

# 000 001 002 003 004 005 SELF-HEALING: RECOVERING PRUNED LARGE REA- 006 SONING MODELS VIA REINFORCEMENT LEARNING 007 008 009

010 **Anonymous authors**  
011 Paper under double-blind review  
012  
013  
014  
015  
016  
017  
018  
019  
020  
021  
022  
023  
024  
025  
026  
027

## ABSTRACT

028 As large reasoning models (LRMs) achieve breakthroughs in reasoning tasks,  
029 building lightweight and efficient LRMAs has become an urgent need for real-  
030 world applications. While structured pruning improves efficiency by reducing  
031 parameters, it often leads to significant performance degradation. To mitigate  
032 this loss, existing methods typically rely on next-token prediction, especially su-  
033 pervised fine-tuning (SFT) for recovery training. However, the effectiveness of  
034 pruning and recovery training in LRMAs remains underexplored. Our empirical  
035 study shows that while structured pruning degrades the mathematical reasoning  
036 ability of LRMAs, it does not completely destroy it, leaving room for compensation  
037 through recovery training. Existing recovery methods merely imitate reasoning  
038 trajectories in the training data, leading to performance bottlenecks and low data  
039 efficiency. To address this, we introduce reinforcement learning with verifiable re-  
040 ward (RLVR) for recovery training, enabling pruned LRMAs to achieve self-healing  
041 performance. Experiments on five representative LRMAs across six mathematical  
042 reasoning benchmarks show that RLVR significantly outperforms SFT-based re-  
043 covery training. At 25% compression, RLVR-based recovery training improves  
044 performance from around 80% (with SFT) to over 95%, approaching or even out-  
045 performing the accuracy of unpruned LRMAs while maintaining efficiency.  
046

## 047 1 INTRODUCTION

048 In recent years, large language models (LLMs) have made significant breakthroughs in traditional  
049 NLP tasks (Bubeck et al., 2023). Research focus gradually shifts towards more challenging rea-  
050 soning tasks. Nowadays, many large reasoning models (LRMs), including OpenAI-01/o3 (OpenAI,  
051 2025), DeepSeek-R1 (Guo et al., 2025), and Gemini-2.5 (DeepMind, 2025) have demonstrated ex-  
052 traordinary reasoning abilities. Reasoning, as a higher cognitive ability, often relies on emergent  
053 abilities enabled by models with tens or even hundreds of billions of parameters (Wei et al., 2022a).  
054 However, the high computational cost of such large-scale models severely limits their practical de-  
055 ployment. To promote the democratization of LRMAs, researchers are exploring efficient reasoning  
056 models (Liu et al., 2025a; Li et al., 2025; Zhang et al., 2025; Wang et al., 2025b;a), to significantly  
057 reduce model size or computational overhead while maintaining reasoning performance.  
058

059 Among various model compression techniques, pruning shows potential in reducing the computa-  
060 tional demands of LLMAs by removing model parameters (Frantar & Alistarh, 2023; Ma et al., 2023).  
061 In particular, structured pruning, which removes channels, blocks, or layers, can produce efficient  
062 models that do not rely on specific hardware. However, even the most advanced structured prun-  
063 ing methods still lead to significant performance degradation (An et al., 2024; Sandri et al., 2025;  
064 Gao et al., 2024), especially on complex tasks involving commonsense understanding and reason-  
065 ing (Yang et al., 2024b). This limitation makes structured pruning still not widely applicable to  
066 lightweight LRMAs. Therefore, post-pruning recovery training is essential to mitigate performance  
067 degradation and recover model capabilities. Designing an effective recovery mechanism is now  
068 crucial for practically deploying pruned LLMAs or LRMAs.  
069

070 The mainstream recovery training relies on next-token prediction (NTP), including continued pre-  
071 training (CPT) and supervised fine-tuning (SFT). For example, Sheared LLaMA (Xia et al., 2024)  
072 uses large-scale corpora for continued pre-training, while LLM-Pruner (Ma et al., 2023) conducts  
073 SFT with instruction data to restore pruned model performance at a low training cost. Although these  
074

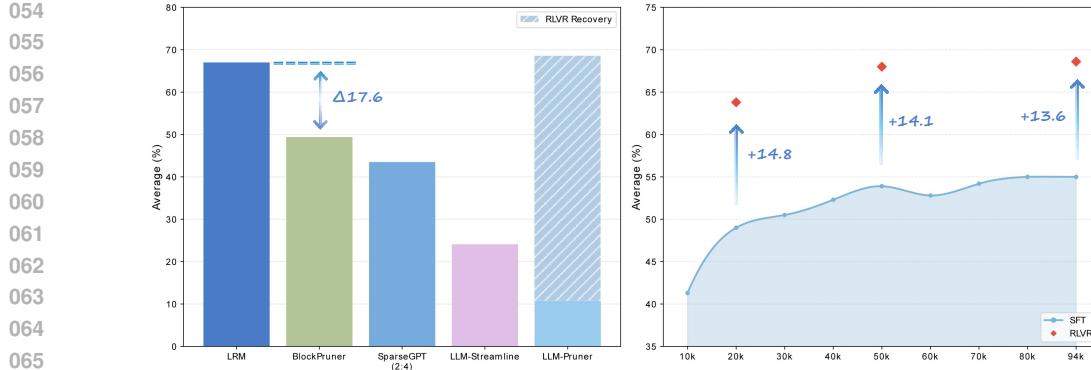


Figure 1: (Left) Structured or semi-structured pruning severely degrades the reasoning ability of DeepSeek-R1-Distill-Llama-8B, but the proposed RLVR recovery training can effectively recover it. (Right) RLVR can break the performance bottleneck in SFT recovery training for pruned LRM.

methods can alleviate the impact of pruning to some extent, they still face critical challenges. First, there is a clear performance bottleneck. When the pruning ratio exceeds 20%, the model typically recovers only about 80% of its original performance, which limits the practicality of highly pruned models. Then, both CPT and SFT demand large-scale data, resulting in low sample efficiency. Moreover, existing pruning and recovery methods have not been thoroughly evaluated on LRM. Therefore, whether they can be used to build effective lightweight LRM remains an open question.

This work systematically investigates the impact of structured pruning on the mathematical reasoning ability of LRM, and explores the potential of recovery training to mitigate performance degradation. Experimental results show that although pruning causes severe degradation in reasoning performance, it does not eliminate the mathematical reasoning ability, leaving room for recovery through appropriate training. However, mainstream SFT-based recovery suffers from both low data efficiency and a performance bottleneck. Moreover, by imitating the reasoning trajectories in training data, it substantially increases reasoning length, which ultimately reduces the efficiency of pruned LRM. These limitations suggest that an effective recovery method should go beyond imitation, enabling the model to actively explore and strengthen its underlying reasoning capability. Therefore, we propose using reinforcement learning with verifiable reward (RLVR) to recover pruned LRM. RLVR enables a self-healing process, allowing the pruned model to refine and explore its reasoning behavior. Considering the characteristics of recovery training, our approach balances exploration and exploitation in the design of rollouts and reference models. At the 25% compression rate, RLVR improves the average performance on six math reasoning benchmarks from 80% (with SFT) to more than 95% of the original performance, even surpassing the original LRM.

In summary, our contributions are as follows:

- We present the first comprehensive study on the impact of structured pruning on the mathematical reasoning ability of LRM and the limitations of existing recovery training.
- This work is the first to introduce RLVR for recovery training, enabling pruned LRM to self-heal through exploration, thereby overcoming the limitations of existing recovery training methods.
- Experiments show that under a 25% pruning ratio, RLVR recovers over 95% of the math reasoning performance of pruned LRM, while achieving better data and inference efficiency than SFT.

## 2 EFFECT OF MODEL PRUNING ON THE MATHEMATICAL REASONING ABILITY OF LRM

### 2.1 DOES PRUNING COMPLETELY ELIMINATE THE REASONING ABILITY OF LRM?

**Experimental settings** We prune 25% of the parameters of DeepSeek-R1-Distill-LLaMA-8B using 3 structured pruning methods, including LLM-Pruner (Ma et al., 2023), BlockPruner (Zhong et al., 2025), and LLM-Streamline (Chen et al., 2025). We mainly focus on the mathematical reasoning ability of LRM, and therefore utilize commonly used mathematical reasoning benchmarks for

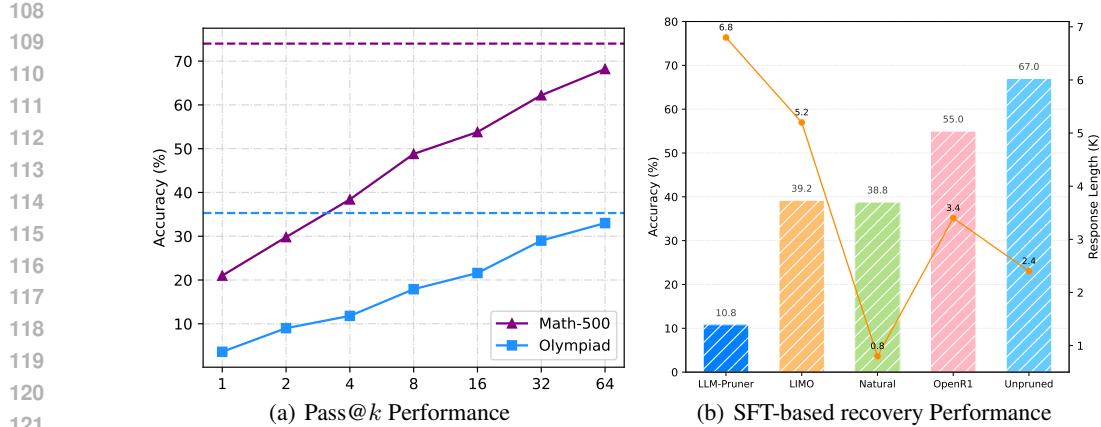


Figure 2: (a) Pass@ $k$  performance of 25% pruned DeepSeek-R1-Distill-Llama-8B on Math-500 and Olympiad Math. The dashed line indicates the performance of the unpruned LRM. (b) Performance and average response length of pruned LRM after SFT recovery with different datasets.

evaluation, including GSM8K (Cobbe et al., 2021), MATH-500 (Hendrycks et al., 2021), Olympiad-Bench (He et al., 2024), College Math (Tang et al., 2024), Gaokao 2023 En (Zhang et al., 2023) and AMC 2023. During evaluation, we set the sampling parameters to the temperature of 0.6, top- $p$  of 0.95, top- $k$  of 20, and the maximum generation length of 8,192 tokens.

