

# 000 001 002 003 004 005 006 007 008 009 010 011 012 013 014 015 016 017 018 019 020 021 022 023 024 025 026 027 028 029 030 031 032 033 034 035 036 037 038 039 040 041 042 043 044 045 046 047 048 049 050 051 052 053 GENERATIVE ADVERSARIAL REASONER: ENHANCING LLM REASONING WITH ADVERSARIAL REINFORCE- MENT LEARNING

Anonymous authors

Paper under double-blind review

## ABSTRACT

Large language models (LLMs) with explicit reasoning capabilities excel at mathematical reasoning yet still commit process errors, such as incorrect calculations, brittle logic, and superficially plausible but invalid steps. In this paper, we introduce Generative Adversarial Reasoner, an on-policy joint training framework designed to enhance reasoning by co-evolving an LLM reasoner and an LLM-based discriminator through adversarial reinforcement learning. A compute-efficient review schedule partitions each reasoning chain into logically complete slices of comparable length, and the discriminator evaluates each slice’s soundness with concise, structured justifications. Learning couples complementary signals: the LLM reasoner is rewarded for logically consistent steps that yield correct answers, while the discriminator earns rewards for correctly detecting errors or distinguishing traces in the reasoning process. This produces dense, well-calibrated, on-policy step-level rewards that supplement sparse exact-match signals, improving credit assignment, increasing sample efficiency, and enhancing overall reasoning quality of LLMs. Across various mathematical benchmarks, the method delivers consistent gains over strong baselines with standard RL post-training. Specifically, on AIME24, we improve DeepSeek-R1-Distill-Qwen-7B from 54.0 to 61.3 (+7.3) and DeepSeek-R1-Distill-Llama-8B from 43.7 to 53.7 (+10.0). The modular discriminator also enables flexible reward shaping for objectives such as teacher distillation, preference alignment, and mathematical proof-based reasoning.

## 1 INTRODUCTION

Large language models (LLMs) have demonstrated remarkable mathematical reasoning abilities, often achieving expert-level performance across diverse benchmarks (Achiam et al., 2023; Dubey et al., 2024; Shao et al., 2024; DeepSeek-AI, 2025). However, despite extensive training on large-scale datasets with sophisticated paradigms, these models still suffer from errors in reasoning, such as incorrect calculations, flawed logic, superficially plausible but invalid arguments, and repetitive or incoherent reasoning steps. To tackle these challenges, researchers have explored approaches such as model debate collaboration, in which models debate against each other (Du et al., 2023; Liang et al., 2023) or with themselves (Kuba et al., 2025a; Liu et al., 2025a), and Process Reward Models (Lightman et al., 2023; Wang et al., 2023), which aim to identify and mitigate process errors throughout the reasoning process. These methods provide finer-grained supervision and contribute to more robust and reliable LLM performance.

Among existing approaches, Process Reward Models (PRMs) have shown strong results on complex reasoning tasks, largely because they leverage detailed step-level annotations. However, PRMs face challenges related to annotation costs and data quality (Lightman et al., 2023), as fine-grained labels are expensive and prone to subjective error, and are sometimes susceptible to over- or under-reward issues (Wen et al., 2024; Lv et al., 2025). Alternatively, prompt-based methods employ LLMs as critics for stepwise judgments at a lower cost (Zhang et al., 2024; Gao et al., 2024; Xia et al., 2025). However, their judgments can sometimes be noisy, inconsistent, and less discriminative.

To bridge this gap, we retain a stepwise critic (referred to as the discriminator) but enable it to co-evolve with the LLM reasoner through joint training, generating effective step-level signals with

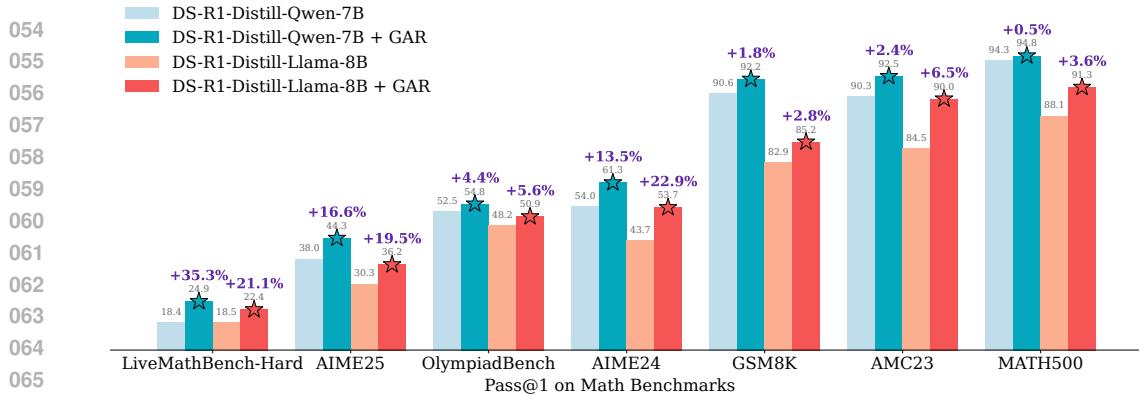


Figure 1: **Pass@1 accuracy on seven mathematical reasoning benchmarks.** Our Generative Adversarial Reasoner (GAR) consistently improves over strong baselines across both Deepseek-R1-Distill-Qwen-7B and Deepseek-R1-Distill-Llama-8B. GAR achieves gains of +22.9% on AIME24 and +19.5% on AIME25 for the Llama backbone, as well as +35.3% on LiveMathBench-Hard for Qwen. These results demonstrate the robustness and generality of GAR in enhancing reasoning performance across diverse mathematical tasks (Tab. 1).

lower annotation costs and increased robustness to label noise and reward mis-specification. Concretely, we optimize the LLM reasoner and an LLM-based discriminator together: the discriminator judges the logical soundness of each intermediate reasoning step and explains its judgment, while the reasoner learns to produce steps the discriminator consistently endorses for valid logic. This co-adaptation dynamically aligns the reward signal with the model’s evolving capabilities, reduces reliance on costly fine-grained annotations, and mitigates miscalibration by continually recalibrating the discriminator to the reasoner’s step distribution. As a result, we obtain better-calibrated, on-policy stepwise judgments, thereby enhancing the reasoning capabilities (Fig. 1).

However, jointly training the LLM reasoner and its stepwise discriminator introduces several challenges. First, stepwise analysis over long, intricate reasoning chains during training increases computational cost and system complexity. Second, ensuring the discriminator’s judgments are rigorous and interpretable requires explicit scoring rubrics and structured rationales. Third, co-evolution can invite reward hacking: the discriminator may drift toward overly positive judgments, while the LLM may learn to produce reasoning steps that look plausible but are semantically shallow.

To address the challenges, we propose **Generative Adversarial Reasoner (GAR)**, which incorporates a compute-efficient review schedule and an adversarial co-training framework. Specifically, we partition each reasoning chain into logically complete slices of comparable length. The discriminator evaluates each slice for logical soundness and generates a concise, structured rationale, providing localized and verifiable feedback on specific reasoning errors. For learning, we jointly update the LLM reasoner and the discriminator in an adversarial reinforcement learning scheme inspired by GANs (Goodfellow et al., 2014). The reasoner is rewarded for logically consistent steps that lead to correct final answers. The discriminator receives two complementary rewards: an alignment reward for correctly detecting errors in the reasoning, and a discriminative reward for distinguishing the reasoner’s trajectories from reference rationales. Together, these signals improve sample efficiency, and deliver calibrated stepwise supervision under a controlled compute budget.

