. We study the question posed in Fig. 1: *How can we effectively post-train LLMs across diverse domains via RL?* Specifically, we analyze cross-domain interference in multi-domain RL post-training and investigate how to mitigate it. Previous works have shown strong success in training reasoning models with a single type of reward over a set of relatively similar domains. Notably, reinforcement learning with verifiable rewards (RLVR) trains reasoning models using rule-based rewards in domains such as mathematics and code (Wen et al., 2025; Zeng et al., 2025a; Seed et al., 2025; Guo et al., 2025a). Reinforcement learning with model-rewarded thinking (RLMT) operates with model-based rewards to optimize open-ended behaviors such as chat and creative writing (Bhaskar et al., 2025). These approaches typically focus on one reward paradigm at a time. In contrast, our work studies RL training that jointly combines different prompts and rewards across domains.

Formally, let π_θ denote reasoning LLM as a policy parameterized by θ, which induces a conditional distribution $\pi(y, z \mid x)$ over responses y and long chain-of-thought z given a prompt x. We consider a set of training domains denoted as

$\mathcal{T} = \{(\mathcal{D}_k, R_k)\}_{k=1}^{K}$, where each domain consists of a tuple of training data $D_k = \{x\}$, and a domain-specific reward function R_k. Given a prompt $x \sim \mathcal{D}_k$ and a response-cot pair (y, z) sampled from the LLM policy $(y, z) \sim \pi_\theta(\cdot \mid x)$, a reward function produces a scalar reward $R(x, y) \in \mathbb{R}$. Our goal is to optimize π_θ to perform well across these domains:

$$\max_{\theta} \mathbb{E}_{(\mathcal{D}_k, R_k)\sim\mathcal{T}} \left[\mathbb{E}_{x\sim\mathcal{D}_k,\,(y,z)\sim\pi_\theta(\cdot|x)} R_k(x, y)\right]. \quad (1)$$

At evaluation time, we assess *in-domain performance* with benchmarks from the training domains. In addition, we are interested in testing general reasoning capabilities beyond in-domain tasks with *out-of-domain* benchmarks.

2.1 Experimental Setup

Tasks and models. We use three representative domains for RL training, including 1) math reasoning (*Math*), 2) general chat (*Chat*), and 3) instruction following (*IF*), and study two model families, `Qwen-2.5-7B` (Yang et al., 2025) and `Llama-3.1-8B` (Dubey et al., 2024). Table 2 summarizes the datasets used in experiments. Specifically, for *Math*, we use `SimpleRL-Hard` for `Qwen-2.5-7B` and `SimpleRL-Easy` for `Llama-3.1-8B` (Zeng et al., 2025a), determined by model capability. Following RLVR (Liu et al., 2025a), we adopt the correctness reward for math: $R_{Math} = \mathbb{1}\{y = y_{ref}\}$, where y_{ref} is the gold answer. For *Chat*, we follow (Bhaskar et al., 2025) and use `Wildchat` as the training data. We provide scalar rewards $R_{Chat} = r(x, y)$ where r is the reward model. We use the open-source reward model `Skywork-Reward-V2-Llama-3.1-8B` (Liu et al., 2025b). For *IF*, we construct our training data (see Appendix D) based on `Nemotron` (Liu et al., 2025c), and use the scoring utilities in IFEval (Zhou et al., 2023) as our reward function: $R_{IF}(x, y) = \frac{1}{|\mathcal{C}_x|}\sum_{c\in\mathcal{C}_x} \mathbb{1}(y \models c)$, where C_x is a list of constraints associated with input x. These tasks, model families, and datasets are standard in recent RL-for-LLM literature (Liu et al., 2025c,a).

RL Training. Across all experiments, we optimize the policy using a token-level GRPO objective used in DAPO (Yu et al., 2025). Formally, the objective is:

$$J_{\text{DAPO}}(\theta) = \mathbb{E}_{\substack{(q,a)\sim\mathcal{D} \\ \{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}}} \left[\frac{1}{\sum_{i=1}^{G}|o_i|}\sum_{i=1}^{G}\sum_{t=1}^{|o_i|} \min\left(r_{i,t}(\theta)\hat{A}_{i,t}, \text{clip}(r_{i,t}(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_{i,t}\right)\right], \quad (2)$$

$$\text{where } r_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t}|q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t}|q, o_{i,<t})},$$

$$\text{and } \hat{A}_{i,t} = \frac{R_i - \text{mean}(\{R_i\}_{i=1}^{G})}{\text{std}(\{R_i\}_{i=1}^{G})}.$$

We also follow the hyper-parameters in previous literature (Bhaskar et al., 2025; Yu et al., 2025) if not explicitly mentioned. Concrete values for each hyper-parameter can be found in Appendix E.

Controlled Compute Budget. To ensure a fair comparison and a controlled setting, we use a compute budget of 2 epochs for each domain across experiments in Sections 3–5, with the exception of the prolonged training experiments.

Evaluation Suites. We construct a comprehensive evaluation suite covering the three training domains, namely ***Math***, ***Chat***, and ***Instruction Following (IF)***. Our training datasets are drawn from and unify those used in prior RL-based post-training work for each domain (Zeng et al., 2025a; Wang et al., 2025a; Bhaskar et al., 2025). In addition, we include a set of *out-of-domain* benchmarks to assess generalization and overall reasoning capabilities. Details of each benchmark are provided in Appendix C.

3 Analyzing Cross-Domain Interference in Multi-Task RL

In this section, we study and analyze cross-domain interference in SEQUENTIAL and MIXED RL.

3.1 Experiment 1: SEQUENTIAL RL

Here, we demonstrate the limitations of SEQUENTIAL RL for learning reasoning skills across multiple domains. Specifically, we show that SEQUENTIAL RL suffers from cross-domain interference, which exhibits two failure modes: (1) Forgetting: performance degradation on previously trained domains, and (2) Rigidity: constrained learning on subsequently trained domains. We collectively refer to this phenomenon as **Mode Interference**, reflecting competition between domain-specific optimization objectives.

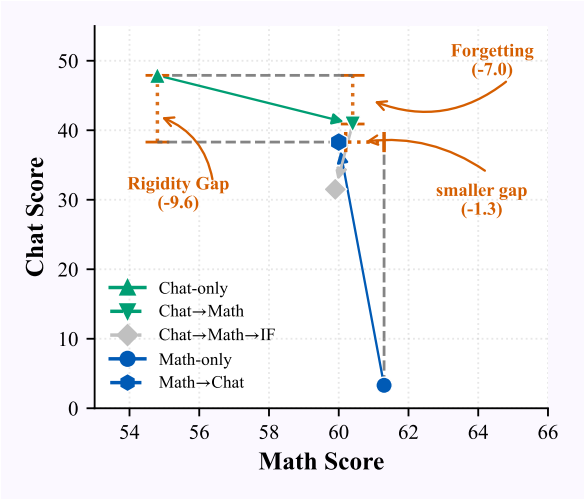

Figure 2: *Chat* vs. *Math* capabilities for SEQUENTIAL RL training, with different training steps in the first stage. We identify Forgetting and Rigidity in this type of approach.

Setting. We study SEQUENTIAL training on two domains: general chat ($\mathcal{D}_{\text{Chat}}$) and mathematical reasoning ($\mathcal{D}_{\text{Math}}$). We train the model on one domain for 2 epochs, then continue RL training on the other domain for another 2 epochs, following the hyperparameters used in prior work (Yu et al., 2025; Bhaskar et al., 2025). We consider both training orders: Chat→Math and Math→Chat. All experiments use the token-level policy gradient loss defined in Eq. (2), with the domain-specific dataset $\mathcal{D} \in \{\mathcal{D}_{\text{Chat}}, \mathcal{D}_{\text{Math}}\}$ applied in each stage. Importantly, the reference policy π_{ref} is reset between stages, following standard practice (Liu et al., 2025a; Wang et al., 2025a). All experiments are conducted on `Qwen-2.5-7B`.

Results. Figure 2 summarizes the performance of models trained under different SEQUENTIAL configurations, compared against single-domain RL baselines and the base model. Full results are provided in Appendix G. Overall, Chat→Math and Math→Chat achieve similar trade-offs, with Chat→Math performing slightly better on average. Below, we highlight two key findings.

RL on one domain degrades performance on previously trained domains. RL training on a second domain **typically** reduces performance on the first domain, a phenomenon resembling **Catastrophic Forgetting** (Luo et al., 2025). While recent work suggests that RL is generally less prone to forgetting than supervised fine-tuning (Chen et al., 2025a; Chu et al., 2025), our results show that it remains a significant limitation in sequential RL. Crucially, however, we observe that this forgetting is **asymmetric** and significantly more severe when *Chat* is the first stage. As shown in Figure 2, continuing RL on *Math* after *Chat* training leads to a sharp drop in *Chat* performance. Training in the reverse order results in less degradation in *Math* capabilities. Additionally, it is also shown in the figure that continual RL training of Chat→Math→IF further degrades the performance on both *Chat* and *Math*.