**Findings** As Figure 1 (left) shows, pruning severely degrades the mathematical reasoning ability of LRM. The state-of-the-art BlockPruner retains only 73% of the original performance, while LLM-Pruner preserves merely 16%. These results indicate that without recovery training, existing pruning techniques cannot serve as effective compression methods for LRM. Then, we evaluate the reasoning ability retained in the pruned LRM. Specifically, we prompt the pruned LRM to solve each math problem  $k$  times and calculate the proportion of problems for which at least a correct answer (i.e., pass@ $k$ ). To encourage exploration of diverse reasoning paths, we set the sampling temperature to 1. As shown in Figure 2(a), although the pruned model performs poorly on pass@1, its pass@ $k$  improves steadily with larger  $k$ , exhibiting an approximately log-linear growth. When  $k$  is 64, its performance approaches that of the unpruned LRM. This indicates that **although structured pruning degrades the quality of individual outputs, it does not eliminate the LRM’s underlying reasoning capability**. With a sufficiently large exploration space, the LRM remains able to generate correct reasoning results.

## 2.2 CAN SFT NEARLY LOSSLESSLY RECOVER THE REASONING ABILITY OF PRUNED LRM?

**Experimental settings** Following most structured pruning settings, we use SFT to recover the performance degradation caused by LLM-Pruner. We use three datasets from different sources for training: OpenR1<sup>1</sup>, Natural Reasoning<sup>2</sup> (Yuan et al., 2025), and LIMO<sup>3</sup> (Ye et al., 2025). OpenR1 contains 94K high-quality mathematical reasoning problems along with reasoning trajectories generated by DeepSeek-R1. Natural Reasoning provides 1.1M multi-domain reasoning problems mined from pre-training corpora and synthetic trajectories. LIMO consists of 800 carefully curated and annotated mathematical reasoning examples, which have been shown to yield performance on par with reinforcement fine-tuning. LIMO is trained for 15 epochs and the other datasets for 1 epoch, with the learning rate of 5e-5 and the batch size of 32.

**Findings** As shown in Figure 2(b), using OpenR1 as training data achieves the best recovery, achieving about 80% of the original performance. In contrast, the high-quality and small LIMO dataset recovers only about 60%, and the large-scaled Natural Reasoning performs similarly to

<sup>1</sup><https://hf-mirror.com/datasets/open-r1/OpenR1-Math-220k>

<sup>2</sup>[https://huggingface.co/datasets/facebook/natural\\_reasoning](https://huggingface.co/datasets/facebook/natural_reasoning)

<sup>3</sup><https://huggingface.co/datasets/GAIR/LIMO-v2>

LIMO due to the low quality of its synthetic reasoning trajectories. For OpenR1, Figure 1 (right) further shows that as the training data increases, the recovery effect of SFT gradually saturates, leaving about 20% performance gap, which remains unacceptable in practice. Figure 2(b) also reports the average response length. Since SFT tends to imitate the reasoning trajectories of training data, models recovered with OpenR1 and LIMO produce longer responses than the unpruned model. In contrast, Natural Reasoning contains many short responses, and the recovered models fail to adaptively extend reasoning length according to problem difficulty, leading to poor performance. These findings indicates that **current SFT-based recovery methods are constrained by the quantity and quality of training data and suffer from performance bottlenecks, data and inference inefficiency** when recovering the mathematical reasoning ability of LRMs.

### 3 RECOVERY TRAINING VIA RLVR

To restore the mathematical reasoning capabilities of pruned language models, we adopt Group Relative Policy Optimization (GRPO) (Shao et al., 2024), a reinforcement learning method designed to enhance model performance while minimizing computational overhead. Unlike traditional Proximal Policy Optimization (PPO), GRPO eliminates the dependency on the critic model by leveraging group-normalized reward signals to estimate advantages. For a given question, GRPO samples candidate responses  $\{o_1, o_2, \dots, o_G\}$  from the previous policy model  $\pi_{\theta_{old}}$ , then computes token-level advantages using a reward normalization strategy based on group statistics. GRPO optimizes the policy model  $\pi_{\theta}$  by maximization the objective below:

$$\begin{aligned} \mathcal{L}_{GRPO}(\theta) &= \frac{1}{G} \sum_{i=1}^G \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \left( \min \left( r_{i,t} \hat{A}_i, \text{clip} (r_{i,t}; 1 - \varepsilon, 1 + \varepsilon) \hat{A}_i \right) \right) - \beta D_{KL} [\pi_{\theta} || \pi_{ref}], \\ r_{i,t} &= \frac{\pi_{\theta}(o_{i,t} | q, o_{i,<t})}{\pi_{\theta_{old}}(o_{i,t} | q, o_{i,<t})}, \hat{A}_i = \frac{r_i - \text{mean}(r)}{\text{std}(r)}, \end{aligned} \quad (1)$$

where  $\pi_{ref}$  denotes the reference model,  $\text{mean}(r)$  and  $\text{std}(r)$  represents the average and standard deviation over rewards on  $\{o_1, o_2, \dots, o_G\}$ , and  $D_{KL}$  is a KL-divergence term. Most previous work sets the reference model as the initial policy and applies  $D_{KL}$  to prevent the RL optimization from deviating too far. However, we argue that restricting policy updates in the recovery training setting may lead to suboptimal results. Therefore, we do not use a reference model and set  $\beta$  to 0.

**Cold start via SFT** Since pruned LRMs suffer significant losses in instruction following and reasoning abilities, the zero RL training therefore suffers from severe reward sparsity, making the training ineffective. Therefore, we use the model trained by SFT recovery as the initial policy for RLVR recovery training. We choose math problem datasets with CoT as the training data. Given the training data  $\mathcal{D} = \{q_i, r_i\}_{i=1}^n$ , the minimization objective function is:

$$\mathcal{L}_{SFT} = -\frac{1}{\sum_{i=1}^n |r_i|} \sum_{i=1}^n \sum_{t=1}^{|r_i|} \log \pi_{\theta}(r_{i,t} | q_i, r_{i,<t}), \quad (2)$$

where  $q_i$  is a question,  $r_i$  is the response to  $q_i$ , including the CoT and answer. This objective encourages the model to imitate high-quality reasoning trajectories token by token.

**Temperature Warm-up for Exploration** A critical challenge in RLVR recovery is balancing stable exploitation and effective exploration. The training objective of SFT inherently induces a relatively fixed distribution. While a high decoding temperature can enlarge the exploration space, it fails to leverage the model’s existing capabilities and may generate highly random reasoning trajectories, leading to sparser rewards and unstable gradients that hinder training stability. To overcome this trade-off, we adopt a temperature warm-up schedule. We start RLVR training with a low temperature, leveraging the stability and the reasoning behavior inherited from SFT. As training progresses, the temperature is gradually increased using a cosine warm-up function:

$$T(t) = \begin{cases} T_{\min} + A \sin \left( \frac{\pi}{2} \frac{t}{N_{\text{warm}}} \right), & 0 \leq t < N_{\text{warm}}, \\ T_{\max}, & t \geq N_{\text{warm}}, \end{cases} \quad (3)$$

216 where  $T_{\min}$  and  $T_{\max}$  are the minimum and maximum temperatures,  $A = T_{\max} - T_{\min}$ ,  $N_{\text{warm}}$   
 217 controls the warm-up steps. In this way, temperature warm-up allows RLVR recovery training to  
 218 smoothly shift from exploiting the cold start policy to exploring better policy.  
 219

220 **4 EXPERIMENTS**  
 221

222 **4.1 EXPERIMENTAL DETAILS**  
 223

224 **LRMs** To evaluate the effectiveness of our proposed recovery training method, we con-  
 225 duct experiments on five LRM: DeepSeek-R1-Distill-Llama-8B, Llama-3.1-Nemotron-Nano-8B,  
 226 DeepSeek-R1-Distill-Qwen-7B, DeepSeek-R1-Distill-Qwen-14B, and Qwen3-8B. Among them,  
 227 the DeepSeek-R1-Distill series are SFT-based reasoning models, whereas Nemotron-Nano and  
 228 Qwen3 are trained with RLVR.  
 229

230 **Pruning** We use two structured pruning methods, width pruning LLM-Pruner (Ma et al., 2023)  
 231 and depth pruning LayerDrop (Lu et al., 2024b), to perform experiments with 25% pruning rate.  
 232 Among them, LLM-Pruner uses 16 samples from MATH with 2,048 tokens as calibration data and  
 233 applies uniform layer pruning rates to facilitate recovery training. LayerDrop discards the last sev-  
 234 eral layers without requiring calibration data.  
 235

236 **Training and Evaluation** We use the same settings as in Section 2.2 for the SFT cold start. In the  
 237 RLVR stage, we utilize the verl (Sheng et al., 2024) framework to conduct GRPO on pruned LRM.  
 238 We adopt the rule-based reward function: the model receives a score of 1 for a correct answer and  
 239 0 for an incorrect one. We process prompts in batches of 1,024, generating 8 responses per prompt,  
 240 with each response limited to 4K tokens. By default, the sampling temperature is 1.0, the clip ratio  
 241 is set to 0.2, and the learning rate is 1e-6. The sampling temperature increased from 0.2 to 1.0 within  
 242 20 warm-up steps. We conduct all our experiments on 8 NVIDIA A100 GPUs. We use 8K math  
 243 problems from MATH level 3-5 training data for 15 epochs reinforcement training. The evaluation  
 244 settings are the same as described in Section 2.1.  
 245