Compared with previous methods, our model offers three key advantages: (i) slice-level evaluation that simplifies the discriminator’s task and yields localized, interpretable justifications; (ii) on-policy joint updates that keep rewards aligned with the model’s current behavior and support continued improvement, with the discriminator evolving to detect subtler errors as the LLM grows stronger; and (iii) dense step-level rewards that augment sparse exact match grading with continuous signals based on the fraction of correct steps, improving credit assignment and sample efficiency.

Our experiments show that, even compared to strong baselines (DeepSeek-AI, 2025), we achieve further improvements across various mathematical reasoning benchmarks, delivering significant gains over standard RL post-training approaches. For instance, on AIME24, we boost DeepSeek-R1-Distill-Qwen-7B from 54.0 to 61.3 (+7.3) and DeepSeek-R1-Distill-Llama-8B from 43.7 to 53.7 (+10.0), with comparable training time. These results highlight the consistent and substantial enhancements in LLM reasoning with our GAR, achieved within a comparable compute budget.

108 

## 2 RELATED WORK

110 **Reinforcement Learning with Process Feedback.** Process-level supervision improves reasoning  
 111 by evaluating intermediate steps rather than outcomes (Lightman et al., 2023; Uesato et al., 2022;  
 112 Ouyang et al., 2022). Beyond costly human PRMs, automatic judges lower labeling cost but in-  
 113 troduce noise; methods include Monte-Carlo from final answers, LLM-as-judge, and formulating  
 114 reward modeling as next-token prediction to improve stability (Zhang et al., 2025; 2024; Gao et al.,  
 115 2024; Xia et al., 2025; Xi et al., 2024). Math verifiers complement PRMs and favor concise, struc-  
 116 tured rationales (Cobbe et al., 2021). RL methods such as DeepSeek-R1 (DeepSeek-AI, 2025) and  
 117 FastCuRL (Song et al., 2025) encourage behaviors like self-reflection and verification but still rely  
 118 on static rewards or fixed critics that drift with the policy. Our approach jointly trains the reasoner  
 119 and a discriminator under an on-policy scheme, yielding dense slice-level rewards with a compute-  
 120 efficient review schedule and alignment regularization to curb reward hacking, addressing PRM  
 121 mis-specification and noisy signals without sacrificing efficiency (Wen et al., 2024; Lv et al., 2025).

122 **Self-Play, Multi-Agent, and Game-Theoretic Training.** Self-play is a unifying mechanism: ad-  
 123 versarial dynamics induce curricula, enabling iterative improvement from self-generated data with  
 124 evolving critics (Chen et al., 2024; Yuan et al., 2024; Chen et al., 2025), while robustness-oriented  
 125 variants such as SPAG instantiate these dynamics via adversarial word games (Cheng et al., 2024).  
 126 Extending to the *zero-data* regime, recent work replaces external traces with task-space exploration  
 127 and language self-play (Zhao et al., 2025; Kuba et al., 2025a) A complementary line recasts the  
 128 adversary as agents whose discourse provides training signals, with debate-style setups improving  
 129 factuality and robustness (Du et al., 2023; Liang et al., 2023) To scale these dynamics, fully online  
 130 multi-agent RL systems implement distributed actor-learner frameworks with role-conditioned ad-  
 131 vantages to sustain open-ended curricula transferable to math and science, while Parallel-R1 specif-  
 132 ically targets parallel thinking (Liu et al., 2025b; Wan et al., 2025; Liu et al., 2025a; Zheng et al.,  
 133 2025). Distinct from these, we embed adversarial dynamics *inside* training by co-evolving the critic  
 134 with the policy to deliver fine-grained, on-policy credit assignment, consistent with recent insights  
 135 on exploration and entropy for reasoning RL (Cui et al., 2025; Cheng et al., 2025; Wang et al., 2025).

136 **Thinking Scaffolds and Curriculum Learning.** Reasoning can be scaffolded via distilled tem-  
 137 plates, uncertainty-aware planning, and staged curricula to stabilize early steps and diversify so-  
 138 lution paths (Yang et al., 2024; Hu et al., 2024; Zheng et al., 2025). Building on this, game-  
 139 theoretic formulations treat reasoning as an interactive, multi-round protocol; Game-of-Thought  
 140 shows that such interaction improves robustness and information seeking (Kempinski et al., 2025).  
 141 Complementing these macro-level curricula, verifier-guided scaffolds such as Math-Shepherd pro-  
 142 vide lightweight stepwise signals without human labels, reinforcing intermediate decisions at low  
 143 supervision costs (Wang et al., 2023). Our method internalizes game theory into a discriminator  
 144 and couples it with compute-efficient slice-level evaluation, providing dense, calibrated, on-policy  
 145 rewards that improve mathematical reasoning and code generation within a unified framework.

146 

## 3 GAR: GENERATIVE ADVERSARIAL REASONER

147 We propose a modular formulation for GAR consisting of two components: a *Reasoner*, denoted  
 148  $\mathcal{M}_r$ , which is a general-purpose LLM that generates reasoning processes and final answers based  
 149 on user input; and a *Discriminator*, denoted  $\mathcal{M}_d$ , which evaluates the outputs of  $\mathcal{M}_r$  slice by slice.  
 150 The two models are jointly trained via reinforcement learning. We provide detailed descriptions of  
 151 each model and the training procedure below.

152 **Reasoner.** The reasoner is implemented as an LLM that generates intermediate reasoning and final  
 153 answers. In principle, any model capable of step-by-step reasoning can serve as the reasoner. In  
 154 this work, to demonstrate the effectiveness of our approach, we instantiate the reasoner with several  
 155 state-of-the-art, open-source reasoning models (namely, variants of the official DeepSeek-R1-Distill  
 156 models (DeepSeek-AI, 2025)), and show that our framework further improves their performance.

157 **Discriminator.** The discriminator evaluates the quality and correctness of the reasoning process,  
 158 assigning a reward signal to each generated response. In our implementation, it is instantiated as a  
 159 smaller, pre-trained variant of the reasoner.