Furthermore, RL on one domain may rigidize training on subsequent domains. Beyond forgetting, we observe a second and more subtle failure mode: training on one domain can constrain performance gains on subsequent domains, a phenomenon we define as **Reasoning Rigidity**. For example, Chat→Math fails to reach the performance of Math-only on *Math* , and similarly Math→Chat underperforms Chat-only on *Chat*. We further observe that, the gap between ⬢ Math→Chat and ▲ Chat-only on *Chat* (-9.6) is much larger than that between ▼ Chat→Math and ● Math-only on *Math* (-1.3). This indicates an even subtler conclusion – Reasoning Rigidity is more severe when *Math* is trained first. This leads to a critical observation regarding RL training dynamics: the entropy usually decreases in *Math* training and increases in *Chat* training, and the entropy behavior affects subsequent training. As shown in Figure 3 (left), when *Math* is trained in the first stage, the overall entropy on *Chat* data is lower than that of Chat-only training, which explains the Rigidity Gap in Figure 2. Conversely (right), *Chat* training in the first stage brings a high-entropy prior to subsequent *Math* training, which helps better exploration and explains the better *Math* score in Chat→Math.

3.2 Analysis: Mode Interference in SEQUENTIAL RL

Our results in §3.1 reveal the limitation of SEQUENTIAL RL training, referred to as Mode Interference, which emphasizes that different domain objectives compete during RL optimization rather than composing additively.

Mode Interference limits the effectiveness of scaling SEQUENTIAL training. Increasing the amount of training on a single domain may not reliably translate into improved multi-domain performance, as gains in one stage can be offset by interference in previous or subsequent stages.

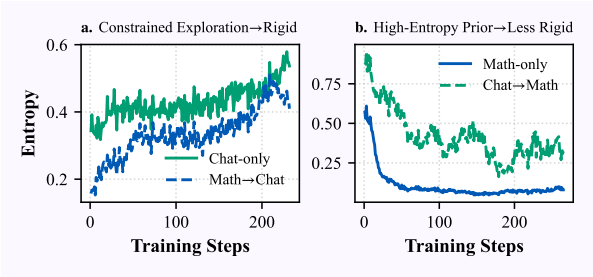

Figure 3: Entropy analysis on Reasoning Rigidity. **Left (Chat training)**: entropy dynamics comparison of Math→Chat and Chat-only. **Right (Math training)**: entropy dynamics comparison of Chat→Math and Math-only.

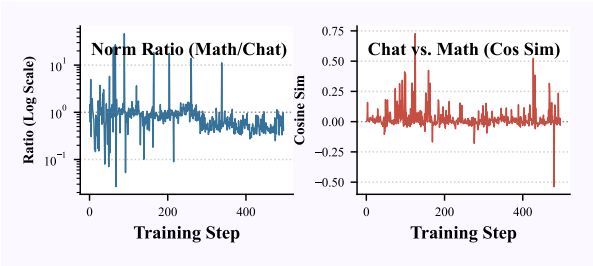

Figure 4: Gradient cosine similarity between *Math* and *Chat* in `Llama-3.1-8B`. Gradient norm ratio is also shown, revealing persistent conflict and imbalance.

Properly ordering training domains mitigates mode interference. As shown in Figure 2 and 3, Chat→Math generally has better performance. These results suggest that, in sequential RL, domains that preserve higher entropy and flexible reasoning behaviors should be trained earlier, as they form a more permissive foundation for later, more structured domains.

3.3 Experiment 2: MIXED RL

We also evaluate MIXED RL training. Unlike SEQUENTIAL RL, which optimizes domains sequentially, MIXED RL mixes data from multiple domains within each batch, allowing gradients from different domains to be applied simultaneously at every optimization step.

Setting. Our primary setting uses a 1:1 mixture of general chat ($\mathcal{D}_{\text{Chat}}$) and mathematical reasoning ($\mathcal{D}_{\text{Math}}$) prompts. We keep the batch size identical to that used in SEQUENTIAL training and train for 2 epochs over the mixed dataset. As a result, the total number of optimization steps matches that of SEQUENTIAL training, ensuring a fair comparison. Unlike SEQUENTIAL training, the reference policy π_{ref} remains fixed as the initial Base model throughout training, following standard practice in multi-task RL (Li et al., 2025a; Zeng et al., 2025b). In addition to the balanced mixture, we evaluate different mixing ratios, including 1:9 and 9:1 for Math versus Chat, to study how different mixing portions affect cross-domain trade-offs.

Results. Figure 5 provides a detailed summary of the performance of MIXED RL under different Math–Chat mixing ratios. Similar to SEQUENTIAL training, we observe persistent negative interplay between domains: models trained with mixed-domain batches consistently underperform the best domain-specific experts. Increasing the proportion

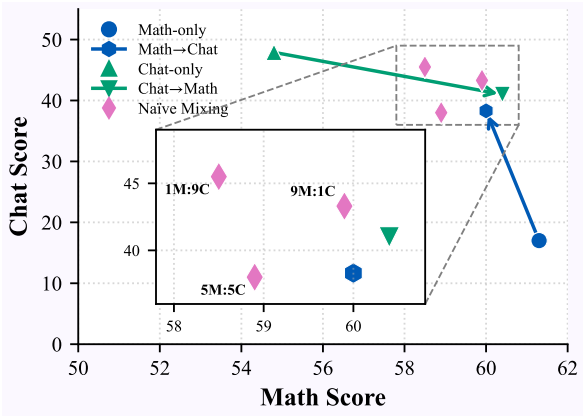

Figure 5: MIXED RL training with different data proportions on `Qwen-2.5-7B`. *Math* performance increases with more *Math* data, while *Chat* shows a more complex trend. Compared to SEQUENTIAL RL, MIXED RL achieves higher *Chat* but lower *Math* performance.

of Math data leads to monotonic improvements in Math performance. However, even with a highly skewed mixture (90% Math, 10% Chat), the resulting model still lags behind a Math-only model trained for 2 epochs. This suggests that simply adjusting data proportions is insufficient to eliminate cross-domain interference in MIXED RL.

3.4 Analysis: Gradient conflicts in MIXED RL

To better understand the source of negative cross-domain interactions in MIXED RL, we analyze the gradients obtained from different domains, which is a commonly used diagnostic for multi-domain training (Yu et al., 2020). Specifically, we examine the alignment between gradients computed from Chat and Math samples across training steps.

As shown in Figure 4, gradient conflicts frequently arise during training, indicating that updates that improve performance on one domain can directly interfere with optimization on the other. These conflicts limit the effectiveness of naive

multi-domain RL training. These observations motivate the need for methods that explicitly detect and resolve cross-domain gradient conflicts. In the next section (§4), we introduce Modular Gradient Surgery, an effective approach for mitigating these conflicts during multi-domain RL.

4 Modular Gradient Surgery

To mitigate gradient conflicts in MIXED training, we introduce a new technique that resolves interference at the module level. Our approach addresses gradient conflicts within individual transformer modules by applying gradient surgery (Yu et al., 2020) locally to each module. We present our method in the following paragraphs.

Background: Gradient Surgery. Negative interference between tasks has been a long-standing challenge in multi-task learning. A representative line of research attributes this to conflicting task gradients, leading to the development of gradient surgery methods. These methods address conflicting objectives in multi-task optimization by modifying gradients to avoid destructive interference. A representative approach is PCGrad (Yu et al., 2020), which detects conflicts between task gradients (e.g., via negative cosine similarity) and projects gradients to remove components that harm other tasks. Suppose there are two tasks, let $\mathbf{g}_i = \nabla_\theta \mathcal{L}_i$ and $\mathbf{g}_j = \nabla_\theta \mathcal{L}_j$ denote the gradients for tasks i and j. PCGrad identifies a conflict when the inner product is negative, i.e., $\mathbf{g}_i \cdot \mathbf{g}_j < 0$. To resolve this, $\mathbf{g}_i$ is replaced by its projection onto the orthogonal complement of $\mathbf{g}_j$, and vice versa:

$$\mathbf{g}_i^{\text{proj}} = \mathbf{g}_i - \frac{\mathbf{g}_i \cdot \mathbf{g}_j}{\|\mathbf{g}_j\|^2}\mathbf{g}_j, \tag{3}$$

$$\mathbf{g}_j^{\text{proj}} = \mathbf{g}_j - \frac{\mathbf{g}_j \cdot \mathbf{g}_i}{\|\mathbf{g}_i\|^2}\mathbf{g}_i, \tag{4}$$

$$\mathbf{g}^{\text{new}} = \mathbf{g}_i^{\text{proj}} + \mathbf{g}_j^{\text{proj}}. \tag{5}$$

Here, $\mathbf{g}^{\text{new}}$ is the final gradient for optimization.

Modular gradient surgery (MGS). We propose Modular Gradient Surgery (MGS), which extends the PCGrad framework by resolving gradient conflicts at the module level. Our approach is motivated by the observation that pretrained transformers are inherently modular, with distinct components specializing in different functional domains. Prior work shows that MLP layers are closely associated with parametric memory and knowledge storage (Meng et al., 2022; Elhage et al., 2021; Du et al., 2025), while attention layers primarily handle information routing and reasoning (Li et al., 2023; Todd et al., 2023; Zhang et al., 2025d; Wu et al., 2024; Olsson et al., 2022), with further specialization across depth (Artzy and Schwartz, 2024). This modularity implies that gradient conflicts are not uniform across the network. In practice, we observe substantial variation in gradient conflict frequency and magnitude across modules during multi-domain RL (Figure 4&10). Some modules may experience severe cross-domain interference, while others exhibit largely compatible gradients. This further motivates module-aware conflict resolution, whereas applying global gradient surgery can therefore be overly conservative. Algorithm 1 in Appendix F details the MGS method. As shown in the algorithm, MGS partitions model parameters into modules (e.g., attention and MLP blocks in `model.named_parameters`) and applies gradient surgery independently for each module.

5 Mitigating Cross-Domain Interference with Modular Gradient Surgery

In this section, we show the effectiveness of MGS in §5.1, and provide extended experiments in §5.2.