246 **4.2 MAIN RESULTS**  
 247

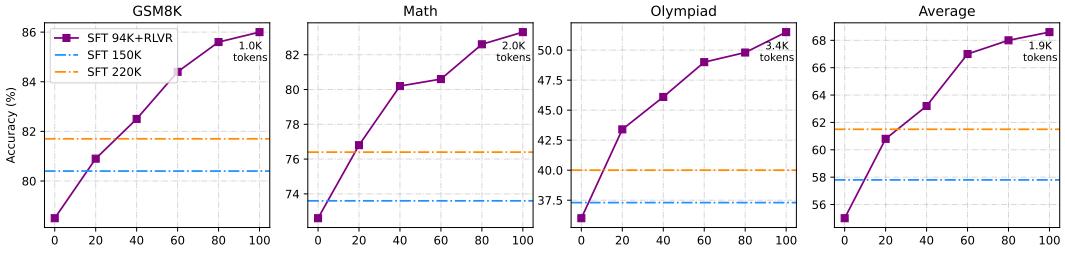
248 **RLVR breaks the recovery performance bottleneck of SFT** We report the main results in 25%  
 249 pruning ratio in Table 1 and Table 4. Overall, RLVR further improves the performance of recovery  
 250 training on top of SFT, even recovering performance comparable to that of the unpruned LRM. For  
 251 all five LRM, width and depth pruning significantly degrade their performance on mathematical  
 252 reasoning benchmarks. In particular, depth pruning reduces performance to less than 5% of the  
 253 original LRM. Recovery training via SFT proves effective, except for Qwen3-8B, all other mod-  
 254 els can be recovered to about 80% of original performance through SFT with 94K long CoT data.  
 255 However, the marginal benefit of collecting more high-quality SFT data for recovery training gradu-  
 256 ally decreases, making it difficult to compensate for the remaining about 20% pruning performance  
 257 degradation by SFT alone. Our experiments show that RLVR may be the final step in narrowing the  
 258 performance gap between pruned LRM and their original models. For DeepSeek-R1-Distill-Llama-  
 259 8B and Qwen3-8B, RLVR improves 15%-26% performance, fully recovering or even surpassing the  
 260 unpruned models. For the other three models, RLVR achieves 94%-96% recovery. Although their  
 261 mathematical reasoning ability is not fully restored, the remaining 4%-6% gap is acceptable given  
 262 the trade-off between performance and efficiency. We believe this gap can be further reduced with  
 263 longer RLVR training steps or sophisticated training tricks. Although RLVR is trained on only 8K  
 264 simple math problems, its performance gains are not limited to basic benchmarks like GSM8K and  
 265 Math. Similar improvements are observed across all evaluation sets, including more challenging  
 266 benchmarks like Olympiad Math, College Math, and AMC 2023.  
 267

268 **RLVR is more data and inference efficient than SFT** It is widely acknowledged that the effec-  
 269 tiveness of SFT depends heavily on the quality and quantity of training data. However, creating high-  
 270 quality long CoT data is costly, relying on expert annotation or generation from advanced LRM.  
 271 Thus, improving either dimension is particularly challenging for long CoT data used in LRM train-  
 272 ing. In contrast, RLVR training only requires questions and ground-truth answers, without the need  
 273 for detailed reasoning traces, making data acquisition significantly more straightforward and more  
 274

270

271 Table 1: Performance of various LRM<sub>s</sub> across multiple math reasoning in 25% pruning ratio. The  
 272 best recovery performance method is indicated in **bold**. The gray percentages indicate the proportion  
 273 of performance retained in the pruned model relative to the original model. Deepseek-LLaMA-8B  
 274 and Nemotron-LLaMA-8B are abbreviations for DeepSeek-R1-Distill-Llama-8B and Llama-3.1-  
 275 Nemotron-Nano-8B, respectively.

Model	GSM8K	MATH-500	Olympiad	College	Gaokao En	AMC23	Avg.
Deepseek-LLaMA-8B	85.6	80.6	46.1	36.2	73.5	80.0	67.0
LLM-Pruner	12.1	15.4	4.1	12.8	15.6	5.0	10.8 (16%)
+ SFT	78.5	72.6	36.0	32.6	65.2	45.0	55.0 (82%)
<b>+ RLVR</b>	<b>86.0</b>	<b>83.3</b>	<b>51.5</b>	<b>41.4</b>	<b>74.2</b>	<b>75.0</b>	<b>68.6 (102%)</b>
LayerDrop	3.5	2.6	1.9	0.9	5.2	2.5	2.8 (4%)
+ SFT	81.5	76.8	38.8	34.7	67.8	50.0	58.7 (87%)
<b>+ RLVR</b>	<b>91.0</b>	<b>83.4</b>	<b>50.8</b>	<b>39.7</b>	<b>76.6</b>	<b>75.0</b>	<b>69.4 (104%)</b>
Nemotron-LLaMA-8B	92.3	90.0	55.1	42.8	78.7	80.0	73.1
LLM-Pruner	7.4	10.6	1.6	9.7	10.6	7.5	7.9 (11%)
+ SFT	79.2	79.2	43.7	36.5	70.6	60.0	61.5 (84%)
<b>+ RLVR</b>	<b>85.3</b>	<b>87.8</b>	<b>51.6</b>	<b>40.3</b>	<b>77.7</b>	<b>77.5</b>	<b>70.0 (96%)</b>
LayerDrop	4.7	5.2	2.6	1.8	3.5	2.5	3.4 (5%)
+ SFT	85.4	80.6	45.5	38.0	75.6	62.5	64.6 (88%)
<b>+ RLVR</b>	<b>85.2</b>	<b>85.4</b>	<b>50.4</b>	<b>40.5</b>	<b>76.4</b>	<b>75.0</b>	<b>68.8 (94%)</b>



299 Figure 3: Recovery performance over the RLVR’s training steps of the pruned DeepSeek-R1-Distill-  
 300 Llama-8B. We show the performance on GSM8K, Math, Olympiad Math, and the average over all  
 301 six benchmarks.

302  
 303 efficient. This gives RLVR strong potential for effective recovery in resource-constrained settings.  
 304 Figure 3 compares RLVR after SFT with 94K data against SFT with extended OpenR1 datasets of  
 305 150K and 220K samples. The results show that while increasing SFT data beyond 94K yields  
 306 further gains, the improvements are marginal. In contrast, RLVR surpasses the benefit of an additional  
 307 100K+ examples with only 8K samples and 20–40 training steps. Figure 3 also shows the average  
 308 response length after RLVR training. Compared with the 3.4K tokens of SFT recovery in Figure  
 309 2(b), RLVR reduces the length by about 1.5K tokens, and by 0.5K tokens relative to the original  
 310 LRM, demonstrating better inference efficiency.

## 312 5 DISCUSSION

### 313 5.1 HOW DOES TEMPERATURE WARM-UP SHAPE THE TRAINING DYNAMICS?

314 To evaluate the effectiveness of temperature warm-up and its underlying causes, we analyze the  
 315 training dynamics of RLVR recovery with temperature warm-up and the fixed high temperature,  
 316 where the latter follows Zeng et al. (2025) with temperature set to 1.0. Figure 4 illustrates that  
 317 temperature warm-up in RLVR training not only reshapes the training dynamics but also achieves  
 318 a better balance between performance and efficiency. In terms of overall accuracy, the warm-up  
 319 strategy consistently outperforms the fixed-temperature baseline, both in the early and late stages  
 320 of training. By the end of training, the fixed-temperature model plateaus at around 67% accuracy,  
 321 whereas the warm-up strategy surpasses 69%. This suggests that dynamic temperature accelerates  
 322 early performance gains and enables pruned models to achieve stronger recovery. For response

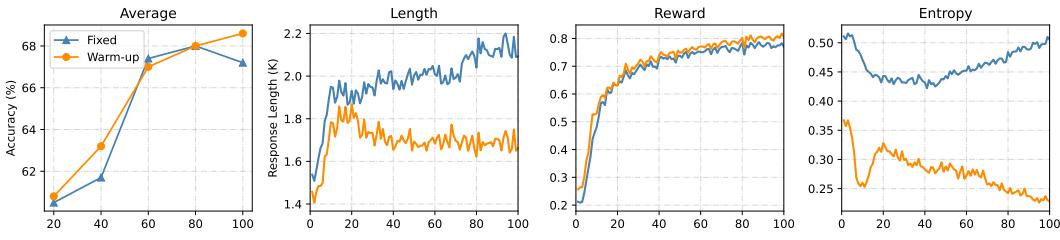


Figure 4: RLVR training dynamics on DeepSeek-R1-Distill-LLaMA-8B. Average denotes the model’s mean performance across six math reasoning datasets, while Length, Reward, and Entropy refer to the policy model’s average response length, reward, and rollout entropy during training.

length, the model trained with the fixed temperature tends to generate longer responses, from 1.4K tokens at the beginning to more than 2.1K tokens, while the warm-up strategy keeps the length stable around 1.7K tokens. This indicates that although fixed temperature encourages exploration, it also induces redundant reasoning trajectories and increases computational cost. In contrast, the warm-up strategy transitions from moderate exploration to concise and efficient reasoning trajectories. Reward and entropy curves further support this analysis. Temperature warm-up leverages reasoning ability inherited from SFT, producing denser reward signals and guiding better policy optimization. As a result, it sustains higher rewards even after both methods converge to the same temperature. Entropy trends reveal the exploration–exploitation balance. The fixed temperature sustains high entropy, reflecting persistent randomness without effective exploitation, while the warm-up strategy transitions from short-term exploration to effective exploitation of high-quality trajectories.