160 However, holistic evaluation of the entire reasoning trace with the discriminator often fails to yield  
 161 reliable results. We hypothesize that lengthy and complex reasoning chains, which may span thou-

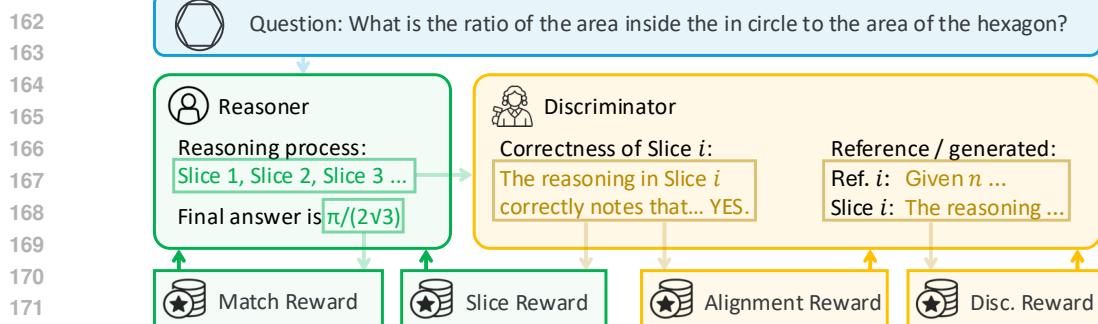


Figure 2: **GAR architecture.** GAR is a reinforcement learning framework that jointly trains an LLM reasoner and a slice-level discriminator to improve reasoning accuracy and explainability. Unlike standard RL for LLMs, which computes reward signals mainly based on final answers, GAR leverages a discriminator to provide dense, slice-level rewards that evaluate the reasoning process at each intermediate step. More importantly, the discriminator is continuously co-evolved with the LLM reasoner, ensuring the reward signal remains aligned with the model’s current behavior and enabling sustained improvement and refinement of reasoning.

sands of tokens, are difficult for language models to process and evaluate faithfully, hindering precise localization of reasoning errors. To mitigate this, we partition the generated reasoning into shorter, logically coherent slices, balancing slice length and semantic completeness. Specifically, we segment the reasoning trajectory based on delimiters, then merge adjacent segments until a clear new semantic beginning is identified or a predefined token length  $L = 320$  is reached. For each slice  $i$ , the discriminator assigns a binary slice reward  $r_i^s \in \{0, 1\}$  to evaluate its reasoning quality, where  $r_i^s = 1$  indicates that the slice is logically sound. The overall reward is then computed as the mean of all slice-level scores:  $R^s = \frac{1}{n} \sum_{i=1}^n r_i^s$ .

This slice-level reward mechanism offers two main advantages. *First, it improves reliability:* assessing the correctness and internal consistency of short slices is substantially easier and more accurate than evaluating a lengthy reasoning chain in its entirety. *Second, it provides a denser and more informative training signal than simple answer matching:* rather than a single binary label on the final answer, it scores the reasoning trajectory at the slice level and aggregates these scores into a fine-grained reward. Consequently, even when all final answers are wrong, the model can differentiate and reinforce better reasoning paths during RL training, improving sample efficiency and mitigating the problem of reward sparsity.

**Reward Functions.** As illustrated in Fig. 2, we jointly train the reasoner and the discriminator. For the reasoner, we use Group Relative Policy Optimization (GRPO) (Shao et al., 2024) with a reward that linearly combines (i) an exact-match term  $R^m \in \{0, 1\}$ , which compares the final answer to the ground truth, and (ii) the continuous reward  $R^s \in [0, 1]$  from the discriminator (the mean of the slice-level scores). The overall reasoner reward is defined as  $R^{\text{rea}} = \lambda_1 R^m + \lambda_2 R^s$ , where  $\lambda_1, \lambda_2 \geq 0$  are hyperparameters that weight the two components.

For the discriminator, we maximize two terms: a discriminator reward  $R^d$  and an alignment reward  $R^a$ . The discriminator reward  $R^d$  follows the standard GAN objective (Goodfellow et al., 2014):

$$R^d = \mathbb{E}_{x \sim p_{\text{ref}}} [\log \mathcal{M}_d(x)] + \mathbb{E}_{x \sim p_{\text{gen}}} [\log(1 - \mathcal{M}_d(x))]$$

where  $\mathcal{M}_d(x)$  represents the discriminator’s estimated probability that slice  $x$  is real, and  $p_{\text{ref}}, p_{\text{gen}}$  denote the distributions of reference reasoning slices and model-generated reasoning slices, respectively. The alignment reward  $R^a$  quantifies the mean agreement between the discriminator’s slice-level scores  $r^s$  and the correctness of the final answer produced by the entire reasoning sequence. Under the hypothesis that correct answers are more likely to be supported by logically sound reasoning, this term encourages consistency between slice-level evaluation and answer-level correctness. The total discriminator reward is given by  $R^{\text{dis}} = \lambda_3 R^d + \lambda_4 R^a$ , where  $\lambda_3, \lambda_4 \geq 0$  control the relative contributions. This joint training encourages the discriminator to provide calibrated, task-aligned feedback while the reasoner improves both reasoning quality and answer accuracy.

**Training Procedure.** For each batch of questions, the reasoner generates both answers and detailed reasoning steps, which we segment into multiple slices. We then mix these generated slices with an

216 equal number of reference slices to form a balanced set and train the discriminator to distinguish  
 217 between them. The discriminator scores each slice; these scores provide the slice reward  $R^s$  for  
 218 the reasoner and contribute to the discriminator’s own objectives ( $R^d$  and  $R^a$ ). We jointly update  
 219 both models with their respective objectives and iterate, yielding improvements in reasoning quality,  
 220 answer accuracy, and the discriminator’s evaluation accuracy.

221 In addition, it is known that generating the reasoning process enhances LLMs’ capabilities in han-  
 222 dling complex tasks (Wei et al., 2022). However, this process can be computationally expensive for  
 223 our task, as it requires analyzing each slice of the entire reasoning chain, potentially resulting in tens  
 224 of slices per question. To improve efficiency, we modify the discriminator’s workflow to (i) briefly  
 225 analyze the reasoning chain, (ii) provide the evaluative judgment (slice reward  $r_i^s$ ), and (iii) provide  
 226 a concise rationale for its assessment, rather than generating a full reasoning chain before scoring.  
 227 The rationale provided after the judgment is used mainly for explainability. During training, the  
 228 discriminator is prompted to generate the analysis, rating (slice reward  $r_i^s$ ), and rationale in a single  
 229 response, but with the maximum generation length limited to  $K = 128$  tokens to curtail the ratio-  
 230 nade and accelerate training. Notably, results in Sec. 4.3 indicate that restricting the discriminator’s  
 231 response at 128 tokens does not degrade performance: the final results remain comparable to those  
 232 with unrestricted response lengths, while substantially accelerating training.

233 To further improve the evaluation accuracy, especially after switching the discriminator to an analy-  
 234 sis-score–rationale format, we introduce a supervised fine-tuning (SFT) stage for the discriminator.  
 235 In this stage, we use a pre-trained LLM to generate reasoning steps on a small subset of the training  
 236 data. These reasoning steps are then evaluated by GPT-o4-mini, which provides a brief analysis,  
 237 an evaluative judgment, and a concise rationale for each example. To build a balanced SFT dataset  
 238 and mitigate bias, we randomly sample equal numbers of examples labeled ‘yes’ and ‘no’, ensuring  
 239 both classes are equally represented. We fine-tune the discriminator on this data with early stopping,  
 240 enabling it to adapt to the new format while preserving the capabilities of the original model.

241 In summary, training proceeds in two stages: (1) SFT of the discriminator to adapt it to the evaluation  
 242 format, and (2) joint optimization of the reasoner and discriminator with GRPO. At inference time,  
 243 only the LLM reasoner is used to produce answers, following the standard inference procedure.

## 244 4 EXPERIMENTS

246 This section presents a comprehensive evaluation of the mathematical reasoning capabilities of our  
 247 model. Sec. 4.1 outlines the experimental setup. Sec. 4.2 demonstrates that GAR significantly im-  
 248 proves over state-of-the-art models on mathematical benchmarks. Sec. 4.3 provides detailed analy-  
 249 ses of GAR, and Sec. 4.4 reports ablation studies. Finally, Sec. 4.5 discusses potential applications  
 250 and usage of our GAR.