5.1 Experiment 3: MIXED Training with MGS

Baselines. We compare our methods with several multi-task baselines: 1) Model Merging (Ablin et al., 2025; Wortsman et al., 2022), we use a simple average method as our baseline. 2) Normalized Advantage (Zhang et al., 2025b), this is a multi-task baseline that normalizes token-level advantages of the same tasks to reduce inter-task variance. In addition, we include the results of Global Gradient Surgery (GGS) for comparison. For the details of other baselines, please refer to Appendix J.

Result Comparisons. As shown in Table 1, MGS achieves the most balanced performance among Naïve Mixing, SEQUENTIAL RL, and other multi-task baselines. Compared with Naïve Mixing, MGS improves the total averaged score by 4.3 and 4.5 points on `Qwen` and `Llama`, respectively. In contrast, SEQUENTIAL RL exhibits substantial trade-offs; for example, Chat→Math on `Llama` yields a *Chat* average of -0.1. MGS does not dominate every individual metric, but it provides a better overall trade-off than Global Surgery. On `Llama`, Global Surgery achieves a higher *Math* average, while MGS improves *Chat* performance

		Math (pass@1, avg@4)				Chat				Misc.		
Backbone	**Method**	**MATH**	**GSM**	**AIME**	**M.A.**	**WB**	**AE2**	**CW3**	**C.A.**	**IF**	**Gen.**	**T.A.**
					Prior Works							
Llama	RLMT	4.9	10.2	0.0	5.0	35.6	15.6	36.3	29.2	20.4	21.5	18.9
Qwen	RLMT	71.8	82.7	10.0	54.8	44.7	57.2	41.7	47.9	25.9	35.4	42.4
	General Reasoner	69.4	81.5	16.7	55.9	13.3	38.8	39.2	30.4	25.1	36.7	38.1
Qwen-Inst	OpenThinker v3	78.8	87.7	29.2	65.2	-50.2	19.3	29.7	-0.4	24.4	39.1	32.8
					Multi-task Baselines							
Qwen	Normalized Adv.	76.1	91.2	9.2	58.8	43.8	44.8	39.0	42.5	27.3	34.2	41.9
	Model Merging	74.8	90.6	11.7	59.6	33.0	12.4	-24.8	6.9	24.8	32.9	31.4
					This Work							
Llama	Math→Chat	24.6	71.6	0.0	32.1	**25.7**	29.0	**38.3**	31.0	21.4	25.0	27.9
	Chat→Math	27.6	78.8	**1.7**	36.0	-35.0	6.5	28.3	-0.1	21.3	28.8	21.5
	Naïve Mixing	25.8	76.6	0.0	34.1	15.0	21.5	33.8	23.4	20.7	23.7	25.9
	Global Surgery	**29.8**	**80.4**	**1.7**	**37.3**	2.3	**48.3**	34.9	28.5	22.9	**29.4**	30.1
	Modular Surgery	26.0	73.6	0.8	33.5	24.7	39.6	34.1	**32.8**	**23.0**	29.3	**30.2**
Qwen	Math→Chat	75.9	90.8	13.3	60.0	35.7	35.4	43.9	38.3	24.8	30.8	39.7
	Chat→Math	**78.2**	**92.2**	10.8	**60.4**	36.0	46.8	40.1	41.0	24.4	36.5	42.0
	Naïve Mixing	76.0	89.8	10.8	58.9	47.1	26.2	40.7	38.0	22.1	36.5	40.4
	Global Surgery	73.0	88.8	**14.2**	58.7	45.4	**62.5**	40.2	49.4	25.6	**36.8**	44.2
	Modular Surgery	75.1	90.6	**14.2**	60.0	**48.3**	59.3	**44.2**	**50.6**	**28.2**	35.3	**44.9**

Table 1: **Main Results on different benchmarks.** Comparison of post-training strategies. We report the average for Math (M.A.) and Chat (C.A.) separately. The highest scores are in bold, and the second best are underlined. Total average score is also shown (T.A.).

and slightly improves the total average. On `Qwen`, MGS achieves a higher total average than Global Surgery while maintaining strong *Math*, *Chat*, and *IF* performance. This supports module-wise conflict resolution as a more balanced alternative to global projection.

MGS outperforms previous general models. As shown in Table 1, MGS achieves better performance compared to general models like RLMT and General Reasoner, in *Math*, *Chat*, and *IF*.

GGS might be overly conservative. GGS is worse than MGS, and even in some cases, Naïve Mixing. This may be due to the overly conservative nature of GGS. For example, a conflict in a single, less-critical module could force GGS to project the entire model's gradient, potentially stalling learning in modules where there was no conflict at all.

5.2 Experiment 4: Scaling Training Steps and Domains

After evaluating MGS on the controlled *Math+Chat* setting, we further test whether its ability to mitigate cross-domain interference persists under more challenging settings: (1) training with three tasks and (2) training for more steps. We also provide a mechanistic analysis of MGS in §5.3, with overhead analysis deferred to Appendix H.

Setting. To demonstrate that resolving gradient conflicts also helps MIXED training with more than two tasks, we blend in the third task, i.e., instruction following (*IF*). We also conduct a prolonged training experiment to demonstrate that MGS scales with more training steps. All settings remain the same as the previous ones.

Additional Baselines. For prolonged training, we compare MGS with SEQUENTIAL RL, and for three-task mixing, we compare MGS with Naïve Mixing and GGS.

MGS consistently surpasses other baselines as the number of tasks increases. We show results of three task setting in Table 5, Appendix H. As shown in the table for results on `Llama-3.1-8B`, MGS (**32.6**) still achieves the best averaged performance over Naïve Mixing (27.3) and GGS (30.3). MGS exhibits a more balanced behavior, with the highest scores on Math, Chat, and General QAs, and also achieves the highest averaged performance on `Qwen-2.5-7B`.

RL with MGS scales with increased training steps. In the normal setting, each model is trained with 2 epochs, which is around 480 steps. As

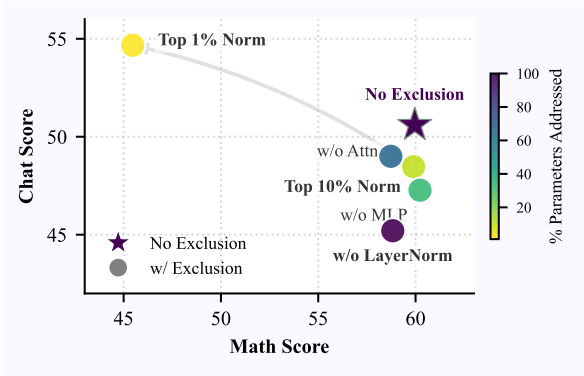

Figure 6: Ablation study of MGS module selection. (1) Ablation by module type (`Attn`, `MLP`, and `LayerNorm`). (2) Ablation by gradient norm ranking (Top-$\mu\%$). %Parameters Addressed refers to the proportion of model parameters to which MGS is applied.

shown in Figure 7 of the Appendix B, we present the performance with a doubled number of training steps. Overall, MGS achieves better performance and scales as the number of training steps increases. Although Math→Chat also gets better scores with more training steps, its performance is far behind MGS. In comparison, Chat→Math degrades in both axes, which reveals the risks of Mode Interference in SEQUENTIAL RL.

5.3 Analysis

In this section, we provide: (1) an evaluation of how gradient norm magnitude correlates with these contributions, and (2) a mechanistic analysis of gradient dynamics to identify which model components contribute most to cross-task interference.

Effect of gradient norm. We further study whether MGS can be restricted to modules with large gradient norms. We rank modules by their candidate gradient norm, $\|\mathbf{g}[\phi]\|_{\text{candidate}} = \max_{k\in\{1,\ldots,K\}} \|\mathbf{g}_k\|_2$, where $\mathbf{g}_k$ is the gradient of task k with respect to module parameters ϕ. We then apply MGS only to the top-$\mu\%$ modules, with $\mu \in \{1, 10\}$. As shown in Fig. 6, applying MGS only to the top-1% norm modules improves *Chat* but substantially hurts *Math* , yielding only marginal gains over the base model. This suggests that high-norm modules are important for *Chat*, whereas *Math* requires coordinated gradient alignment across a broader set of modules. Applying MGS to the top-10% modules gives a more balanced trade-off, indicating that gradient norm is a useful but incomplete heuristic for selecting modules. Overall, these results suggest that effective conflict mitigation cannot be reduced to only the largest-gradient modules.

Effect of different modules. We further examine whether MGS benefits from resolving conflicts at a modular granularity. As shown in Fig. 10, Math–Chat gradient alignment is highly non-uniform across transformer components, suggesting that global surgery may unnecessarily suppress useful updates in non-conflicting modules. We conduct a module-exclusion ablation over `MLP`, `Attention` (`Attn`), and `LayerNorm`, where each variant applies MGS to all modules except one selected family. For example, "w/o MLP" excludes `MLP` blocks from gradient surgery. As shown in Fig. 6, excluding different module families leads to distinct trade-offs: removing MGS from `MLP` mainly degrades *Chat*, while removing it from `Attn` or `LayerNorm` more strongly hurts *Math*. Notably, despite its small parameter count, `LayerNorm` has a large effect, suggesting that conflicts in representation-normalization modules can strongly affect multi-domain behavior.

These results support MGS: localized, module-dependent conflicts favor module-wise surgery over global projection.