## 5.2 DOES RECOVERY TRAINING RELEARN OR AWAKEN MATH REASONING ABILITY?

As shown in Section 4.2, structured pruning severely degrades LRM reasoning, with some models retaining under 10% of their original performance. Yet, combining SFT and RLVR recovery restores over 95%. This raises an important question: does recovery training relearn or awaken mathematical reasoning ability? we compare the performance of pruned LRMs and LLMs under the same recovery training settings. As shown in Figure 5, after SFT recovery training, the LLaMA-3.1-8B-Instruct model, which lacks prior reasoning ability training, performs about 25% worse than the two LRMs. After RLVR recovery training, all three models show substantial improvement, but the performance gap between the LLM and LRMs does not narrow. This result further suggests that the effectiveness of recovery training depends not only on the training strategy itself but also on the reasoning ability retained in the pruned model. Therefore, studying pruned LRMs is especially important. Compared to training a small model from scratch, an LRM obtained via pruning from a larger LRM is more likely to retain stronger reasoning capabilities.

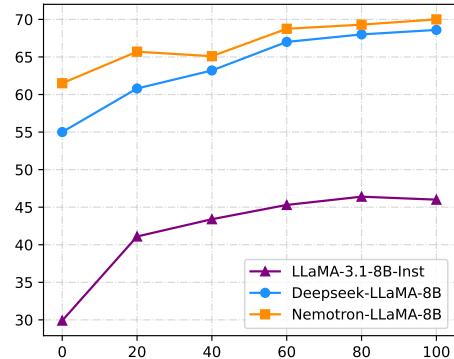


Figure 5: Comparison of the recovery performance of pruned LRM and LLM.

## 5.3 HOW DOES RECOVERY TRAINING AFFECT THE BASIC CAPABILITIES OF PRUNED LRMs?

In LLM pruning research, most work evaluates the language modeling and commonsense knowledge capabilities. For LRMs, generalized abilities still need to be considered in practical applications. Therefore, we further investigate whether recovery training can also positively affect these foundational capabilities. Following standard evaluation settings in prior LLM pruning studies, we report perplexity on WikiText2 to evaluate language modeling ability, and average accuracy across seven commonsense understanding benchmarks (including BoolQ, Winogrande, PIQA, Hellaswag, ARC-e/c, and MMLU) to evaluate commonsense knowledge. As shown in Table 2, pruning significantly degrades both language modeling and commonsense performance.

378

379

Table 3: Performance of pruned DeepSeek-R1-Distill-Llama-8B in 50% pruning ratio.

380

381

Model	GSM8K	MATH	Olympiad	College	Gaokao En	AMC23	Avg.
Dense	85.6	80.6	46.1	36.2	73.5	80.0	67.0
<i>50% LLM-Pruner</i>							
SFT	55.0	54.0	25.6	27.0	47.8	37.5	41.2 (61%)
RLVR	81.1	69.3	36.2	34.7	61.6	52.5	55.9 (83%)
Two-stage	81.0	69.1	35.7	34.8	62.0	50.0	55.4 (83%)
<i>50% LLM-Pruner + LayerDrop</i>							
SFT	69.3	58.4	28.7	29.5	51.4	45.0	47.1 (70%)
RLVR	86.4	72.6	41.1	34.5	66.3	50.0	58.5 (87%)
Two-stage	86.2	72.6	41.5	35.9	65.7	65.0	61.2 (91%)

383

384

385

386

387

388

389

390

391

392

393

Although our recovery training via SFT uses only math-related data, the results show that SFT can still partially recover language modeling and commonsense capabilities, especially in commonsense tasks, where performance improves by nearly 20%. In contrast, RLVR focuses on improving reasoning performance and has a slight impact on these basic abilities. In summary, if the practical application requires language modeling or commonsense understanding, we recommend supplementing recovery training with advanced SFT strategies (e.g., data selection or knowledge distillation). At the same time, RLVR can still be used as a powerful method to improve reasoning performance without harming these foundational capabilities.

407

408

#### 5.4 IS RECOVERY TRAINING EFFECTIVE FOR A HIGHER PRUNING RATIO?

409

410

411

In real-world applications, building lighter and more efficient LRM s requires applying higher pruning ratios. However, the aggressive pruning ratio always leads to more severe performance degradation. Can our proposed recovery training method remain effective under high pruning ratios?

412

413

414

415

416

417

418

419

420

421

422

423

424

We conduct experiments in the 50% pruning ratio with two different pruning configurations: 50% width pruning and pruning 50% across both width and depth dimensions. For the latter configuration, we prune parameters by 25% in width and 33% in depth. Table 3 shows the recovery performance in both settings. First, in terms of overall recovery, models pruned across both width and depth can achieve higher performance. This suggests that a more structurally balanced pruning strategy facilitates better recovery performance. Second, the recovery ability of SFT becomes significantly limited under high pruning ratios, recovering only 60%-70% of the original LRM’s performance. In contrast, RLVR restores about 87%-91% of the original performance. Lastly, we compare one-shot pruning with the two-stage pruning strategy. While the two-stage method achieves slightly better performance, it also requires twice the training cost. Therefore, considering the trade-off between cost and gains, we do not recommend adopting this strategy in resource-constrained real-world scenarios.

425

426

## 6 RELATED WORK

427

428

### 6.1 LARGE REASONING MODELS

429

430

431

Large reasoning models, based on large language models, employ test-time or training-time strategies to enable human-like problem-solving in complex reasoning tasks. Chain-of-thought (CoT; Wei et al., 2022b) is the basic of LRM s, which prompts LLMs to generate intermediate reasoning steps before achieving the result. Empirical and theoretical works show that CoT expands the boundaries

Table 2: Performance of language modeling and commonsense understanding.

Model	Wikitext	Commonsense
Deepseek-LLaMA-8B	13.14	64.84
LLM-Pruner	3753	34.29
+SFT	22.29	47.80
+RLVR	22.27	49.44
Nemotron-LLaMA-8B	626.2	50.38
LLM-Pruner	1042	36.60
+SFT	29.75	45.82
+RLVR	29.54	47.39

432 of LLMs' reasoning capabilities (Feng et al., 2024; Li et al., 2024b). To break the error accumu-  
 433 lation and linear thinking pattern limitation of vanilla CoT, researchers propose test-time compute  
 434 strategies (Snell et al., 2025), such as repeated sampling (Wang et al., 2023; Cobbe et al., 2021; Li  
 435 et al., 2023; Liu et al., 2025b), reflection (Shinn et al., 2023; Gou et al., 2024; Madaan et al., 2023;  
 436 Kim et al., 2023) and tree search (Yao et al., 2023; Besta et al., 2024; Hao et al., 2023). On the  
 437 other hand, training is also an effective way to evolve LLMs into LRM<sub>s</sub>. Supervised fine-tuning  
 438 leverages human-annotated or LRM-generated long CoT data to improve weaker reasoning mod-  
 439 els (Ding et al., 2024; Tang et al., 2024). The strong performance of the Deepseek-R1-Distill series  
 440 models highlights the importance of SFT. LIMO (Ye et al., 2025) and s1 (Muennighoff et al., 2025)  
 441 assume that LLMs inherently possess reasoning abilities that can be activated with only around  
 442 1k training samples. Nevertheless, SFT still suffers from limited generalization and robustness (Li  
 443 et al., 2024a), while reinforcement learning helps alleviate these challenges (Chu et al., 2025). For  
 444 example, Qwen2.5-Math (Yang et al., 2024a) and DeepseekMath (Shao et al., 2024) use PPO and  
 445 GRPO for post-training, respectively. DeepSeek-R1 (Guo et al., 2025) further demonstrates that  
 446 even without SFT as a cold start, rule-based rewards alone can enable RL to unlock strong rea-  
 447 soning capabilities. Recently, many studies (Zeng et al., 2025; Dang & Ngo, 2025) have followed  
 448 DeepSeek-R1's recipe to train LRM<sub>s</sub> on small and mid-sized open-source LLMs.

## 449 6.2 RECOVERY TRAINING FOR PRUNING LARGE LANGUAGE MODELS

450 Pruning is a promising model compression method that reduces model size and accelerates com-  
 451 putation by removing unimportant parameters. Based on the pruning granularity, it can be mainly  
 452 categorized into unstructured pruning (Frantar & Alistarh, 2023; Sun et al., 2024; Yin et al., 2024;  
 453 Zhang et al., 2024b; Dong et al., 2024; Ji et al., 2025), semi-structured pruning (Mishra et al., 2021;  
 454 Zhang et al., 2024a), and structured pruning (Ma et al., 2023; Ashkboos et al., 2024; Ling et al.,  
 455 2024). Unstructured and semi-structured pruning require specific GPU operator support for acceler-  
 456 ation, while structured pruning, which does not depend on specific hardware, has attracted more at-  
 457 tention. However, existing structured pruning methods for LLMs often result in severe performance  
 458 degradation. As a result, retraining has become an essential step in most LLM pruning methods  
 459 to compensate for the performance loss (Muralidharan et al., 2024; Xia et al., 2024). Chen et al.  
 460 (2024) and Sengupta et al. (2025) conclude the scaling law for recovery training, showing that the  
 461 final performance of the pruned LLM follows a power-law relationship with the number of tokens for  
 462 recovery training. To achieve better recovery performance, Sheared LLaMA (Xia et al., 2024) uses  
 463 the dynamic batch loading strategy, sampling recovery data at different proportions according to the  
 464 loss on various domains. PASER (He et al., 2025) clusters data in the semantic space and adaptively  
 465 selects appropriate proportions from each cluster based on the model's performance degradation,  
 466 while filtering out conflicting or irrelevant recovery samples. Additionally, recent studies (Hu et al.,  
 467 2025; Lu et al., 2024a; Munoz et al., 2024) have proposed mask-aware LoRA fine-tuning methods  
 468 tailored to specific sparsity patterns in semi-structured pruning, reducing recovery costs while pre-  
 469 serving the sparse structure. Existing recovery training methods mainly focus on improving LLMs'  
 470 language modeling, commonsense understanding, and basic instruction following abilities, relying  
 471 primarily on continued pretraining or SFT. Unlike previous works, our paper focuses on recover-  
 472 ing more advanced reasoning capabilities in LRM<sub>s</sub> and explores the potential of RLVR in recovery  
 473 training for the first time.