### 252 4.1 EXPERIMENTAL SETUP

254 **Model Details.** Our implementation builds on OpenR1 (Hugging Face, 2025) and vLLM (Kwon  
 255 et al., 2023), and we evaluate two backbones. For the Qwen-based setup (Qwen Team, 2024),  
 256 we use DeepSeek-R1-Distill-Qwen-7B as the reasoner and DeepSeek-R1-Distill-Qwen-1.5B as the  
 257 discriminator. For the Llama-based setup (Dubey et al., 2024), we use DeepSeek-R1-Distill-Llama-  
 258 8B for both the reasoner and the discriminator, as no smaller Llama reasoning variant is available.

259 **Tasks and Benchmarks.** We evaluate GAR on mathematical reasoning tasks across seven public  
 260 benchmarks: AIME 2024/2025 (MAA, a), MATH500 (Hendrycks et al., 2021), GSM8K (Cobbe  
 261 et al., 2021), AMC23 (MAA, b), and LiveMathBench (Liu et al., 2024). For LiveMathBench, we  
 262 evaluate its hard splits (v202505\_all.en and v202412\_hard.en) and report the average performance.  
 263 For all evaluations, we adopt Pass@1 accuracy (averaged over 30 samples) as the metric, and fix the  
 264 decoding parameters to temperature = 0.6, top p = 0.95, and max tokens = 32K.

265 **Dataset.** All experiments are conducted using the OpenR1-Math-220k dataset from the OpenR1  
 266 project. To construct instruction-tuning data for the discriminator, we randomly sample 10% of the  
 267 training set, partition the dataset-provided DeepSeek-R1 chains of thought into slices, and annotate  
 268 each slice with binary (yes/no) judgments evaluating its soundness, along with brief rationales, using  
 269 the GPT-o4-mini API. To mitigate class imbalance in these judgments, we downsample the majority  
 class to achieve a 1:1 label ratio.

270 **Table 1: Pass@1 accuracy on mathematical reasoning benchmarks.** Reported scores are aver-  
 271 aged over 30 runs per benchmark to reduce evaluation noise.

Model	AIME24	AIME25	MATH500	GSM8K	AMC23	Olympiad Bench	LiveMath Bench-Hard
DS-R1-Distill-Qwen-7B + GAR (Ours)	54.0 61.3 (+7.3)	38.0 44.3 (+6.3)	94.3 94.8 (+0.5)	90.6 92.2 (+1.6)	90.3 92.5 (+2.2)	52.5 54.8 (+2.3)	18.4 24.9 (+6.5)
DS-R1-Distill-Llama-8B + GAR (Ours)	43.7 53.7 (+10.0)	30.3 36.2 (+5.9)	88.1 91.3 (+3.2)	82.9 85.2 (+2.3)	84.5 90.0 (+5.5)	48.2 50.9 (+2.7)	18.5 22.4 (+3.9)

278 **Training Details.** The discriminator is first instruction-tuned with AdamW (Loshchilov & Hutter,  
 279 2017) for 500 steps (learning rate  $1 \times 10^{-4}$ , 100 warm-up steps, weight decay 0.0001) using a global  
 280 batch size of 128 on 8 H100 GPUs. Then for adversarial reinforcement learning, we jointly optimize  
 281 the LLM reasoner and discriminator for 400 steps with AdamW (initial learning rate  $1 \times 10^{-6}$  with  
 282 a 10% warm-up and cosine learning rate decay to  $5 \times 10^{-7}$ ), using a global batch size of 192 on 8  
 283 H100 GPUs. Reward weights are set to  $\lambda_1 = \lambda_2 = \lambda_3 = 1$  and  $\lambda_4 = 0.5$ .

## 285 4.2 ADVANCING STATE-OF-THE-ART MODELS ON MATHEMATICAL REASONING

287 Table 1 summarizes the Pass@1 accuracy of our method compared to strong baselines across  
 288 diverse mathematical reasoning benchmarks. All results are averaged over 30 trials per bench-  
 289 mark, ensuring reliability by conducting three independent training runs and evaluating with 10  
 290 inference seeds per run. To ensure fair comparison, we re-evaluate all baselines under a unified  
 291 evaluation protocol (Habib et al., 2023) to eliminate scripting variance. Despite starting from  
 292 strong baselines, our method demonstrates consistent improvements across all benchmarks, espe-  
 293 cially on challenging datasets such as AIME24, AIME25, and LiveMathBench-Hard. For example,  
 294 our approach improves the accuracy of DeepSeek-R1-Distill-Qwen-7B on AIME24 by 7.3 and on  
 295 LiveMathBench-Hard by 6.5, and achieves an even larger improvement of 10.0 on AIME24 when  
 296 applied to DeepSeek-R1-Distill-Llama-8B. These results highlight our method’s effectiveness in  
 297 addressing difficult reasoning tasks. In particular, *the discriminator model plays a crucial role by*  
 298 *supervising reasoning traces, thereby enhancing the system’s ability to solve complex questions with*  
 299 *greater accuracy.* Beyond challenging datasets, our approach achieves notable improvements across  
 300 all benchmarks, highlighting its versatility and robustness in enhancing reasoning capabilities.

## 301 4.3 ANALYSES AND DISCUSSIONS

303 This section presents a detailed analysis and discussion of the proposed method.

305 **Slice-Level Feedback from Our Discriminator.** Table 2 presents training-time examples of the  
 306 LLM reasoning slices and our discriminator’s judgments. GAR provides concise, structured assess-  
 307 ments of each slice’s soundness, yielding localized, checkable feedback. The discriminator is able  
 308 to affirm correct algebraic and logical transformations, flag subtle arithmetic slips and flawed  
 309 reasoning steps, and identify the exact symbols or steps responsible, while keeping rationales brief to  
 310 control cost. This dense, slice-level supervision localizes errors early, improves credit assignment,  
 311 and yields better learning than sparse outcome-only rewards.

### 312 **Discriminator with Improved Efficiency.** To

313 obtain accurate and comprehensive analysis,  
 314 our discriminator evaluates each slice of the full  
 315 reasoning chain, which can yield tens of slices  
 316 per question and increase review costs. To im-  
 317 prove efficiency, we modify the discriminator

318 workflow as shown in Table 2: the discriminator first gives a brief analysis, then a binary ‘yes/no’  
 319 verdict on the slice’s soundness, and finally a concise rationale. During joint training of the LLM  
 320 reasoner and the discriminator, this design yields reliable judgments without requiring a full ra-  
 321 tionale. Specifically, we cap the discriminator’s output at 128 tokens, preserving the verdict and  
 322 truncating any justification beyond that limit. Table 3 compares three training settings: standard RL  
 323 without discriminator, discriminator with truncation, and discriminator without truncation. Apply-  
 324 ing the cap preserves accuracy while significantly improving training efficiency. It demonstrates that  
 325 our analysis-score–rationale design delivers dense supervision with minimal overhead.

Table 3: Rollout speed.