6 Related Work

Our work connects LLM post-training and multi-task RL (Ouyang et al., 2022; Shi et al., 2025; Guo et al., 2025a). Despite extensive study of gradient-based balancing in multi-task RL (Yu et al., 2020; Chen et al., 2018), its role in LLM post-training remains underexplored (Wu et al., 2025; Li et al., 2025a; Liu et al., 2025a). We address this with Modular Gradient Surgery, resolving conflicts at the module level for stable, efficient LLM RL. See Appendix M for the complete related work.

7 Conclusion

We study cross-domain interference in multi-domain RL post-training of LLMs. We show that SEQUENTIAL training suffers from mode interference, while MIXED mixed training exhibits cross-task gradient conflicts. To mitigate this, we propose Modular Gradient Surgery (MGS), which resolves conflicts at the module level and preserves compatible updates. Empirically, MGS improves balanced multi-domain performance, and our analyses show that cross-domain conflicts are localized and module-dependent.

Limitations

While Modular Gradient Surgery (MGS) effectively mitigates cross-domain interference, the approach has several limitations. First, the method introduces additional memory overhead for storing task-specific gradients, although this is partially amortized by FSDP parallelism. Second, MGS relies on a predefined module partition (e.g., separating attention, MLP, and LayerNorm), and the optimal granularity of module-level surgery remains unexplored. Third, the current study focuses on three domains (math, chat, and instruction following) with two model families (`Qwen-2.5-7B` and `Llama-3.1-8B`); whether MGS scales to more diverse domains or larger models is not yet verified. Finally, the method requires access to per-task gradient signals, which may not always be available in settings with entangled or implicit task boundaries.

Ethical Considerations

This work focuses on improving multi-domain reasoning in LLMs for math, chat, and instruction following. All experiments use public benchmarks and involve no human or private data. While general risks of LLM misuse exist, we identify no domain-specific ethical concern that warrants special emphasis beyond standard responsible AI practices.

References

Pierre Ablin, Angelos Katharopoulos, Skyler Seto, and David Grangier. 2025. Soup-of-experts: Pretraining specialist models via parameters averaging. In *Forty-second International Conference on Machine Learning*.

AI-MO Team. 2024. AIMO Validation AIME Dataset. https://huggingface.co/datasets/AI-MO/aimo-validation-aime.

Amit Ben Artzy and Roy Schwartz. 2024. Attend first, consolidate later: On the importance of attention in different LLM layers. In *The 7th BlackboxNLP Workshop*.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, and 1 others. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Adithya Bhaskar, Xi Ye, and Danqi Chen. 2025. Language models that think, chat better. *arXiv preprint arXiv:2509.20357*.

Yapei Chang, Yekyung Kim, Michael Krumdick, Amir Zadeh, Chuan Li, Chris Tanner, and Mohit Iyyer. 2025. Bleuberi: Bleu is a surprisingly effective reward for instruction following. *arXiv preprint arXiv:2505.11080*.

Howard Chen, Noam Razin, Karthik Narasimhan, and Danqi Chen. 2025a. Retaining by doing: The role of on-policy data in mitigating forgetting. *arXiv preprint arXiv:2510.18874*.

Weiyu Chen, Baijiong Lin, Xiaoyuan Zhang, Xi Lin, Han Zhao, Qingfu Zhang, and James T Kwok. 2025b. Gradient-based multi-objective deep learning: Algorithms, theories, applications, and beyond. *arXiv preprint arXiv:2501.10945*.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International conference on machine learning*, pages 794–803. PMLR.

Zhao Chen, Jiquan Ngiam, Yanping Huang, Thang Luong, Henrik Kretzschmar, Yuning Chai, and Dragomir Anguelov. 2020. Just pick a sign: Optimizing deep multitask models with gradient sign dropout. *Advances in Neural Information Processing Systems*, 33:2039–2050.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

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

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Jacob Dineen, Aswin Rrv, Qin Liu, Zhikun Xu, Xiao Ye, Ming Shen, Zhaonan Li, Shijie Lu, Chitta Baral, Muhao Chen, and Ben Zhou. 2025. QA-LIGN: Aligning LLMs through constitutionally decomposed QA. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 20619–20642, Suzhou, China. Association for Computational Linguistics.

Hongzhe Du, Weikai Li, Min Cai, Karim Saraipour, Zimin Zhang, Himabindu Lakkaraju, Yizhou Sun, and Shichang Zhang. 2025. How post-training reshapes llms: A mechanistic view on knowledge, truthfulness, refusal, and confidence. *arXiv preprint arXiv:2504.02904*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. 2024. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, and 6 others. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2021/framework/index.html.

Andrew Zhuoer Feng, Cunxiang Wang, Yidong Wang, Yilin Niu, Yu Luo, Hongning Wang, and Minlie Huang. 2026. RLAR: An agentic reward system for multi-task reinforcement learning on large language models.

Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, and 1 others. 2025. Are we done with mmlu? In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5069–5096.

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, and 1 others. 2025. Openthoughts: Data recipes for reasoning models. *arXiv preprint arXiv:2506.04178*.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. 2025. Rubrics as rewards: Reinforcement learning beyond verifiable domains. *arXiv preprint arXiv:2507.17746*.

D. Guo, D. Yang, H. Zhang, Y. Sun, J. Ye, Y. Sun, S. Zhao, Z. Chen, X. Yang, W. Liang, and 1 others. 2025a. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645:633–638.

Jiaxin Guo, Zewen Chi, Li Dong, Qingxiu Dong, Xun Wu, Shaohan Huang, and Furu Wei. 2025b. Reward reasoning model. *arXiv preprint arXiv:2505.14674*.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Xiaomeng Jin, Zhiqi Bu, Bhanukiran Vinzamuri, Anil Ramakrishna, Kai-Wei Chang, Volkan Cevher, and Mingyi Hong. 2025. Unlearning as multi-task optimization: A normalized gradient difference approach with an adaptive learning rate. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 11278–11294.

Dohyeong Kim, Mineui Hong, Jeongho Park, and Songhwai Oh. 2025. Conflict-averse gradient aggregation for constrained multi-objective reinforcement learning. In *The Thirteenth International Conference on Learning Representations*.

Derek Li, Jiaming Zhou, Leo Maxime Brunswic, Abbas Ghaddar, Qianyi Sun, Liheng Ma, Yu Luo, Dong Li, Mark Coates, Jianye Hao, and 1 others. 2025a. Omni-thinker: Scaling multi-task rl in llms with hybrid reward and task scheduling. *arXiv preprint arXiv:2507.14783*.

Jiaze Li, Jingyang Chen, Yuxun Qu, Jianzhong Ju, Zhenbo Luo, Jian Luan, Shijie Xu, Zhenru Lin, Junyou Zhu, Boshen Xu, and 1 others. 2025b. Xiaomi mimo-vl-miloco technical report. *arXiv preprint arXiv:2512.17436*.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36:41451–41530.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Bill Yuchen Lin, Ronan Le Bras, Kyle Richardson, Ashish Sabharwal, Radha Poovendran, Peter Clark, and Yejin Choi. 2025. Zebralogic: On the scaling limits of llms for logical reasoning. *arXiv preprint arXiv:2502.01100*.

Bill Yuchen Lin, Yuntian Deng, Khyathi Chandu, Faeze Brahman, Abhilasha Ravichander, Valentina Pyatkin, Nouha Dziri, Ronan Le Bras, and Yejin Choi. 2024. Wildbench: Benchmarking llms with challenging tasks from real users in the wild. *arXiv preprint arXiv:2406.04770*.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, and 1 others. 2025a. Deepseek-v3. 2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. 2021. Conflict-averse gradient descent for multi-task learning. *Advances in Neural Information Processing Systems*, 34:18878–18890.

Chris Yuhao Liu, Liang Zeng, Yuzhen Xiao, Jujie He, Jiacai Liu, Chaojie Wang, Rui Yan, Wei Shen, Fuxiang Zhang, Jiacheng Xu, and 1 others. 2025b. Skywork-reward-v2: Scaling preference data curation via human-ai synergy. *arXiv preprint arXiv:2507.01352*.

Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. 2025c. Prorl: Prolonged reinforcement learning expands reasoning boundaries in large language models. *arXiv preprint arXiv:2505.24864*.

Tianci Liu, Ran Xu, Tony Yu, Ilgee Hong, Carl Yang, Tuo Zhao, and Haoyu Wang. 2025d. Openrubrics: Towards scalable synthetic rubric generation for reward modeling and llm alignment. *arXiv preprint arXiv:2510.07743*.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025e. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. 2025. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *IEEE Transactions on Audio, Speech and Language Processing*.

Xueguang Ma, Qian Liu, Dongfu Jiang, Ge Zhang, Zejun Ma, and Wenhu Chen. 2025. General-reasoner: Advancing llm reasoning across all domains. *Preprint*, arXiv:2505.14652.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372.

Philipp Mondorf and Barbara Plank. 2024. Beyond accuracy: evaluating the reasoning behavior of large language models–a survey. *arXiv preprint arXiv:2404.01869*.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, and 1 others. 2022. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Samuel J. Paech. 2025. EQ-Bench Creative Writing Benchmark v3. https://github.com/EQ-bench/creative-writing-bench.

Hoang Phan, Xianjun Yang, Kevin Yao, Jingyu Zhang, Shengjie Bi, Xiaocheng Tang, Madian Khabsa, Lijuan Liu, and Deren Lei. 2025. Beyond reasoning gains: Mitigating general capabilities forgetting in large reasoning models. *arXiv preprint arXiv:2510.21978*.