## 474 7 CONCLUSION

475 In this work, we systematically investigate the impact of structured pruning on large reasoning mod-  
 476 els and propose reinforcement learning with verifiable reward as an effective recovery strategy. Our  
 477 study reveals that, although pruning substantially reduces model capacity, the mathematical reason-  
 478 ing ability of LRM<sub>s</sub> is not entirely eliminated, leaving significant potential for recovery. Compared  
 479 with SFT-based recovery training, which primarily imitates training trajectories and suffers from  
 480 performance bottlenecks and data inefficiency, RLVR leverages verifiable reward to guide recovery  
 481 more effectively. Extensive experiments on five LRM<sub>s</sub> across six mathematical reasoning bench-  
 482 marks demonstrate that RLVR consistently surpasses SFT, restoring pruned models to over 95%  
 483 accuracy at the 25% pruning ratio and in some cases exceeding the performance of unpruned LRM<sub>s</sub>.  
 484 These findings highlight the promise of RLVR as a self-healing mechanism for efficient LRM<sub>s</sub>,  
 485 making it a promising direction for future research and optimization.

486 REPRODUCIBILITY STATEMENT  
487488 To ensure reproducibility, we describe the detailed experimental settings in Sections 2 and 4.1.  
489 Furthermore, we provide the source code in the supplementary materials to facilitate independent  
490 verification of our results.  
491492 REFERENCES  
493494 Yongqi An, Xu Zhao, Tao Yu, Ming Tang, and Jinqiao Wang. Fluctuation-based adaptive struc-  
495 tured pruning for large language models. In *Proceedings of the AAAI Conference on Artificial*  
496 *Intelligence*, volume 38, pp. 10865–10873, 2024.497 Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari do Nascimento, Torsten Hoefer, and  
498 James Hensman. SliceGPT: Compress large language models by deleting rows and columns.  
499 In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=vXxardq6db>.  
500501 Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gian-  
502 inazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczek, et al. Graph of  
503 thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI*  
504 *Conference on Artificial Intelligence*, volume 38, pp. 17682–17690, 2024.  
505506 Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece  
507 Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi,  
508 Marco Tulio Ribeiro, and Yi Zhang. Sparks of artificial general intelligence: Early experiments  
509 with gpt-4, 2023.510 Xiaodong Chen, Yuxuan Hu, Xiaokang Zhang, Yanling Wang, Cuiping Li, Hong Chen, and Jing  
511 Zhang. P<sup>2</sup> law: Scaling law for post-training after model pruning, 2024. URL <https://arxiv.org/abs/2411.10272>.  
512513 Xiaodong Chen, Yuxuan Hu, Jing Zhang, Yanling Wang, Cuiping Li, and Hong Chen. Streamlining  
514 redundant layers to compress large language models. In *The Thirteenth International Confer-  
515 ence on Learning Representations*, 2025. URL <https://openreview.net/forum?id=IC5RJvRoMp>.  
516517 Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V.  
518 Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation  
519 model post-training, 2025. URL <https://arxiv.org/abs/2501.17161>.  
520521 Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser,  
522 Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John  
523 Schulman. Training verifiers to solve math word problems, 2021. URL <https://arxiv.org/abs/2110.14168>.  
524525 Quy-Anh Dang and Chris Ngo. Reinforcement learning for reasoning in small llms: What works  
526 and what doesn't, 2025. URL <https://arxiv.org/abs/2503.16219>.  
527528 Google DeepMind. Gemini 2.5: Our most intelligent ai model, 2025.  
529 URL <https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/>.  
530531 Yuyang Ding, Xinyu Shi, Xiaobo Liang, Juntao Li, Qiaoming Zhu, and Min Zhang. Unleashing  
532 reasoning capability of llms via scalable question synthesis from scratch, 2024. URL <https://arxiv.org/abs/2410.18693>.  
533534 Peijie Dong, Lujun Li, Zhenheng Tang, Xiang Liu, Xinglin Pan, Qiang Wang, and Xiaowen Chu.  
535 Pruner-zero: Evolving symbolic pruning metric from scratch for large language models. In Ruslan  
536 Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and  
537 Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*,  
538 volume 235 of *Proceedings of Machine Learning Research*, pp. 11346–11374. PMLR, 21–27 Jul  
539 2024. URL <https://proceedings.mlr.press/v235/dong24b.html>.  
540

540 Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. Towards revealing  
 541 the mystery behind chain of thought: a theoretical perspective. *Advances in Neural Information  
 542 Processing Systems*, 36, 2024.

543

544 Elias Frantar and Dan Alistarh. Sparsegpt: Massive language models can be accurately pruned in  
 545 one-shot. In *International Conference on Machine Learning*, pp. 10323–10337. PMLR, 2023.

546

547 Shangqian Gao, Chi-Heng Lin, Ting Hua, Tang Zheng, Yilin Shen, Hongxia Jin, and  
 548 Yen-Chang Hsu. Disp-llm: Dimension-independent structural pruning for large  
 549 language models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Pa-  
 550 quet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Process-  
 551 ing Systems*, volume 37, pp. 72219–72244. Curran Associates, Inc., 2024. URL  
 552 [https://proceedings.neurips.cc/paper\\_files/paper/2024/file/84a7fc24ed52e8eff514c33e8ac76ea3-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2024/file/84a7fc24ed52e8eff514c33e8ac76ea3-Paper-Conference.pdf).

553

554 Zhibin Gou, Zhihong Shao, Yeyun Gong, yelong shen, Yujiu Yang, Nan Duan, and Weizhu Chen.  
 555 CRITIC: Large language models can self-correct with tool-interactive critiquing. In *The Twelfth  
 556 International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=Sx038qxjek>.

557

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

561

562 Shibo Hao, Yi Gu, Haodi Ma, Joshua Hong, Zhen Wang, Daisy Wang, and Zhiting Hu. Reasoning  
 563 with language model is planning with world model. In Houda Bouamor, Juan Pino, and Ka-  
 564 lika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language  
 565 Processing*, pp. 8154–8173, Singapore, December 2023. Association for Computational Linguis-  
 566 tics. doi: 10.18653/v1/2023.emnlp-main.507. URL <https://aclanthology.org/2023.emnlp-main.507>.

567

568 Bowei He, Lihao Yin, Hui-Ling Zhen, Xiaokun Zhang, Mingxuan Yuan, and Chen Ma. Paser: Post-  
 569 training data selection for efficient pruned large language model recovery, 2025. URL <https://arxiv.org/abs/2502.12594>.

570

571

572 Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han,  
 573 Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench:  
 574 A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal sci-  
 575 entific problems. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the  
 576 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Pa-  
 577 pers)*, pp. 3828–3850, Bangkok, Thailand, August 2024. Association for Computational Linguis-  
 578 tics. doi: 10.18653/v1/2024.acl-long.211. URL <https://aclanthology.org/2024.acl-long.211>.

579

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

584

585 Yuxuan Hu, Jing Zhang, Xiaodong Chen, Zhe Zhao, Cuiping Li, and Hong Chen. Lors: Efficient  
 586 low-rank adaptation for sparse large language model, 2025. URL <https://arxiv.org/abs/2501.08582>.

587

588 Yixin Ji, Yang Xiang, Juntao Li, Qingrong Xia, Ping Li, Xinyu Duan, Zhefeng Wang, and Min  
 589 Zhang. Beware of calibration data for pruning large language models. In *The Thirteenth Interna-  
 590 tional Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=x83w6yGIWb>.

591

592

593 Geunwoo Kim, Pierre Baldi, and Stephen McAleer. Language models can solve computer tasks,  
 2023. URL <https://arxiv.org/abs/2303.17491>.

594 Qintong Li, Leyang Cui, Xueliang Zhao, Lingpeng Kong, and Wei Bi. GSM-plus: A comprehensive  
 595 benchmark for evaluating the robustness of LLMs as mathematical problem solvers. In Lun-Wei  
 596 Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the*  
 597 *Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2961–2984, Bangkok,  
 598 Thailand, August 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.  
 599 acl-long.163. URL <https://aclanthology.org/2024.acl-long.163/>.

600 Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. Mak-  
 601 ing language models better reasoners with step-aware verifier. In Anna Rogers, Jordan Boyd-  
 602 Gruber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association*  
 603 *for Computational Linguistics (Volume 1: Long Papers)*, pp. 5315–5333, Toronto, Canada, July  
 604 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.291. URL  
 605 <https://aclanthology.org/2023.acl-long.291>.

606 Zhen Li, Yupeng Su, Runming Yang, Congkai Xie, Zheng Wang, Zhongwei Xie, Ngai Wong, and  
 607 Hongxia Yang. Quantization meets reasoning: Exploring llm low-bit quantization degradation for  
 608 mathematical reasoning, 2025. URL <https://arxiv.org/abs/2501.03035>.

609 Zhiyuan Li, Hong Liu, Denny Zhou, and Tengyu Ma. Chain of thought empowers transformers to  
 610 solve inherently serial problems. In *The Twelfth International Conference on Learning Represen-*  
 611 *tations*, 2024b. URL <https://openreview.net/forum?id=3EWTEy9MTM>.

612 Gui Ling, Ziyang Wang, Yuliang Yan, and Qingwen Liu. SlimGPT: Layer-wise structured prun-  
 613 ing for large language models. In *The Thirty-eighth Annual Conference on Neural Information*  
 614 *Processing Systems*, 2024. URL <https://openreview.net/forum?id=MxF0IKJtKW>.

615 Ruikang Liu, Yuxuan Sun, Manyi Zhang, Haoli Bai, Xianzhi Yu, Tiezheng Yu, Chun Yuan, and  
 616 Lu Hou. Quantization hurts reasoning? an empirical study on quantized reasoning models, 2025a.  
 617 URL <https://arxiv.org/abs/2504.04823>.

618 Yantao Liu, Zijun Yao, Rui Min, Yixin Cao, Lei Hou, and Juanzi Li. Pairwise rm: Perform best-of-n  
 619 sampling with knockout tournament. *arXiv preprint arXiv:2501.13007*, 2025b.