	AIME24	Training Time (hours)
Standard RL	56.3	16
GAR w/ truncation	61.3	19
GAR w/o truncation	60.8	43

324 **Table 2: Training-time examples of LLM reasoning slices and our discriminator judgments.**  
 325 We underline the discriminator’s key analysis leading to its judgment. Our model provides localized,  
 326 interpretable justifications for each reasoning slice. Additional examples in Appendix B.

328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	524	525	526	527	528	529	530	531	532	533	534	535	536	537	538	539	540	541	542	543	544	545	546	547	548	549	550	551	552	553	554	555	556	557	558	559	560	561	562	563	564	565	566	567	568	569	570	571	572	573	574	575	576	577	578	579	580	581	582	583	584	585	586	587	588	589	590	591	592	593	594	595	596	597	598	599	600	601	602	603	604	605	606	607	608	609	610	611	612	613	614	615	616	617	618	619	620	621	622	623	624	625	626	627	628	629	630	631	632	633	634	635	636	637	638	639	640	641	642	643	644	645	646	647	648	649	650	651	652	653	654	655	656	657	658	659	660	661	662	663	664	665	666	667	668	669	670	671	672	673	674	675	676	677	678	679	680	681	682	683	684	685	686	687	688	689	690	691	692	693	694	695	696	697	698	699	700	701	702	703	704	705	706	707	708	709	710	711	712	713	714	715	716	717	718	719	720	721	722	723	724	725	726	727	728	729	730	731	732	733	734	735	736	737	738	739	740	741	742	743	744	745	746	747	748	749	750	751	752	753	754	755	756	757	758	759	760	761	762	763	764	765	766	767	768	769	770	771	772	773	774	775	776	777	778	779	780	781	782	783	784	785	786	787	788	789	790	791	792	793	794	795	796	797	798	799	800	801	802	803	804	805	806	807	808	809	810	811	812	813	814	815	816	817	818	819	820	821	822	823	824	825	826	827	828	829	830	831	832	833	834	835	836	837	838	839	840	841	842	843	844	845	846	847	848	849	850	851	852	853	854	855	856	857	858	859	860	861	862	863	864	865	866	867	868	869	870	871	872	873	874	875	876	877	878	879	880	881	882	883	884	885	886	887	888	889	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	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	972	973	974	975	976	977	978	979	980	981	982	983	984	985	986	987	988	989	990	991	992	993	994	995	996	997	998	999	1000
1	Given $n = (a - b)(a + b)$ , and both $a - b$ and $a + b$ are either both even or both odd. If $n$ is odd, then both $a - b$ and $a + b$ are odd, and their product is $n$ . So, any odd number can be expressed as such. If $n$ is even, then both $a - b$ and $a + b$ must be even, so $n$ must be divisible by 4. Let's write $n = 4k$ , where $k$ is a positive integer. Then, $a - b = 2m$ and $a + b = 2n$ , where $m \times n = k$ . Thus, $a = m + n$ and $b = n - m$ . For $b$ to be positive, $n > m$ , and both $m$ and $n$ must be positive integers. Therefore, for $n$ to be a wise number, if $n$																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																															

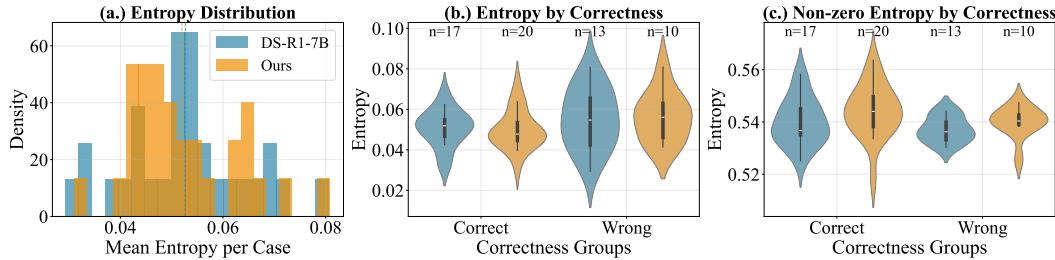


Figure 3: **Improving model without entropy collapse.** Compared to DeepSeek-R1-Distill-Qwen-7B, our method raises AIME24 accuracy (+7.3), but maintains a comparable overall mean-entropy distribution (5.20% vs. 5.27%) (a) and tightens the “wrong” distribution (b), indicating better calibration with fewer extreme-uncertainty failures. Removing zero-entropy tokens (c) flips the ordering (entropy is higher on correct cases), revealing a selective-entropy behavior (decisive on deterministic spans, exploratory on decision-critical tokens) that aligns with the AIME24 accuracy gains.

Table 4: **Ablation study.** Beginning with the baseline (DeepSeek-R1-Distill-Qwen-7B), we verify the effectiveness of each component.

Model	Exact Match Grading $\mathcal{R}^m$	Judger Score $\mathcal{R}^g$	Alignment Reward $\mathcal{R}^g$	Discriminator Reward $\mathcal{R}^g$	AIME24	AIME25
1 Baseline (BL)					54.0	38.0
2 BL + Standard RL	✓				56.3	40.7
3 BL + Fixed Standard Critic	✓	✓			56.7	40.4
4 BL + Fixed GAR Discriminator	✓	✓			58.6	42.0
5 BL + Trainable GAR Discriminator	✓	✓	✓		59.4	42.8
6 BL + Trainable GAR Discriminator	✓	✓		✓	60.2	43.3
7 BL + Trainable GAR Discriminator	✓	✓	✓	✓	61.3	44.3

pattern explains the observed accuracy gains on AIME24 and suggests a practical control signal: token- or slice-level entropy can trigger self-checks or adaptive sampling precisely where uncertainty is concentrated, improving both efficiency and reliability.

#### 4.4 ABLATION STUDY

We conduct various ablations in Table 4, progressively building on the baseline DeepSeek-R1-Distill-Qwen-7B to validate each component and culminate in our final model (row 7).

**Discriminator Design Analysis.** We ablate the discriminator in Table 4 (rows 2 – 4). Row 2 is a baseline LLM fine-tuned with standard GRPO using an outcome-based exact-match reward, without any discriminator. Row 3 adds a fixed standard critic (DeepSeek-R1-Distill-Qwen-1.5B) to provide feedback. Row 4 keeps the LLM critic’s capacity fixed, instruction-tunes it to our slice-level judgment format with brief rationales, and deploys it under the compute-efficient review schedule.

The table shows that row 3 improves over row 2 on AIME24, confirming the benefit of adding a discriminator. Moreover, Row 4 consistently outperforms both, indicating that the discriminator drives the gains. By reframing the discriminator’s role from holistic solution grading to slice-level soundness judgments with concise rationales, we obtain more accurate and interpretable feedback. The resulting dense, slice-level rewards provide continuous learning signals compared with sparse exact-match grading, improving credit assignment and sample efficiency, and thereby significantly boosting performance across benchmarks.

**Reward for Discriminator Training.** We ablate the discriminator’s reward design in Table 4 (rows 5 - 7). Both the alignment and discriminator rewards individually improve performance over the baseline. Combining them yields the best results, indicating the two signals are complementary. The alignment term sharpens the discriminator’s ability to distinguish correct from incorrect reasoning, but its supervision can be noisy because it depends on the correctness of that step’s generated final answer. The discriminator term stabilizes learning by steering the discriminator toward reference judgments. Together, these complementary signals yield a stronger and more reliable training signal.