Valentina Pyatkin, Saumya Malik, Victoria Graf, Hamish Ivison, Shengyi Huang, Pradeep Dasigi, Nathan Lambert, and Hannaneh Hajishirzi. 2025. Generalizing verifiable instruction following. *arXiv preprint arXiv:2507.02833*.

ByteDance Seed, Jiaze Chen, Tiantian Fan, Xin Liu, Lingjun Liu, Zhiqi Lin, Mingxuan Wang, Chengyi Wang, Xiangpeng Wei, Wenyuan Xu, and 1 others. 2025. Seed1. 5-thinking: Advancing superb reasoning models with reinforcement learning. *arXiv preprint arXiv:2504.13914*.

Ozan Sener and Vladlen Koltun. 2018. Multi-task learning as multi-objective optimization. *Advances in neural information processing systems*, 31.

Zhengliang Shi, Yiqun Chen, Haitao Li, Weiwei Sun, Shiyu Ni, Yougang Lyu, Run-Ze Fan, Bowen Jin, Yixuan Weng, Minjun Zhu, and 1 others. 2025. Deep research: A systematic survey. *arXiv preprint arXiv:2512.02038*.

Eric Todd, Millicent L Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. 2023. Function vectors in large language models. *arXiv preprint arXiv:2310.15213*.

Vijay Viswanathan, Yanchao Sun, Shuang Ma, Xiang Kong, Meng Cao, Graham Neubig, and Tongshuang Wu. 2025. Checklists are better than reward models for aligning language models. *arXiv preprint arXiv:2507.18624*.

Boxin Wang, Chankyu Lee, Nayeon Lee, Sheng-Chieh Lin, Wenliang Dai, Yang Chen, Yangyi Chen, Zhuolin Yang, Zihan Liu, Mohammad Shoeybi, and 1 others. 2025a. Nemotron-cascade: Scaling cascaded reinforcement learning for general-purpose reasoning models. *arXiv preprint arXiv:2512.13607*.

Junqiao Wang, Zeng Zhang, Yangfan He, Zihao Zhang, Xinyuan Song, Yuyang Song, Tianyu Shi, Yuchen Li, Hengyuan Xu, Kunyu Wu, and 1 others. 2024. Enhancing code llms with reinforcement learning in code generation: A survey. *arXiv preprint arXiv:2412.20367*.

Shudong Wang, Xinfei Wang, Chenhao Zhang, Shanchen Pang, Haiyuan Gui, Wenhao Ji, and Xiaojian Liao. 2025b. Soft conflict-resolution decision transformer for offline multi-task reinforcement learning. *arXiv preprint arXiv:2511.13133*.

Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. 2020. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. *arXiv preprint arXiv:2010.05874*.

Xumeng Wen, Zihan Liu, Shun Zheng, Shengyu Ye, Zhirong Wu, Yang Wang, Zhijian Xu, Xiao Liang, Junjie Li, Ziming Miao, and 1 others. 2025. Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base llms. *arXiv preprint arXiv:2506.14245*.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and 1 others. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pages 23965–23998. PMLR.

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

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. 2024. Retrieval head mechanistically explains long-context factuality. *arXiv preprint arXiv:2404.15574*.

Wei Xiong, Jiarui Yao, Yuhui Xu, Bo Pang, Lei Wang, Doyen Sahoo, Junnan Li, Nan Jiang, Tong Zhang, Caiming Xiong, and 1 others. 2025. A minimalist approach to llm reasoning: from rejection sampling to reinforce. *arXiv preprint arXiv:2504.11343*.

Fengli Xu, Qianyue Hao, Zefang Zong, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, Tianjian Ouyang, Fanjin Meng, and 1 others. 2025. Towards large reasoning models: A survey of reinforced reasoning with large language models. *arXiv preprint arXiv:2501.09686*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Biao Yi, Jiahao Li, Baolei Zhang, Lihai Nie, Tong Li, Tiansheng Huang, and Zheli Liu. 2025. Gradient surgery for safe llm fine-tuning. *arXiv preprint arXiv:2508.07172*.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, and 1 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. *Advances in neural information processing systems*, 33:5824–5836.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. 2025a. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *arXiv preprint arXiv:2503.18892*.

Yuyuan Zeng, Yufei Huang, Can Xu, Qingfeng Sun, Jianfeng Yan, Guanghui Xu, Tao Yang, and Fengzong Lian. 2025b. Zero reinforcement learning towards general domains. *arXiv preprint arXiv:2510.25528*.

Dan Zhang, Min Cai, Jonathan Light, Ziniu Hu, Yisong Yue, and Jie Tang. 2025a. Tdrm: Smooth reward models with temporal difference for llm rl and inference. *arXiv preprint arXiv:2509.15110*.

Hanchen Zhang, Xiao Liu, Bowen Lv, Xueqiao Sun, Bohao Jing, Iat Long Iong, Zhenyu Hou, Zehan Qi, Hanyu Lai, Yifan Xu, and 1 others. 2025b. Agentrl: Scaling agentic reinforcement learning with a multi-turn, multi-task framework. *arXiv preprint arXiv:2510.04206*.

Kaiyan Zhang, Yuxin Zuo, Bingxiang He, Youbang Sun, Runze Liu, Che Jiang, Yuchen Fan, Kai Tian, Guoli Jia, Pengfei Li, and 1 others. 2025c. A survey of reinforcement learning for large reasoning models. *arXiv preprint arXiv:2509.08827*.

Wuwei Zhang, Fangcong Yin, Howard Yen, Danqi Chen, and Xi Ye. 2025d. Query-focused retrieval heads improve long-context reasoning and re-ranking. *arXiv preprint arXiv:2506.09944*.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

Sining Zhoubian, Dan Zhang, and Jie Tang. 2025. Rest-rl: Achieving accurate code reasoning of llms with optimized self-training and decoding. *arXiv preprint arXiv:2508.19576*.

A Datasets

We show the details of datasets used in training in Table 2.

Table 2: Overview of Datasets and Prompt Types

Datasets	Size	Prompt Types
`SimpleRL-Zoo`	8,523	*Math* (GSM8K and MATH)
`Wildchat`	7,544	*Code, Creative writing, etc.*
`Nemotron`	7,930	*Instruction Following*

B Results with Increased Training Steps

The results of the scaling experiment in §5.2 is shown in Figure 7

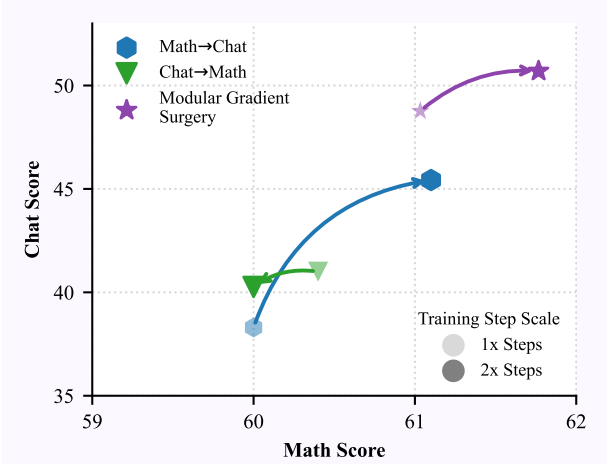

Figure 7: Performance trend comparison as training steps scaled up. Training steps are doubled against the normal setting.

C Details of Evaluation Suites

The datasets we use for each task include:

- **Math**: MATH500 (Lightman et al., 2023), AIME24 (AI-MO Team, 2024), and GSM8K (Cobbe et al., 2021)
- **Chat**: WildBench (Lin et al., 2024), AlpacaEvalv2 (Dubois et al., 2024), and CreativeWriting (Paech, 2025)
- **I**nstruction **F**ollowing: IFBench (Pyatkin et al., 2025) and IFEval (Zhou et al., 2023)
- **General** Capabilities: MMLU redux (Gema et al., 2025), PopQA (Mallen et al., 2023), and Zebra Logic (Lin et al., 2025)

Here, we present the details of the evaluation suites:

- **MATH500** (Lightman et al., 2023): A subset of the MATH benchmark consisting of 500 challenging, competition-level mathematics problems.
- **GSM8K** (Cobbe et al., 2021): A benchmark comprising 8.5k high-quality grade school math word problems designed to test multi-step mathematical reasoning.
- **AIME24** (AI-MO Team, 2024): A dataset containing problems from the 2024 American Invitational Mathematics Examination, evaluating advanced problem-solving capabilities.
- **WildBench** (Lin et al., 2024): This suite evaluates the model's ability to converse with users via 1,024 prompts, including multi-turn interactions. Unlike AlpacaEval 2, WildBench uses an instance-wise manual rubric to mitigate reward hacking. Responses are compared against a GPT-4 reference and

scored as -100 (much worse), -50 (worse), 0 (similar), 50 (better), or 100 (much better). The final metric is the mean of these instance-wise scores.