620 Xudong Lu, Aojun Zhou, Yuhui Xu, Renrui Zhang, Peng Gao, and Hongsheng Li. SPP: Sparsity-  
 621 preserved parameter-efficient fine-tuning for large language models. In Ruslan Salakhutdinov,  
 622 Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix  
 623 Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, vol-  
 624 ume 235 of *Proceedings of Machine Learning Research*, pp. 33254–33269. PMLR, 21–27 Jul  
 625 2024a. URL <https://proceedings.mlr.press/v235/lu24p.html>.

626 Yao Lu, Hao Cheng, Yujie Fang, Zeyu Wang, Jiaheng Wei, Dongwei Xu, Qi Xuan, Xiaoniu Yang,  
 627 and Zhaowei Zhu. Reassessing layer pruning in llms: New insights and methods, 2024b. URL  
 628 <https://arxiv.org/abs/2411.15558>.

629 Xinyin Ma, Gongfan Fang, and Xinchao Wang. LLM-pruner: On the structural pruning of large  
 630 language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.  
 631 URL <https://openreview.net/forum?id=J8Ajf9WfXP>.

632 Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri  
 633 Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad  
 634 Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine:  
 635 Iterative refinement with self-feedback. In *Thirty-seventh Conference on Neural Information Pro-*  
 636 *cessing Systems*, 2023. URL <https://openreview.net/forum?id=S37h0erQLB>.

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

640 Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke  
 641 Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time  
 642 scaling, 2025. URL <https://arxiv.org/abs/2501.19393>.

648 Juan Pablo Munoz, Jinjie Yuan, and Nilesh Jain. SQFT: Low-cost model adaptation in low-  
 649 precision sparse foundation models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen  
 650 (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 12817–  
 651 12832, Miami, Florida, USA, November 2024. Association for Computational Linguistics.  
 652 doi: 10.18653/v1/2024.findings-emnlp.749. URL [https://aclanthology.org/2024.  
 653 findings-emnlp.749/](https://aclanthology.org/2024.findings-emnlp.749/).

654 Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Joshi, Marcin Chochowski, Mostofa  
 655 Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. Compact  
 656 language models via pruning and knowledge distillation, 2024. URL [https://arxiv.org/  
 657 abs/2407.14679](https://arxiv.org/abs/2407.14679).

658 OpenAI. Introducing openai o3 and o4-mini, 2025. URL [https://openai.com/index/  
 659 introducing-o3-and-o4-mini](https://openai.com/index/introducing-o3-and-o4-mini).

660 Fabrizio Sandri, Elia Cunegatti, and Giovanni Iacca. 2ssp: A two-stage framework for structured  
 661 pruning of llms, 2025. URL <https://arxiv.org/abs/2501.17771>.

662 Ayan Sengupta, Siddhant Chaudhary, and Tanmoy Chakraborty. Compression laws for large lan-  
 663 guage models, 2025. URL <https://arxiv.org/abs/2504.04342>.

664 Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang,  
 665 Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathe-  
 666 matical reasoning in open language models, 2024. URL [https://arxiv.org/abs/2402.  
 667 03300](https://arxiv.org/abs/2402.03300).

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

671 Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and  
 672 Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023. URL  
 673 <https://arxiv.org/abs/2303.11366>.

674 Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute  
 675 optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth Interna-  
 676 tional Conference on Learning Representations*, 2025. URL [https://openreview.net/  
 677 forum?id=4FWAwZtd2n](https://openreview.net/forum?id=4FWAwZtd2n).

678 Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. A simple and effective pruning approach  
 679 for large language models. In *The Twelfth International Conference on Learning Representations*,  
 680 2024. URL <https://openreview.net/forum?id=PxoFut3dWW>.

681 Zhengyang Tang, Xingxing Zhang, Benyou Wang, and Furu Wei. MathScale: Scaling instruc-  
 682 tion tuning for mathematical reasoning. In Ruslan Salakhutdinov, Zico Kolter, Katherine  
 683 Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceed-  
 684 ings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings  
 685 of Machine Learning Research*, pp. 47885–47900. PMLR, 21–27 Jul 2024. URL <https://proceedings.mlr.press/v235/tang24k.html>.

686 Jikai Wang, Juntao Li, Lijun Wu, and Min Zhang. Efficient reasoning for llms through speculative  
 687 chain-of-thought, 2025a. URL <https://arxiv.org/abs/2504.19095>.

688 Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha  
 689 Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language  
 690 models. In *The Eleventh International Conference on Learning Representations*, 2023. URL  
 691 <https://openreview.net/forum?id=1PL1NIMMrw>.

692 Zhihai Wang, Jie Wang, Jilai Pan, Xilin Xia, Huiling Zhen, Mingxuan Yuan, Jianye Hao, and Feng  
 693 Wu. Accelerating large language model reasoning via speculative search, 2025b. URL <https://arxiv.org/abs/2505.02865>.

702 Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani  
 703 Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto,  
 704 Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large lan-  
 705 guage models. *Transactions on Machine Learning Research*, 2022a. ISSN 2835-8856. URL  
 706 <https://openreview.net/forum?id=yzkSU5zdwD>. Survey Certification.

707 Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V  
 708 Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models.  
 709 In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in*  
 710 *Neural Information Processing Systems*, volume 35, pp. 24824–24837. Curran Associates, Inc.,  
 711 2022b. URL [https://proceedings.neurips.cc/paper\\_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2022/file/9d5609613524ecf4f15af0f7b31abca4-Paper-Conference.pdf).

712 Mengzhou Xia, Tianyu Gao, Zhiyuan Zeng, and Danqi Chen. Sheared LLaMA: Accelerat-  
 713 ing language model pre-training via structured pruning. In *The Twelfth International Confer-  
 714 ence on Learning Representations*, 2024. URL <https://openreview.net/forum?id=09iOdaeOzp>.

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

719 Ge Yang, Changyi He, Jinyang Guo, Jianyu Wu, Yifu Ding, Aishan Liu, Haotong Qin, Pengliang  
 720 Ji, and Xianglong Liu. LLMCBench: Benchmarking large language model compression for ef-  
 721 ficient deployment. In *The Thirty-eight Conference on Neural Information Processing Systems*  
 722 *Datasets and Benchmarks Track*, 2024b. URL <https://openreview.net/forum?id=wm07z57wNK>.

723 Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik  
 724 Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In  
 725 A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in*  
 726 *Neural Information Processing Systems*, volume 36, pp. 11809–11822. Curran Associates, Inc.,  
 727 2023. URL [https://proceedings.neurips.cc/paper\\_files/paper/2023/file/271db9922b8d1f4dd7aaef84ed5ac703-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2023/file/271db9922b8d1f4dd7aaef84ed5ac703-Paper-Conference.pdf).

728 Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. Limo: Less is more  
 729 for reasoning, 2025. URL <https://arxiv.org/abs/2502.03387>.

730 Lu Yin, You Wu, Zhenyu Zhang, Cheng-Yu Hsieh, Yaqing Wang, Yiling Jia, Gen Li, Ajay Jaiswal,  
 731 Mykola Pechenizkiy, Yi Liang, Michael Bendersky, Zhangyang Wang, and Shiwei Liu. Outlier  
 732 weighed layerwise sparsity (owl): A missing secret sauce for pruning llms to high sparsity, 2024.  
 733 URL <https://arxiv.org/abs/2310.05175>.

734 Weizhe Yuan, Jane Yu, Song Jiang, Karthik Padthe, Yang Li, Dong Wang, Ilia Kulikov, Kyunghyun  
 735 Cho, Yuandong Tian, Jason E Weston, and Xian Li. Naturalreasoning: Reasoning in the wild with  
 736 2.8m challenging questions, 2025. URL <https://arxiv.org/abs/2502.13124>.

737 Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplerl-  
 738 zoo: Investigating and taming zero reinforcement learning for open base models in the wild, 2025.  
 739 URL <https://arxiv.org/abs/2503.18892>.

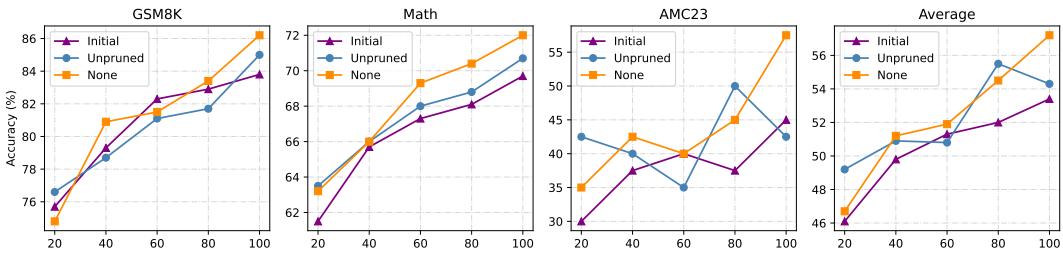
740 Nan Zhang, Yusen Zhang, Prasenjit Mitra, and Rui Zhang. When reasoning meets compression:  
 741 Benchmarking compressed large reasoning models on complex reasoning tasks, 2025. URL  
 742 <https://arxiv.org/abs/2504.02010>.

743 Xiaotian Zhang, Chunyang Li, Yi Zong, Zhengyu Ying, Liang He, and Xipeng Qiu. Evaluating the  
 744 performance of large language models on gaokao benchmark. *arXiv preprint arXiv:2305.12474*,  
 745 2023.

746 Yingtao Zhang, Haoli Bai, Haokun Lin, Jialin Zhao, Lu Hou, and Carlo Vittorio Cannistraci. Plug-  
 747 and-play: An efficient post-training pruning method for large language models. In *The Twelfth*  
 748 *International Conference on Learning Representations*, 2024a. URL <https://openreview.net/forum?id=Tr01Px9woF>.