**Effectiveness of Joint Training.** Finally, we evaluate joint training of the LLM reasoner and the discriminator in Table 4 (rows 4 and 7). Compared with using a fixed discriminator (row 4), joint on-policy updates (row 7) yield consistent gains by keeping rewards aligned with the reasoner’s current

432 behavior. As the reasoner improves, the co-trained discriminator adapts to detect subtler errors and  
 433 provides more informative slice-level feedback, which raises the performance ceiling and mitigates  
 434 drift or overfitting to a static reward signal.  
 435

#### 436 4.5 GAR UNLOCKS NEW APPLICATIONS AND FUTURE DIRECTIONS

438 Finally, we briefly discuss novel use cases and capabilities enabled by our method in this section.  
 439

440 **RL without Full Chain-of-Thought or Verifiable**  
 441 **Final Answers.** Another advantage of our approach  
 442 is that it decouples RL post-training from judgeable  
 443 final answers. Standard RL-based post-training re-  
 444 quires generating a complete chain of thought and  
 445 an automatically verifiable final answer. This makes  
 446 training much slower than supervised fine-tuning  
 447 (which only predicts the next token) and restricts applicability to tasks with clear evaluators (e.g.,  
 448 requiring an execution engine for code generation and struggling with open-ended math proofs).  
 449 In contrast, our model provides additional reward signals, enabling us to remove the final-answer  
 450 reward and update the model solely using our discriminator’s scores on intermediate reasoning. We  
 451 demonstrate this advantage in Table 5. Rather than generating a complete reasoning trajectory and  
 452 a final answer, we stop after three reasoning slices and have the discriminator evaluate these partial  
 453 traces, providing dense early feedback without a full chain of thought. This yields substantial effi-  
 454 ciency gains while improving accuracy: our method surpasses standard RL with significantly less  
 455 training time. Moreover, because it does not rely on final-answer rewards or external executors, it  
 456 naturally extends to tasks with hard-to-evaluate outputs, such as mathematical proofs.  
 457

458 **Distilling Reasoning Patterns with GAR.** GANs have  
 459 been widely used for distilling patterns in tasks like im-  
 460 age generation (Sauer et al., 2024). Similarly, we show  
 461 that GAR enables reasoning distillation, aligning a stu-  
 462 dent model’s reasoning pattern with that of a teacher.  
 463

464 In a demo experiment, we use the S1K-1.1 (Muennighoff  
 465 et al., 2025) dataset from the OpenR1 project, containing  
 466 two reasoning trajectories, “gemini\_thinking\_trajectory”  
 467 and “deepseek\_thinking\_trajectory”, to train the discriminator to distinguish between the two styles.  
 468 We then use this discriminator to jointly train the reasoner and discriminator within our GAR frame-  
 469 work. Notably, the reasoner is only trained on the Math220K dataset, without exposure to the  
 470 Gemini trajectory during training. In the evaluation, human experts are asked to differentiate be-  
 471 tween the generated and Gemini trajectories in a randomized side-by-side comparison. As shown  
 472 in Table 6, our model significantly increases similarity to the Gemini reasoning style, reducing the  
 473 distinguishability success rate from 82.3% to 55.9% (close to 50% random guess baseline), making  
 474 it substantially harder for experts to tell the two apart. Experimental details are included in the Ap-  
 475 pendix A. A natural extension of GAR’s distillation ability is human preference alignment, where  
 476 teacher reasoning comes from human explanations, enabling alignment with human-like reasoning.  
 477 We leave empirical validation to future work.  
 478

## 479 5 CONCLUSION

480 In this paper, we presented an adversarial co-training framework that couples an LLM Reasoner  
 481 with an LLM-based Discriminator to deliver dense, calibrated, slice-level rewards that supplement  
 482 sparse exact-match grading. Our model partitions reasoning into logically complete slices, and the  
 483 Discriminator provides concise and structured feedback. During training, the reasoner is rewarded  
 484 for logically consistent steps that lead to correct answers, and the Discriminator is rewarded for  
 485 correctly detecting reasoning errors. This design reduces annotation burden, mitigates reward mis-  
 486 specification and reward hacking, and improves credit assignment and sample efficiency under a  
 487 controlled compute budget. Empirically, the approach yields consistent gains over strong RL base-  
 488 lines on mathematical tasks, including +7.3 on AIME24 for DeepSeek-R1-Distill-Qwen-7B and  
 489 +10.0 for DeepSeek-R1-Distill-Llama-8B, along with better calibration and fewer extreme failures.  
 490

Table 5: **Partial-trace evaluation without a final-answer reward yields faster training and higher accuracy than standard RL.**

	AIME24	Training Time (hours)
Standard RL	56.3	16
Ours (3 slices)	57.7	6

Table 6: **Distinguishability of reasoning patterns.** We report the success rate of human experts distinguishing generated reasoning from Gemini reasoning before and after GAR training.

	w/o GAR	w/ GAR
Success rate	82.3%	55.9%

486 REFERENCES  
487

488 Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Ale-  
489 man, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical  
490 report. *arXiv preprint arXiv:2303.08774*, 2023.

491 Jiaqi Chen, Bang Zhang, Ruotian Ma, Peisong Wang, Xiaodan Liang, Zhaopeng Tu, Xiaolong Li,  
492 and Kwan-Yee K Wong. Spc: Evolving self-play critic via adversarial games for llm reasoning.  
493 *arXiv preprint arXiv:2504.19162*, 2025.

494 Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. Self-play fine-tuning  
495 converts weak language models to strong language models. *arXiv preprint arXiv:2401.01335*,  
496 2024.

497 Daixuan Cheng, Shaohan Huang, Xuekai Zhu, Bo Dai, Wayne Xin Zhao, Zhenliang Zhang, and  
498 Furu Wei. Reasoning with exploration: An entropy perspective. *arXiv preprint arXiv:2506.14758*,  
499 2025.

500 Pengyu Cheng, Yong Dai, Tianhao Hu, Han Xu, Zhisong Zhang, Lei Han, Nan Du, and Xiaolong Li.  
501 Self-playing adversarial language game enhances llm reasoning. *Advances in Neural Information  
502 Processing Systems*, 37:126515–126543, 2024.

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

506 Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen  
507 Fan, Huayu Chen, Weize Chen, et al. The entropy mechanism of reinforcement learning for  
508 reasoning language models. *arXiv preprint arXiv:2505.22617*, 2025.

509 DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning,  
510 2025. URL <https://arxiv.org/abs/2501.12948>.

511 Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving fac-  
512 tuality and reasoning in language models through multiagent debate. In *Forty-first International  
513 Conference on Machine Learning*, 2023.

514 Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha  
515 Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models.  
516 *arXiv e-prints*, pp. arXiv–2407, 2024.

517 Bofei Gao, Zefan Cai, Runxin Xu, Peiyi Wang, Ce Zheng, Runji Lin, Keming Lu, Dayiheng Liu,  
518 Chang Zhou, Wen Xiao, et al. Llm critics help catch bugs in mathematics: Towards a better  
519 mathematical verifier with natural language feedback. *arXiv preprint arXiv:2406.14024*, 2024.

520 Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair,  
521 Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information  
522 processing systems*, 27, 2014.