- **AlpacaEval V2** (Dubois et al., 2024): Contains 805 user prompts paired with reference responses from GPT-4-1106-preview. It outputs a head-to-head win rate (0%–100%) using a generative judge. We use the length-controlled win rate as recommended by Dubois et al., substituting the default judge with GPT-4o.
- **Creative Writing V3** (Paech, 2025): Evaluates the ability to write 96 story chapters under various constraints. We generate an absolute score between 0–100 using Claude-Sonnet-4 as the judge.
- **IFEval** (Zhou et al., 2023): Provides models with 541 simple questions under verifiable constraints (e.g., "do not use commas"). It generates a score between 0–100 signifying how well the model adheres to instructions.
- **IFBench** (Pyatkin et al., 2025): Provides 294 prompts from varied domains, each containing multiple constraints (e.g., "include three numbers in your 22nd sentence"). We average the compliance rate across all examples to produce a score between 0–100.
- **$\text{MMLU}_{\text{redux}}$** (Gema et al., 2025): A manually cleaned version of the MMLU (Hendrycks et al., 2021) benchmark. It consists of 5,700 questions testing general knowledge across 57 subjects.
- **PopQA** (Mallen et al., 2023): Consists of approximately 14k factual questions regarding popular and long-tail entities. The result is reported as a percentage score between 0%–100%.
- **ZebraLogic** (Lin et al., 2025): Tests language models on 1,000 logical grid puzzles. We report the exact match accuracy ranging from 0%–100%.

Hyperparameter	SEQUENTIAL *Math*	SEQUENTIAL *Chat*	SEQUENTIAL *IF*	MIXED
Batch size	64	64	64	64
Samples per prompt	8	8	8	8
Group size	8	8	8	8
Max prompt length	1024	896	896	896
Max response length	4096	4096	4096	4096
Num. epochs	2	2	2	2
Actor learning rate	1e-6	1e-6	1e-6	1e-6
Weight decay	0.01	0.01	0.01	0.01
Scheduler	constant	constant	constant	constant
Warmup ratio	0	0	0	0
Advantage estimator	GRPO	GRPO	GRPO	GRPO
KL coefficient	0.0001/0[d]	0.001	0.001	0.001
Entropy coefficient	0.001/0[d]	0	0	0

[d] We used 0 for both KL and Entropy coefficient at the same time following Yu et al..

Table 3: Hyperparameters for SEQUENTIAL (*Math*, *Chat*, and *IF*), and MIXED training configurations.

D Data Construction for *IF* Data

The original `Nemotron` Instruction Following dataset contains 56,339 data points. To ensure that the number of data is close to that of *Math* and *Chat*, we uniformly sample 7,930 data according to the number of constraints $|C_x|$.

E Hyperparameters and Configurations

In Table 3, we present the hyper-parameters used in SEQUENTIAL and MIXED training. Furthermore,

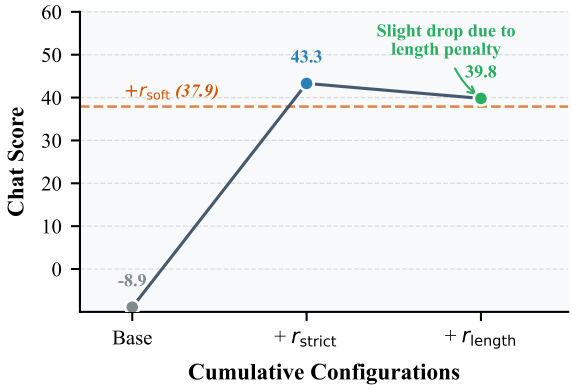

Figure 8: Configuration study of *Math* training in Math→Chat. Scores of the average of AlpacaEvalV2, WildBench, and CreativeWritingV3 when cumulatively adding different training configurations for the first stage *Math* training. All the checkpoints evaluated here are trained with *Math* and *Chat* . The dashed-line shows the score of using soft format reward compared to the strict format reward.

we investigate how using different format reward and adding length penalty affects training. We adopt the

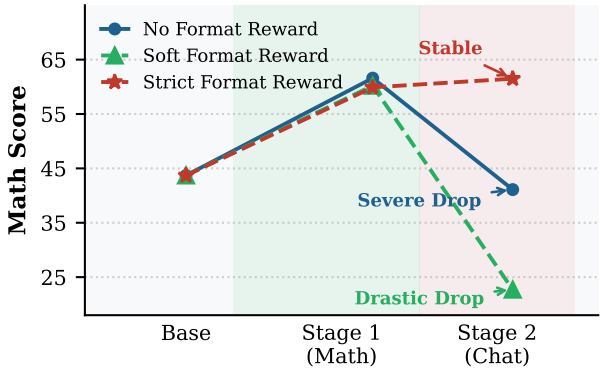

Figure 9: Results on *Math* using different format rewards for *Math* training in SEQUENTIAL RL. No format reward or using soft format reward leads to performance drop, while with strict format reward, SEQUENTIAL shows a stabler behavior.

length penalty in Yu et al. shown below:

$$R_{\text{length}}(y) = \begin{cases} 0, & |y| \leq L_{\text{max}} - L_{\text{cache}} \\ \frac{(L_{\text{max}} - L_{\text{cache}}) - |y|}{L_{\text{cache}}}, & \text{otherwise} \\ -1, & |y| > L_{\text{max}} \end{cases} \tag{6}$$

Figure 8 shows that adding a length penalty slightly downgrades the capability compared to simply using strict format reward. In addition, we also show that using soft format reward will also lead to lower *Chat* score compared to using strict format reward. In our experiments, we by default add length penalty to MIXED to prevent unexpected model behaviors. For format reward, we compare two variants: (1) soft format reward and (2) strict format reward. As shown in Figure 9, training solely with correctness reward, the model achieves high performance on *Math* in the first stage. However, in the second stage of *Chat* training, the performance drops drastically. After adding a strict format reward, although the performance is lower, it continues to grow in the second stage, showing a stabler behavior. In addition, compared to adding soft format reward, we found that only after adding a strict format reward that requires models to generate "`<think></think><response></response>`" pairs in order, can we achieve stable *Math* performance at the second stage. The strict vs. soft format rewards are:

$$S_{\text{tags}} = (\texttt{<think>}, \texttt{</think>}, \texttt{<response>}, \texttt{</response>}) \tag{7}$$

$$R_{\text{soft}}(y) = \mathbb{1}[S_{\text{tags}} \subseteq y] \tag{8}$$

$$R_{\text{strict}}(y) = \mathbb{1}[y \text{ follows order of } S_{\text{tags}}]. \tag{9}$$

The final math reward is multiplied by the format reward. For example, when using R_{strict} as the format reward, the math reward $R_{\text{Math}}(x, y)$ is defined as

$$R_{\text{Math}}^{\text{format}}(x, y) = \mathbb{1}[\text{correct}] * \mathbb{1}[y \text{ follows order of } S_{\text{tags}}] \tag{10}$$

It is worth noting that we only used non-zero entropy coefficient in our preliminary study for "No-Format Math" and "Soft Format Math" as shown in Figure 9. And in our normal setting, we only use length penalty for *Math* training.

Objective with KL divergence and entropy constraints. In our preliminary study, we explored an objective function for *Math* tasks that incorporates both Kullback-Leibler (KL) divergence and entropy

regularization to maintain policy stability and exploration:

$$J_{\text{SEQUENTIAL}}(\theta) = \mathbb{E}_{\substack{(q,a)\sim\mathcal{D} \\ \{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}}} \Bigg[\frac{1}{\sum_{i=1}^{G}|o_i|} \sum_{i=1}^{G}\sum_{t=1}^{|o_i|} \Big(\min\Big(r_{i,t}(\theta)\hat{A}_{i,t}, \text{clip}(r_{i,t}(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_{i,t} \Big) - \beta\, \mathbb{D}_{\text{KL}}\big(\pi_\theta \parallel \pi_{\text{ref}}\big)_{i,t} + \gamma\, \text{H}\big(\pi_\theta\big)_{i,t} \Big) \Bigg], \tag{11}$$

where the token-level KL divergence and entropy are defined as:

$$\mathbb{D}_{\text{KL}}(\cdot)_{i,t} = \sum_{v\in\mathcal{V}} \pi_\theta(v|\cdot) \log \frac{\pi_\theta(v|\cdot)}{\pi_{\text{ref}}(v|\cdot)}, \text{ and } \text{H}(\pi_\theta)_{i,t} = -\sum_{v\in\mathcal{V}} \pi_\theta(v|\cdot) \log \pi_\theta(v|\cdot). \tag{12}$$

Policy entropy analysis. To study the entropy dynamics during training, we measure the average token-level policy entropy across the training dataset $\mathcal{D}$. For a policy π_θ, the mean entropy $\mathcal{H}$ is defined as:

$$\mathcal{H}(\pi_\theta, \mathcal{D}) = \mathbb{E}_{x\sim\mathcal{D}} \left[\frac{1}{|\boldsymbol{y}|} \sum_{t=1}^{|\boldsymbol{y}|} \text{H}(\pi_\theta)_t \right] = -\frac{1}{|\mathcal{D}|} \sum_{x\in\mathcal{D}} \frac{1}{|\boldsymbol{y}|} \sum_{t=1}^{|\boldsymbol{y}|} \sum_{v\in\mathcal{V}} \pi_\theta(v|\cdot) \log \pi_\theta(v|\cdot). \tag{13}$$

In our experiments, we compute this metric over each batch of prompts used for training to monitor the evolution of policy diversity during training.

F Full Algorithm of Modular Gradient Surgery

We provide the MGS algorithm in Alg.1. We also show a comparison between MGS and GGS in Alg.2.