756  
757 Table 4: Performance of various LRMs across multiple math reasoning in 25% pruning ratio. The  
758 best recovery performance method is indicated in **bold**. The gray percentages indicate the proportion  
759 of performance retained in the pruned model relative to the original model. Deepseek-Qwen-7B/14B  
760 is an abbreviation for DeepSeek-R1-Distill-Qwen-7B/14B.

Model	GSM8K	MATH-500	Olympiad	College	Gaokao En	AMC23	Avg.
Deepseek-Qwen-7B	93.2	88.1	51.1	41.3	78.5	78.3	71.8
LayerDrop	2.6	2.3	0.9	0.9	5.2	0.0	2.0 (3%)
+ SFT	83.4	78.0	38.7	37.7	67.8	50.0	59.3 (83%)
<b>+ RLVR</b>	<b>85.7</b>	<b>85.4</b>	<b>48.3</b>	<b>42.1</b>	<b>76.4</b>	<b>75.0</b>	<b>68.8 (96%)</b>
Deepseek-Qwen-14B	92.7	89.0	52.9	40.3	80.0	87.5	73.7
LayerDrop	0.5	0.3	0.0	0.2	1.0	2.5	0.8 (1%)
+ SFT	85.1	70.6	36.7	35.6	66.5	65.0	59.9 (81%)
<b>+ RLVR</b>	<b>92.0</b>	<b>84.2</b>	<b>47.9</b>	<b>38.6</b>	<b>72.2</b>	<b>80.0</b>	<b>69.3 (94%)</b>
Qwen3-8B	94.8	84.4	45.9	40.1	76.6	75.0	69.5
LayerDrop	5.5	3.0	3.6	1.8	5.2	5.0	4.0 (6%)
+ SFT	76.0	60.4	34.8	26.1	57.1	50.0	50.7 (74%)
<b>+ RLVR</b>	<b>90.7</b>	<b>84.6</b>	<b>48.1</b>	<b>38.6</b>	<b>77.9</b>	<b>75.0</b>	<b>69.2 (100%)</b>



773  
774 Figure 6: Recovery performance over the RLVR’s training steps of the pruned DeepSeek-R1-Distill-  
775 Llama-8B. We show the performance on GSM8K, Math, AMC23, and the average over six bench-  
776 marks. “Initial”, “Unpruned”, and “None” represent using the initial policy, unpruned LRM as a  
777 reference model, and without a reference model, respectively.

778  
779 Yuxin Zhang, Lirui Zhao, Mingbao Lin, Sun Yunyun, Yiwu Yao, Xingjia Han, Jared Tanner, Shiwei  
780 Liu, and Rongrong Ji. Dynamic sparse no training: Training-free fine-tuning for sparse LLMs.  
781 In *The Twelfth International Conference on Learning Representations*, 2024b. URL <https://openreview.net/forum?id=1ndDmZdT4g>.

782  
783 Longguang Zhong, Fanqi Wan, Ruijun Chen, Xiaojun Quan, and Liangzhi Li. BlockPruner:  
784 Fine-grained pruning for large language models. In Wanxiang Che, Joyce Nabende, Ekaterina  
785 Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational  
786 Linguistics: ACL 2025*, pp. 5065–5080, Vienna, Austria, July 2025. Association for Computational  
787 Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.262. URL  
788 <https://aclanthology.org/2025.findings-acl.262/>.

## A THE USE OF LARGE LANGUAGE MODELS

801 In preparing this paper, we used LLMs solely as a writing assistant to improve the fluency and  
802 readability of our manuscript. Specifically, LLMs were employed for grammar correction, phrasing  
803 refinement, and polishing of sentences. No parts of the research ideation, experimental design, or  
804 substantive content generation relied on LLMs. The authors take full responsibility for all scientific  
805 contributions presented in this paper.

## B REFERENCE MODEL

881 The reference model is a key component in GRPO. In previous work, it is typically set as the initial  
882 policy model and used to constrain policy updates via a KL divergence penalty. This helps main-

810 tain training stability by encouraging exploration of high-reward regions while preventing the policy  
 811 from drifting too far from its original distribution. However, as discussed in Section 3.1, this setting  
 812 conflicts with recovery training. Recovery training’s central goal is to compensate for performance  
 813 degradation, which requires the policy model to move beyond the initial state and align more closely  
 814 with the unpruned model. Thus, using the initial policy as the reference potentially hinders perfor-  
 815 mance recovery. To address this contradiction, we explore two alternative reference models. The  
 816 first uses the unpruned LRM as the reference model, serving as knowledge distillation, encouraging  
 817 the current policy to align with a stronger model, and guiding recovery training in the right direc-  
 818 tion. The second approach removes the reference model entirely, eliminating the KL constraint and  
 819 allowing the policy to fully explore and exploit its potential via RLVR.

820 Figure 6 illustrates the evolution of recovery performance under different reference model config-  
 821 urations with a 20k-long CoT cold-start during RLVR training. Overall, the choice of reference  
 822 model significantly impacts recovery effectiveness. Using the initial policy model as the reference  
 823 yields the weakest performance. In contrast, the unconstrained variant, trained without any ref-  
 824 erence model, achieves the best results, with an average performance gain of approximately 4%.  
 825 Notably, employing the unpruned LRM as a reference model accelerates recovery in the early stages  
 826 of training, helping the policy converge more quickly to a better solution. This guidance effect is par-  
 827 ticularly beneficial at the beginning of training, where alignment with the unpruned LRM helps the  
 828 policy avoid poor local optima. However, as training progresses, this guidance gradually becomes a  
 829 constraint. As the policy model continues to align with the unpruned LRM’s distribution, its explo-  
 830 ration space is increasingly restricted. Since the LRM’s reasoning patterns are inherently complex  
 831 and diverse, enforcing distributional similarity may limit the policy’s ability to achieve further per-  
 832 formance gains. In contrast, the unconstrained variant can continue to explore beyond the behavior  
 833 of the reference model, ultimately achieving better performance. These findings suggest that ref-  
 834 erence models in recovery training should be used adaptively. Strong guidance is helpful during  
 835 early recovery, while later stages benefit from relaxed constraints to foster exploration. This offers  
 836 valuable insight for designing more flexible, dynamic reference mechanisms in future research.

## 837 C CASE STUDY

838 We conduct the case study on the pruned DeepSeek-R1-Distill-Llama-8B after SFT and RLVR re-  
 839 covery training. The results show that both methods generate responses exhibiting fundamental  
 840 reasoning patterns such as decomposition, verification, and reflection, with correct answers (marked  
 841 in blue) appearing within the CoT. However, the model recovered with SFT shows a more chaotic  
 842 reasoning process: although it initially generates the correct answer, it later rejects it during veri-  
 843 fication and reflection, ultimately producing an incorrect final answer (marked in red). In contrast,  
 844 the model trained with RLVR demonstrates a clearer reasoning process. While there are still some  
 845 redundancies, it retains the correct answer without overriding it in later steps and produces a final re-  
 846 sponse that follows the verification format, despite no explicit format rewards during training. This  
 847 highlights RLVR’s advantage in improving both reasoning coherence and answer accuracy.

864

**Case 1**

865

866

**Question:**

867

The set of points  $(x, y, z)$  that satisfy  $2x = 3y = -z$  is a line.

868

The set of points  $(x, y, z)$  that satisfy  $6x = -y = -4z$  is another line.

869

Find the angle between these lines, in degrees.

870

**Ground truth:** 90

871

872

**SFT:** Let's think step by step. <think> Okay, so I need to find the angle between two lines in 3D space. The lines are defined by the equations  $2x = 3y = -z$  and  $6x = -y = -4z$ . Hmm, let me recall how to approach this.

873

First, I remember that the angle between two lines is determined by the angle between their direction vectors. So, I need to find the direction vectors of each line and then use the dot product formula to find the angle between them.

874

Let me start with the first line:  $2x = 3y = -z$ . This is a line in 3D space, and the direction vector can be found by looking at the coefficients of x, y, and z in the equations. Since all three expressions are equal, the direction vector should be proportional to the vector  $(2, 3, -1)$  .....Alternatively, .....But let me check .....Wait, .....But let me check if the direction vectors are correct. .....But let me check the calculation again. .....Alternatively, ..... But maybe there's a mistake here. Let me check with another approach. Let me parametrize both lines and find the angle between them.

875

First line:  $2x = 3y = -z$ . Let me set  $2x = 3y = -z = t$ . Then,  $x = t/2$ ,  $y = t/3$ ,  $z = -t$ . So, parametric equations:  $x = t/2$ ,  $y = t/3$ ,  $z = -t$ . So, direction vector is  $(1/2, 1/3, -1)$ .

876

Second line:  $6x = -y = -4z$ . Let me set  $6x = -y = -4z = s$ . Then,  $x = s/6$ ,  $y = -s$ ,  $z = -s/4$ . So, parametric equations:  $x = s/6$ ,  $y = -s$ ,  $z = -s/4$ . Direction vector is  $(1/6, -1, -1/4)$ .

877

Now, to find the angle between the two lines, we can take the direction vectors of each line and compute the angle between them. So, direction vectors are  $(1/2, 1/3, -1)$  and  $(1/6, -1, -1/4)$ . Let's compute the dot product:

878

$(1/2)(1/6) + (1/3)(-1) + (-1)(-1/4) = (1/12) - (1/3) + (1/4) = (1/12 - 4/12 + 3/12) = (0/12) = 0$ . Wait, that can't be. **If the dot product is zero, the angle is 90 degrees. But that contradicts the previous result. What's going on here?**

879

Wait, this is a problem. If the direction vectors are  $(1/2, 1/3, -1)$  and  $(1/6, -1, -1/4)$ , then their dot product is zero. But that would mean the lines are perpendicular. **But according to the previous calculation, the angle was  $\arccos(1/(3 \sqrt{14})) \approx 85$  degrees.**

880

Predict: 85

881

**RLVR:**

882

Let's think step by step. Okay, so I need to find the angle between two lines given by these equations. Let me start by understanding what each line is. The problem says that the first line is defined by  $2x = 3y = -z$ , and the second one is  $6x = -y = -4z$ . Hmm, okay. So both are lines in 3D space, right? Because they have three variables x, y, z. So each equation here is like a parametric equation for the line.