523 Nathan Habib, Clémentine Fourrier, Hynek Kydlíček, Thomas Wolf, and Lewis Tunstall. Light-  
524 eval: A lightweight framework for llm evaluation, 2023. URL [https://github.com/  
525 huggingface/lighteval](https://github.com/huggingface/lighteval).

526 Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song,  
527 and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv  
528 preprint arXiv:2103.03874*, 2021.

529 Zhiyuan Hu, Chumin Liu, Xidong Feng, Yilun Zhao, See-Kiong Ng, Anh Tuan Luu, Junxian He,  
530 Pang Wei Koh, and Bryan Hooi. Uncertainty of thoughts: Uncertainty-aware planning enhances  
531 information seeking in large language models. *arXiv preprint arXiv:2402.03271*, 2024.

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

540 Benjamin Kempinski, Ian Gemp, Kate Larson, Marc Lanctot, Yoram Bachrach, and Tal Kachman.  
 541 Game of thoughts: Iterative reasoning in game-theoretic domains with large language models.  
 542 2025.

543 Jakub Grudzien Kuba, Mengting Gu, Qi Ma, Yuandong Tian, and Vijai Mohan. Language self-play  
 544 for data-free training. *arXiv preprint arXiv:2509.07414*, 2025a.

545 Jakub Grudzien Kuba, Mengting Gu, Qi Ma, Yuandong Tian, and Vijai Mohan. Language self-play  
 546 for data-free training, 2025b. URL <https://arxiv.org/abs/2509.07414>.

547 Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E.  
 548 Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model  
 549 serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating  
 550 Systems Principles*, 2023.

551 Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming  
 552 Shi, and Zhaopeng Tu. Encouraging divergent thinking in large language models through multi-  
 553 agent debate. *arXiv preprint arXiv:2305.19118*, 2023.

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

557 Bo Liu, Leon Guertler, Simon Yu, Zichen Liu, Penghui Qi, Daniel Balcells, Mickel Liu, Cheston  
 558 Tan, Weiyan Shi, Min Lin, et al. Spiral: Self-play on zero-sum games incentivizes reasoning via  
 559 multi-agent multi-turn reinforcement learning. *arXiv preprint arXiv:2506.24119*, 2025a.

560 Junnan Liu, Hongwei Liu, Linchen Xiao, Ziyi Wang, Kuikun Liu, Songyang Gao, Wenwei Zhang,  
 561 Songyang Zhang, and Kai Chen. Are your llms capable of stable reasoning? *arXiv preprint  
 562 arXiv:2412.13147*, 2024.

563 Mickel Liu, Liwei Jiang, Yancheng Liang, Simon Shaolei Du, Yejin Choi, Tim Althoff, and Natasha  
 564 Jaques. Chasing moving targets with online self-play reinforcement learning for safer language  
 565 models. *arXiv preprint arXiv:2506.07468*, 2025b.

566 Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint  
 567 arXiv:1711.05101*, 2017.

568 Ang Lv, Ruobing Xie, Xingwu Sun, Zhanhui Kang, and Rui Yan. The climb carves wisdom deeper  
 569 than the summit: On the noisy rewards in learning to reason. *arXiv preprint arXiv:2505.22653*,  
 570 2025.

571 MAA. American invitational mathematics examination (aime). Mathematics Competition Series, a.  
 572 URL <https://maa.org/math-competitions/aime>. Accessed: October 5, 2023.

573 MAA. American mathematics competitions (amc 10/12). Mathematics Competition Series, b. URL  
 574 <https://maa.org/math-competitions/amc>. Accessed: October 5, 2023.

575 Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke  
 576 Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time  
 577 scaling. *arXiv preprint arXiv:2501.19393*, 2025.

578 Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong  
 579 Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to fol-  
 580 low instructions with human feedback. *Advances in neural information processing systems*, 35:  
 581 27730–27744, 2022.

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

583 Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion dis-  
 584 tillation. In *European Conference on Computer Vision*, pp. 87–103. Springer, 2024.

594 Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang,  
 595 Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathemati-  
 596 cal reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

597

598 Mingyang Song, Mao Zheng, Zheng Li, Wenjie Yang, Xuan Luo, Yue Pan, and Feng Zhang.  
 599 Fastcurl: Curriculum reinforcement learning with stage-wise context scaling for efficient train-  
 600 ing r1-like reasoning models. *arXiv preprint arXiv:2503.17287*, 2025.

601 Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia  
 602 Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process-and  
 603 outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

604

605 Ziyu Wan, Yunxiang Li, Xiaoyu Wen, Yan Song, Hanjing Wang, Linyi Yang, Mark Schmidt, Jun  
 606 Wang, Weinan Zhang, Shuyue Hu, et al. Rema: Learning to meta-think for llms with multi-agent  
 607 reinforcement learning. *arXiv preprint arXiv:2503.09501*, 2025.

608 Peiyi Wang, Lei Li, Zhihong Shao, RX Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang  
 609 Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *arXiv*  
 610 *preprint arXiv:2312.08935*, 2023.

611 Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen,  
 612 Jianxin Yang, Zhenru Zhang, et al. Beyond the 80/20 rule: High-entropy minority tokens drive  
 613 effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*, 2025.

614

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

618 Xueru Wen, Jie Lou, Yaojie Lu, Hongyu Lin, Xing Yu, Xinyu Lu, Ben He, Xianpei Han, Debing  
 619 Zhang, and Le Sun. Rethinking reward model evaluation: Are we barking up the wrong tree?  
 620 *arXiv preprint arXiv:2410.05584*, 2024.

621

622 Zhiheng Xi, Wenxiang Chen, Boyang Hong, Senjie Jin, Rui Zheng, Wei He, Yiwen Ding, Shichun  
 623 Liu, Xin Guo, Junzhe Wang, et al. Training large language models for reasoning through reverse  
 624 curriculum reinforcement learning. *arXiv preprint arXiv:2402.05808*, 2024.

625 Shijie Xia, Xuefeng Li, Yixin Liu, Tongshuang Wu, and Pengfei Liu. Evaluating mathematical  
 626 reasoning beyond accuracy. In *Proceedings of the AAAI Conference on Artificial Intelligence*,  
 627 volume 39, pp. 27723–27730, 2025.

628

629 Ling Yang, Zhaochen Yu, Tianjun Zhang, Shiyi Cao, Minkai Xu, Wentao Zhang, Joseph E Gonzalez,  
 630 and Bin Cui. Buffer of thoughts: Thought-augmented reasoning with large language models.  
 631 *Advances in Neural Information Processing Systems*, 37:113519–113544, 2024.

632

633 Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason  
 634 Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 3, 2024.

635

636 Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh  
 637 Agarwal. Generative verifiers: Reward modeling as next-token prediction. *arXiv preprint*  
 638 *arXiv:2408.15240*, 2024.

639

640 Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu,  
 641 Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical  
 642 reasoning. *arXiv preprint arXiv:2501.07301*, 2025.

643

644 Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Matthieu Lin, Shenzhi Wang, Qingyun  
 645 Wu, Zilong Zheng, and Gao Huang. Absolute zero: Reinforced self-play reasoning with zero  
 646 data. *arXiv preprint arXiv:2505.03335*, 2025.