Algorithm 1 Modular Gradient Surgery

Require: Parameters θ, Batch $\mathcal{B}$ for tasks $\{1 \dots K\}$

Definitions:
  Let θ be partitioned into disjoint modules Φ s.t. $\theta = \bigcup_{\phi \in \Phi} \phi$.
  Let Flatten($\mathbf{g}$) map a gradient tensor (or list of tensors) to $\mathbb{R}^D$.
  Let Unflatten($\mathbf{v}$) map a vector $\mathbf{v} \in \mathbb{R}^D$ back to the shape of $\mathbf{g}$.
// 1. Compute initial gradients
$\mathbf{g}_k \leftarrow \sum_{\tau \in \mathcal{B}_k} \nabla_\theta \mathcal{L}(\tau; \theta) \quad \forall k \in \{1 \dots K\}$
$\mathbf{g}_k^{\text{PC}} \leftarrow \mathbf{g}_k \quad \forall k$ // Initialize working gradients
// 2. Apply Gradient Surgery
for task i in $\{1 \dots K\}$ **do**
  for task j in RandomPermutation($\{1 \dots K\} \setminus \{i\}$) **do**
    // Apply PCGrad independently per module
    for each module $\phi \in \Phi$ **do**
      Let $\mathbf{v}_i, \mathbf{v}_j = \text{Flatten}(\mathbf{g}_i^{\text{PC}}[\phi]), \text{Flatten}(\mathbf{g}_j[\phi])$
      if $\mathbf{v}_i \cdot \mathbf{v}_j < 0$ **then**
        $\mathbf{v}_i \leftarrow \mathbf{v}_i - \frac{\mathbf{v}_i \cdot \mathbf{v}_j}{\|\mathbf{v}_j\|^2 + \epsilon} \mathbf{v}_j$
        $\mathbf{g}_i^{\text{PC}}[\phi] \leftarrow \text{Unflatten}(\mathbf{v}_i)$
      end if
    end for
  end for
end for
return $\Delta\theta = \sum_i \mathbf{g}_i^{\text{PC}}$

Algorithm 2 Global vs. Modular Gradient Surgery

Require: Parameters θ, Batch $\mathcal{B}$ for tasks $\{1 \dots K\}$, Mode $\in \{\text{GLOBAL}, \text{MODULE}\}$

Definitions:
  Let θ be partitioned into disjoint modules Φ s.t. $\theta = \bigcup_{\phi \in \Phi} \phi$.
  Let Flatten($\mathbf{g}$) map a gradient tensor (or list of tensors) to $\mathbb{R}^D$.
  Let Unflatten($\mathbf{v}$) map a vector $\mathbf{v} \in \mathbb{R}^D$ back to the shape of $\mathbf{g}$.
// 1. Compute initial gradients
$\mathbf{g}_k \leftarrow \sum_{\tau \in \mathcal{B}_k} \nabla_\theta \mathcal{L}(\tau; \theta) \quad \forall k \in \{1 \dots K\}$
$\mathbf{g}_k^{\text{PC}} \leftarrow \mathbf{g}_k \quad \forall k$ // Initialize working gradients
// 2. Apply Gradient Surgery
for task i in $\{1 \dots K\}$ **do**
  for task j in RandomPermutation($\{1 \dots K\} \setminus \{i\}$) **do**
    if Mode is GLOBAL **then**
      // Flatten to perform projection in vector space
      Let $\mathbf{v}_i, \mathbf{v}_j = \text{Flatten}(\mathbf{g}_i^{\text{PC}}), \text{Flatten}(\mathbf{g}_j)$
      if $\mathbf{v}_i \cdot \mathbf{v}_j < 0$ **then**
        // Project $\mathbf{v}_i$ onto orthogonal complement of $\mathbf{v}_j$
        $\mathbf{v}_i \leftarrow \mathbf{v}_i - \frac{\mathbf{v}_i \cdot \mathbf{v}_j}{\|\mathbf{v}_j\|^2 + \epsilon} \mathbf{v}_j$
        $\mathbf{g}_i^{\text{PC}} \leftarrow \text{Unflatten}(\mathbf{v}_i)$
      end if
    else if Mode is MODULE **then**
      // Apply PCGrad independently per module
      for each module $\phi \in \Phi$ **do**
        Let $\mathbf{v}_i, \mathbf{v}_j = \text{Flatten}(\mathbf{g}_i^{\text{PC}}[\phi]), \text{Flatten}(\mathbf{g}_j[\phi])$
        if $\mathbf{v}_i \cdot \mathbf{v}_j < 0$ **then**
          $\mathbf{v}_i \leftarrow \mathbf{v}_i - \frac{\mathbf{v}_i \cdot \mathbf{v}_j}{\|\mathbf{v}_j\|^2 + \epsilon} \mathbf{v}_j$
          $\mathbf{g}_i^{\text{PC}}[\phi] \leftarrow \text{Unflatten}(\mathbf{v}_i)$
        end if
      end for
    end if
  end for
end for
return $\Delta\theta = \sum_i \mathbf{g}_i^{\text{PC}}$

G Additional Results on SEQUENTIAL Training

We provide the full results on *Math* and *Chat* for SEQUENTIAL training checkpoints in Table 4.

		Math (avg@4)				Chat			
Strategy	**Training Configuration**	**MATH**	**GSM**	**AIME**	**M.A.**	**WB**	**AE2**	**CW3**	**C.A.**
		Single-Domain Experts							
Math-only	2 Epochs	77.2	91.8	15.0	61.3	-27.1	4.4	32.7	3.3
	4 Epochs	77.1	92.1	11.7	60.3	-14.1	7.5	30.9	8.1
Chat-only	2 Epochs	71.8	82.7	10.0	54.8	44.7	57.2	41.7	47.9
	4 Epochs	72.7	86.8	16.7	58.7	46.9	57.5	47.6	50.7
		SEQUENTIAL *RL*							
Math → Chat	$\text{Math}^{2ep} \rightarrow \text{Chat}^{2ep}$	75.9	90.8	13.3	60.0	35.7	35.4	43.9	38.3
	$\text{Math}^{4ep} \rightarrow \text{Chat}^{2ep}$	78.0	92.3	12.5	60.9	33.1	41.3	43.5	39.3
Chat → Math	$\text{Chat}^{2ep} \rightarrow \text{Math}^{2ep}$	78.2	92.2	10.8	60.4	36.0	46.8	40.1	41.0
	$\text{Chat}^{4ep} \rightarrow \text{Math}^{2ep}$	77.6	91.5	15.8	61.6	52.0	49.3	40.4	47.2
	$\text{Chat}^{4ep} \rightarrow \text{Math}^{4ep}$	76.6	91.7	11.7	60.0	37.8	46.6	36.4	40.3

Table 4: **Analysis of Sequential RL Training across Different Epochs and Orders.** We evaluate the trade-offs between Math and Chat capabilities. M.A. and C.A. represent the mean performance across their respective benchmarks. All experiments use the Qwen-2.5-7B backbone.

H Additional Results on MIXED Training

Complementary Analysis: Modular gradient conflicts. Complementary to gradient conflict analysis shown in 4, we also show modular gradient conflicts in Figure 10.

Additional Analysis: Parallel training amortizes extra memory cost. Gradient surgery methods were known for their high memory overhead. However, this overhead can be significantly amortized in parallel training. Here we analyze the extra cost on time and memory, brought by MGS. For memory, the overhead is $\mathcal{O}(T \times N)$, where T is the number of tasks and N is the number of parameters. We need to store all the gradients and apply PCGrad before updating the model parameters. For time, the theoretical overhead is $\mathcal{O}(M \cdot T^2)$, where M is the number of modules when using MODULE PCGrad, other $M = 1$, and T is the number of tasks. In practice, the time cost in applying PCGrad is ~ 163 seconds in a ~ 30 hours training, which is negligible. As for GPU memory overhead, the peak memory cost is $Memory_{\text{peak}} \approx (T_{\text{buffer}} + T_{\text{projected}}) \times N \times Bytes$. However, due to the use of Fully-Sharded Data Parallel training, the actual cost for a single GPU is $\mathcal{O}(T \times \frac{N}{world_size})$. Thereby, the peak memory overhead for a single GPU is $Memory_{\text{peak}} \approx (T_{\text{buffer}} + T_{\text{projected}}) \times \frac{N}{world_size} \times Bytes$, where $Bytes$ is determined by the training precision.

I Additional Results for Extended Experiments

Here we show the concrete results of mixing three tasks and prolonged training. Results of mixing three tasks are in Table 5. Results of prolonged training are in Figure 7.

As shown in 5, for `Qwen-2.5-7B`, SEQUENTIAL training suffers from Mode Interference while MIXED methods maintain high performance compared to their corresponding two-task counterparts.

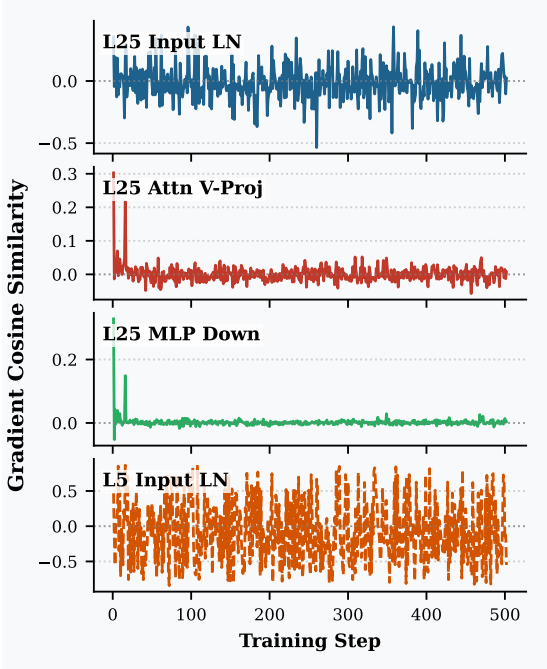

Figure 10: **Modular** gradient cosine similarity between *Math* and *Chat* in `Qwen-2.5-7B` during training. Gradient conflicts exist between different tasks across different training steps and modules.