883

First, maybe I should find the direction vectors of each line. Because once I have the direction vectors, I can use the dot product formula to find the angle between them. That makes sense. So direction vectors are key here. Let me recall that for a line defined by equations like  $ax = by = cz$ , the direction vector is  $(a, b, c)$ . Wait, is that right? Let me check. ..... Alternatively .....

884

Alternatively, ..... Let me verify. ..... Wait..... Wait, ..... But this seems complicated, and the problem probably expects the answer to be 90 degrees. But why? Because the dot product is zero. So maybe the answer is 90 degrees. So perhaps the problem is designed so that the direction vectors are perpendicular. Therefore, the angle is 90 degrees. So maybe the answer is 90 degrees. Therefore, boxed answer is

885

boxed90. Wait, but let me check again. Maybe I made a mistake in direction vectors. Wait, if the direction vectors are perpendicular, then the angle is 90 degrees. So if the dot product is zero, then yes. So in that case, the answer is 90 degrees. Therefore, the angle is 90 degrees. So I think that's correct.

886

Therefore, the answer is 90 degrees.

887

\*Final Answer\*\* 90 </think>

888

.....

889

Thus, the angle between the lines is 90 degrees.

890

891

892

893

894

895

896

897

898

899

900

901

902

903

904

905

906

907

908

909

910

911

912

913

914

915

916

917

918

## Case 2

919

920

921

922

923

924

925

926

927

928

929

930

931

932

933

934

935

936

937

938

939

940

941

942

943

944

945

946

947

948

949

950

951

952

953

954

955

956

957

958

959

960

961

962

963

964

965

966

967

968

969

970

971

**Question:**

What is the least positive integer multiple of 30 that can be written with only the digits 0 and 2?

**Ground truth:** 2220

**SFT:** Let's think step by step. <think> Okay, so I need to find the smallest positive multiple of 30 that only uses the digits 0 and 2. Hmm, let's start by recalling that a multiple of 30 must be divisible by both 3 and 10. That means the number has to end with a 0 (since it's divisible by 10) and the sum of its digits must be a multiple of 3 (divisible by 3).

First, since the number can only have 0s and 2s, the last digit has to be 0. So whatever number I come up with, it must end with a 0. Also, the digits can only be 0 or 2, so the number can't have any other digits. That simplifies things a bit because I don't have to worry about other digits interfering with divisibility.

Now, the number has to be the least such multiple. So I need to find the smallest number composed of 0s and 2s that ends with 0 and has a digit sum divisible by 3. Let's think about how to approach this. Since the number must end with 0, the last digit is fixed. The rest of the digits can be 0 or 2. The digit sum (sum of all digits) must be a multiple of 3. So, for example, if the number is 20, the digit sum is 2, which isn't divisible by 3. If it's 200, the digit sum is 2, still not divisible by 3. If it's 220, the digit sum is 4, not divisible by 3. 2020: sum is 4, same problem. 2220: sum is 6, which is divisible by 3. Wait, 2220 is a multiple of 30? Let me check.

**2220 divided by 30 is 74, so yes, 2220 is 30 times 74.** But is there a smaller number? Let's see. Let's check numbers with fewer digits first.....

But wait, maybe there's a smaller number with more digits but smaller in value? For example, 2220 is 2220, but maybe a number like 2020 is smaller? Wait, 2020 is 2020, which is smaller than 2220. But 2020's digit sum is  $2 + 0 + 2 + 0 = 4$ , which isn't divisible by 3. So even though 2020 is smaller, it's not a multiple of 30. Similarly, 220 is 220, which is smaller than 2220, but digit sum 4. So 2220 is the first one that works. But let's check if there's a smaller number with more digits but smaller in value..... So maybe 2220 is the answer. **But wait, let's check if there's a three-digit number. Wait, three digits: 200, 220, 2220?.....**

Predict: 220

**RLVR:**

Let's think step by step Okay, so I need to find the smallest positive multiple of 30 that only uses the digits 0 and 2. Hmm, let's start by recalling that a multiple of 30 must be divisible by both 3 and 10. Right? Because 30 is 3 times 10, so the number has to satisfy both divisibility rules. Let me confirm that. Yes, 30 is  $2 \times 3 \times 5$ , so for a number to be divisible by 30, it must be divisible by 2, 3, and 5. Wait, but actually, since 10 is  $2 \times 5$ , and 3 is separate, so the number must be divisible by 10 (which requires a 0 at the end) and divisible by 3. So, the number must end with 0 (since it's divisible by 10) and the sum of its digits must be a multiple of 3. Got it.

So, the problem reduces to finding the smallest number composed of 0s and 2s that ends with 0 and has a digit sum divisible by 3. Let's think about how to approach this. .....

Wait, ..... Wait, but let's check.

Wait, 2220 divided by 30 is 74, so yes, it's a multiple. But is there a smaller number? Let's see. Let's think of numbers with two digits. ..... So, 2220 is the first one. So, maybe 2220 is the answer. Wait, but let's check if there's a smaller number with more digits. For example, 2220 is four digits, but maybe 2220 is the minimal. Wait, but maybe 2220 is the answer. Wait, but let's check 2220. Wait, 2220 is 2220, which is 2220. So, yes, that's the answer. Wait, but maybe there's a smaller number. Wait, let's think of numbers with digits 0 and 2, ending with 0, and sum divisible by 3. Let's try to find the minimal number. So, starting from the smallest possible numbers. Let's see. The minimal number would have the fewest digits. So, starting with two digits: 20. Sum is 2. Not divisible by 3. Three digits: 200. Sum is 2. Not divisible by 3. Four digits: 2220. Sum is 6. Divisible by 3. So, 2220 is the first one. Therefore, 2220 is the answer. So, the least positive integer multiple of 30 that can be written with only the digits 0 and 2 is **2220**.

Wait, but let me check again. ..... Therefore, 2220 is the answer. So, I think that's the answer.

**\*Final Answer\*\* 2220 </think>**

.....

**Thus, the least positive integer multiple of 30 that can be written with only the digits 0 and 2 is 2220.**

972  
973

## Case 3

974

**Question:**975  
976

In how many ways can 8 people sit around a round table if 3 of the people – Pierre, Rosa, and Thomas – all want to sit together? (Two seatings are considered the same if one is a rotation of the other.)

977

**Ground truth:** 720

978

**SFT:** Let's think step by step. <think> Okay, so I need to figure out how many ways 8 people can sit around a round table, with the condition that Pierre, Rosa, and Thomas all want to sit together. And rotations of the seating are considered the same. Hmm, let's start by recalling some basics about circular permutations.

982

First, for a circular table with  $n$  people, the number of distinct seating arrangements is  $(n-1)!$  because rotations are considered the same. So for 8 people, it would be  $7! = 5040$  ways. But here, we have a restriction: Pierre, Rosa, and Thomas must all sit together. So I need to adjust for that.

983

When there are groups that must sit together, we can treat them as a single unit or block. So instead of 8 individual people, we have 3 people in one block and the remaining 5 people. That makes a total of 6 units to arrange around the table. But wait, since it's a circular table, the number of arrangements for these 6 units would be  $(6-1)! = 5! = 120$ . But hold on, the block of Pierre, Rosa, and Thomas can themselves be arranged in different ways within the block. Since there are 3 people in the block, the number of ways to arrange them is  $3! = 6$ . So we need to multiply the number of arrangements of the blocks by the number of arrangements within the block.

984

So putting it all together, the total number of arrangements would be  $5! * 3! = 120 * 6 = 720$ . But wait, ..... So yes,  $5! * 3! = 720$ . That seems correct.

985

But let me verify with a smaller example. ..... So the answer should be 720.

986

Wait, but let me check another way. ..... So total arrangements would be  $5! * 5 * 3! = 120 * 5 * 6 = 3600$ . Wait, that's different. Hmm, so which one is correct?

987

Wait, this is conflicting. So which approach is right? .....

988

Wait, now I'm confused. .....

989

Wait, perhaps the first approach is correct because when we treat the block as a single entity, .....

990

Wait, but if we fix the 5 people in a certain order, then the gaps are fixed. So inserting the block into a gap would fix the block's position relative to the 5 people.....

991

Predict: 5

992

**RLVR:** Let's think step by step Okay, so I need to figure out how many ways 8 people can sit around a round table, but with the condition that Pierre, Rosa, and Thomas all want to sit together. And rotations are considered the same, so we don't count those as different. Hmm, let's start by recalling some basics about circular permutations.

993

First, normally, if you have  $n$  people sitting around a round table, the number of ways is  $(n-1)!$  because rotations are considered the same. So for 8 people, it would be  $7! = 5040$ . But here, we have a restriction: Pierre, Rosa, and Thomas must all sit together. So how do we handle that?

994

I remember that when certain people must sit together, we can treat them as a single unit or block. So instead of 8 separate people, we have this block plus the other 5 people. So that's 1 block + 5 people = 6 units total. Then, arranging these 6 units around a circular table. But wait, since it's a circular table, the number of arrangements would be  $(6-1)! = 5! = 120$ . But wait, but the block itself can be arranged in different ways, right? Because Pierre, Rosa, and Thomas can be arranged among themselves in  $3!$  ways. So then, total arrangements would be  $5!$  multiplied by  $3!$ .

995

So that would be  $120 * 6 = 720$ . But wait, hold on. Let me check again. .....

996

But let me think again. .....

997

But wait, another way to think about it: .....

998

Wait, but let me check again. .....

999

Alternatively, ..... So total is  $120 * 6 = 720$ . So yes, that's correct.

1000

Therefore, the answer is 720. So I think that's right.

1001

\*Final Answer\*\* [720] </think>

1002

.....

1003

Therefore, the final answer is [720].

1004

1005

1006

1007

1008

1009

1010

1011

1012

1013

1014

1015

1016

1017

1018

1019

1020

1021

1022

1023

1024

1025