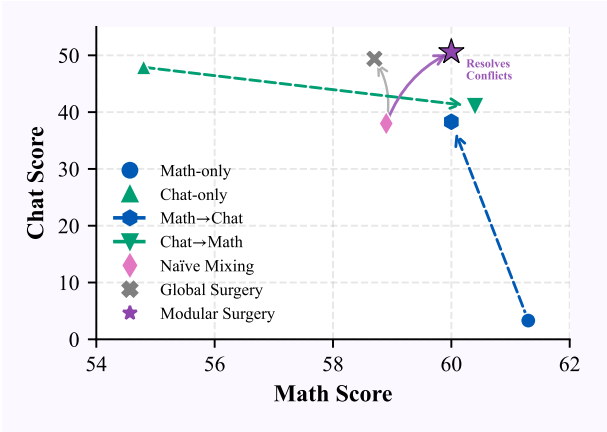

Figure 11: Performance of MGS compared with other methods. While both Modular and Global Gradient Surgery have higher *Chat* score compared to Naïve Mixing, GGS is lower than MGS and has lower *Math* score. MGS also achieves the highest *Chat* score and surpasses Chat-only.

Method	Math	Chat	IF	Gen	Avg
`Qwen-2.5-7B`					
SEQUENTIAL Training					
Chat-Math-IF	59.9	31.5	**39.2**	30.4	40.3
Math-Chat-IF	0.9	20.3	26.2	35.3	20.2
MIXED Training					
Näive Mixing	59.3	**49.3**	33.2	32.8	44.6
Global Surgery	**60.8**	44.2	29.9	34.9	43.0
Mod. Surgery	58.8	48.2	33.7	**35.7**	**45.0**
`Llama-3.1-8B`					
MIXED Training					
Näive Mixing	35.0	22.9	25.4	25.2	27.3
Global Surgery	35.8	24.2	**31.1**	30.3	30.3
Mod. Surgery	**36.0**	**30.5**	30.0	**33.1**	**32.6**

Table 5: Results on mixing three tasks: *Math* , *Chat* , and *IF*.

J Details of Single-task Baselines

Here, we include details of the single-task baselines:

1. RLMT (Bhaskar et al., 2025): this is a single-task baseline that leverages a reward model to train a reasoning LLM with thinking. In our setting, it is equivalent to Chat-only.
2. OpenThinker v3 (Guha et al., 2025): this is an expert model in *Math* with multi-stage training.
3. General Reasoner (Ma et al., 2025): a model that trained on heterogeneous data with a unified reward. Despite the data diversity, we consider it as a single-task baseline because all the data they trained on isn't open-ended and has verifiable answers.

K Case Study

Here we demonstrate a case from the checkpoint of MIXED RL with MGS, trained on *Math* , *Chat* and *IF*. The example is from MATH500. As we can see, the thinking procedures do not contain clear step-by-step structured planning, which is different from the thinking procedures reported in previous single-task checkpoints.

Case Study: MATH500

Question:
Convert the point $(0, 3)$ in rectangular coordinates to polar coordinates. Enter your answer in the form (r, θ), where $r > 0$ and $0 \leq \theta < 2\pi$.

Answer:
<think>
To convert the point (0, 3) from rectangular (Cartesian) coordinates to polar coordinates, we need to determine the values of r and θ. In polar coordinates, a point is represented as (r, θ), where:

- r is the radial distance from the origin (0,0) to the point.
- θ is the angle measured from the positive x-axis to the line connecting the origin and the point.

The formulas to convert from rectangular to polar coordinates are:
$r = \sqrt{x^2 + y^2}$
$\theta = \arctan(y/x)$ if $x \neq 0$, otherwise $\theta = \pi/2$ if $y > 0$, $3\pi/2$ if $y < 0$, and undefined if $y = 0$ and $x = 0$ (which is not the case here).
For the given point (0, 3):
$x = 0$ and $y = 3$.
</think>

<response>
First, calculate r (the radial distance):

$$r = \sqrt{x^2 + y^2} = \sqrt{0^2 + 3^2} = \sqrt{9} = 3$$

Second, calculate θ (the angle):
Since $x = 0$ and $y > 0$, the point (0, 3) lies on the positive y-axis. The angle θ from the positive x-axis to the positive y-axis is $\pi/2$ radians.

$$\theta = \frac{\pi}{2}$$

Therefore, the polar coordinates of the point (0, 3) are (3, $\pi/2$).
The answer within the required format is:

$$\boxed{\left(3, \frac{\pi}{2}\right)}$$

</response>

L Model Template

Because our work focuses on RL training from a base models without a preceding Supervised Fine-tuning (SFT) phase, aka., Zero-RL, we use a customized chat template to incentivize a CoT process. The template is:

Chat Template

A conversation between User and Assistant. Following the User's query, the Assistant first plans a response, and then provides the response. The internal reasoning process is enclosed within `<think> </think>` tags and the response is enclosed within `<response> </response>` tags, i.e., in the format `<think>` reasoning process here `</think> <response>` response here `</response>`.
User: `<query>` ...user text... `</query>` Assistant:

M Complete Related Works

LLM RL Post-training LLM post-training typically refers to the sequence of alignment and specialization stages applied to a base model before deployment. Reinforcement learning has become a central component of this process, most notably through reinforcement learning with human feedback (RLHF) (Christiano et al., 2017; Ouyang et al., 2022; Bai et al., 2022) and more recently reinforcement learning with verifiable rewards (RLVR). RLVR has been shown to significantly improve reasoning performance in domains with objective correctness signals such as mathematics and coding (Wen et al., 2025; Zeng et al., 2025a; Shi et al., 2025). Other works such as rubric-based (Liu et al., 2025d; Gunjal et al., 2025; Dineen et al., 2025) and checklist-based (Viswanathan et al., 2025) rewards show the possibility of LLM RL training with miscellaneous reward functions (Chang et al., 2025; Zhang et al., 2025a; Guo et al., 2025b). Recent work demonstrates that RL can also improve open-ended chat and instruction following. Bhaskar et al. show that applying RL on top of reasoning traces leads to stronger conversational ability, while Zeng et al. demonstrate that zero-RL can generalize beyond strictly verifiable domains. Large-scale systems such as DeepSeek-R1 (Guo et al., 2025a) further validate the effectiveness of RL for reasoning, though they rely on complex multi-stage pipelines involving cold-start SFT (Seed et al., 2025), rejection sampling (Xiong et al., 2025), and multiple rounds of RL. Despite these advances, existing RLHF and RLVR approaches typically focus on a single target domain at a time. How to jointly train a model that is strong at both verifiable reasoning (e.g., Math) and open-ended interaction (e.g., Chat) using a simple RL recipe remains under-explored. Our work addresses this gap by studying RL post-training with multiple heterogeneous rewards simultaneously.

General Multi-task RL Our MIXED training setting is closely related to classical multi-task reinforcement learning, where a single policy is optimized over multiple tasks. A central challenge in this literature is negative interference between tasks, especially when gradients are imbalanced. A line of work addresses this issue through Gradient Manipulation methods (Chen et al., 2025b), including GradVec (Wang et al., 2020), PCGrad (Yu et al., 2020), and GradDrop (Chen et al., 2020), which aim to find Pareto-improving or conflict-averse updates. Another line of work addresses gradient imbalance through Gradient Weighting. This line of work aims at finding a common update direction by adaptively aggregating gradients from different objectives/tasks (Kim et al., 2025; Wang et al., 2025b; Liu et al., 2021; Sener and Koltun, 2018; Chen et al., 2018). PCGrad, in particular, performs gradient surgery by projecting conflicting task gradients to remove destructive components, and has been shown effective in both supervised learning and RL settings. However, these methods treat model parameters as a monolithic vector. In contrast, large language models exhibit strong modular structure across attention, MLP, and normalization components. Our work extends classical gradient conflict resolution to LLM RL by introducing Modular Gradient Surgery, which resolves conflicts at the module level rather than globally. This design better aligns with transformer architectures and leads to improved stability and performance in multi-domain RL post-training.

LLM Multi-task RL Several recent works explore multi-task or multi-objective RL in the context of large language models (Feng et al., 2026; Zhang et al., 2025b; Phan et al., 2025). Guo et al. perform mixed RL training across Math, STEM, and general domains, showing that multi-task zero-RL can be effective with appropriate regularization. OmniThinker (Li et al., 2025a) scales multi-task RL using hybrid rewards and task scheduling strategies. Industrial systems such as DeepSeek-V3.2 (Liu et al., 2025a)

report mixed RL training after expert distillation, though detailed analyses are not publicly available. MiMo-VL-Miloco (Li et al., 2025b) from Xiaomi also conducts mixed training for tasks from different modality. Apart from LLM reasoning, LLM multi-task learning is more often seen in tasks such as LLM unlearning (Jin et al., 2025), whose innate goal is to optimize two conflicted objectives. Other tasks such as LLM safety (Yi et al., 2025) may also aim at optimizing multiple conflicting goals at the same time. However, in LLM RL post-training, few tryouts have been taken to address gradient imbalance. Wu et al. is the first to identify the significance of gradient imbalance in LLM RL post-training, and study through the perspective of Gradient Weighting. They find that the imbalance can hardly be explained by a series of metrics, e.g., learning gains, advantages, etc. Our work serves as the first to address this imbalance through Gradient Manipulation and showcase that resolving the conflicts/imbalances in this way leads to substantial and consistent gains.