647

Tong Zheng, Hongming Zhang, Wenhao Yu, Xiaoyang Wang, Xinyu Yang, Runpeng Dai, Rui Liu,  
 648 Huiwen Bao, Chengsong Huang, Heng Huang, et al. Parallel-r1: Towards parallel thinking via  
 649 reinforcement learning. *arXiv preprint arXiv:2509.07980*, 2025.

648 APPENDIX  
649650 In the appendix, we provide additional information as listed below:  
651

- 652 • Sec. A provides experimental details for reasoning distillation.
- 653 • Sec. B provides additional training-time examples of the LLM reasoning slices and our  
654 discriminator judgments.
- 655 • Sec. C lists the system prompts used for the reasoner and the discriminator.
- 656 • Sec. D provides additional experiments on coding ability.
- 657 • Sec. E demonstrates that GAR can further improve more advanced RL methods.
- 658 • Sec. F provides ablations on the slice segmentation design.
- 659 • Sec. G provides ablations on the discriminator size.
- 660 • Sec. H demonstrates that the discriminator reliably detects model correctness instead of  
661 incentivizing reward hacking.
- 662 • Sec. I discusses the limitations of our method.
- 663 • Sec. J provides the ethics statement.
- 664 • Sec. K provides the reproducibility statement.
- 665 • Sec. L reports the LLM usage.

669 A EXPERIMENTAL DETAILS FOR REASONING DISTILLATION.  
670671  
672 In the demo experiment, we use the S1K-1.1 (Muennighoff et al., 2025) dataset from the  
673 OpenR1 project, which contains two reasoning trajectories, "gemini\_thinking\_trajectory" and  
674 "deepseek\_thinking\_trajectory", to train a discriminator to distinguish between the two styles. We  
675 use the same settings as our main experiments and train the discriminator for 1,000 iterations. We  
676 then integrate this discriminator into our GAR framework and jointly train the reasoner and the dis-  
677 criminator under the Partial trace setting: instead of generating the full reasoning trajectory and a  
678 final answer, we stop after three reasoning slices and have the discriminator evaluate these partial  
679 traces. We train this model for 3,000 steps, and all other hyperparameters are the same as in the  
680 main experiment. The reasoner is trained only on the Math220K dataset, without exposure to the  
681 Gemini trajectory during training.682 For the human preference study, we generate 200 reasoning slices, pair them side by side with the  
683 corresponding Gemini slices, and randomly shuffle the order. We then ask 10 experts from three  
684 academic institutions, including 3 undergraduate students and 7 PhD students, to first familiarize  
685 themselves with the differences between Gemini and DeepSeek-R1 reasoning styles by reviewing  
686 the S1K-1.1 dataset, and then conduct the evaluation. We compare the success rate of experts dis-  
687 tinguishing generated reasoning from Gemini reasoning before and after GAR training, and report  
688 the results in Table 6.689 B SLICE-LEVEL FEEDBACK FROM OUR DISCRIMINATOR.  
690691 We provide additional examples of the LLM reasoning slices and our discriminator's judgments  
692 during joint training in Table 7 and Table 8.  
693694 C SYSTEM PROMPTS  
695696 We provide the system prompts for the LLM reasoner and the discriminator as follows:  
697698 System prompt for the LLM reasoner:  
699700 You are a helpful AI Assistant that provides well-reasoned and  
701 detailed responses. You first think about the reasoning process as  
an internal monologue and then provide the user with the answer.

702 Respond in the following format:<think>\n...\\n</think>\\n<answer>  
 703 \\n...\\n</answer>  
 704  
 705 System prompt for the LLM discriminator:  
 706  
 707 You are an evaluator responsible for assessing whether a reasoning  
 708 / thinking process is reasonable, rigorous, and accurate. Based on  
 709 these criteria, determine if the analysis is of high quality. First,  
 710 analyze the reasoning very briefly, then respond with '\*\*YES\*\*' for  
 711 high quality or '\*\*NO\*\*' if it is not. Finally, provide a brief but  
 712 specific explanation for your judgment. Hint: You can first summarize  
 713 the given thinking process to identify the main reasoning chain, then  
 714 analyze the reasoning chain sentence by sentence.  
 715

## D RESULTS ON CODING GENERATION

718 We further evaluate our model on coding-based reasoning tasks. As shown in Table 9, our method  
 719 yields substantial improvements across multiple coding benchmarks. To ensure statistical reliabil-  
 720 ity, all results are averaged over **30 trials per benchmark** (three independent training runs, each  
 721 evaluated with 10 inference seeds).

722 Experiments are conducted on the **CodeForces-CoT** dataset, where the (approximate) ground-truth  
 723 reasoning traces are produced by DeepSeek-R1. The maximum rollout length during training is  
 724 fixed at **800 steps**. The reward function is an **equal-weight combination** of three components:

- 726 • **CF-Code Reward**: Computes a weighted sum over public test cases, assigning 1 for pass  
 727 and 0 for fail.
- 728 • **Code-Format Reward**: Assigns a reward of 1 when the model output includes a valid  
 729 code-block wrapper (e.g., `` `python` `').
- 730 • **Critic Reward**: Incorporates alignment and discriminative terms, using the same structure  
 731 as in the math-reasoning setting.

734 Table 9: Pass@1 accuracy on coding benchmarks. Similar to the results reported in the main paper,  
 735 scores are averaged over 30 runs per benchmark to reduce evaluation noise. Our method significantly  
 736 improves coding performance.

737 Model	738 LiveCodeBench	739 HumanEval	740 HumanEval+
738 DS-R1-Distill-Qwen-7B	739 37.4	40.4	37.8
739 <b>DS-R1-Distill-Qwen-7B + GAR (Ours)</b>	740 <b>43.6</b>	<b>42.7</b>	<b>39.3</b>

## E COMPARISON TO RL-BASED APPROACHES AND COMPLEMENTARITY

745 Although several related approaches are also RL-based, they are **not directly comparable** to our  
 746 setting due to differences in base models, the presence or absence of explicit reasoning, task for-  
 747 mulations, and training strategies. For instance, SPIRAL(Liu et al., 2025a) frames training as a  
 748 zero-sum game for self-play and self-evolution, while LSP(Kuba et al., 2025b) employs an LLM to  
 749 alternate between a Challenger that generates new instructions and a Solver that attempts to follow  
 750 them for data-free training. These paradigms differ substantially from ours, making a fair head-to-  
 751 head comparison infeasible.

752 At the same time, our method is **complementary** to these approaches and can be incorporated on  
 753 top of stronger RL-style frameworks. We demonstrate this by considering a recent improvement to  
 754 GRPO, KL Cov(Cui et al., 2025). As shown in Table 10, adding GAR on top of KL Cov yields  
 755 further gains across all benchmarks. This suggests that our framework can enhance more advanced  
 RL-based reasoning algorithms as well.

756

757

Table 10: GAR further improves KL Cov (Enhanced GRPO Baseline).

758

759

	AIME24	AIME25	AMC	MATH-500	OlympiadBench
GRPO	21.2	9.6	58.7	78.8	40.7
GRPO + KL Cov(Cui et al., 2025)	22.6	12.9	61.4	80.8	42.6
<b>GRPO + KL Cov + GAR (ours)</b>	<b>25.7</b>	<b>16.2</b>	<b>62.5</b>	<b>84.2</b>	<b>44.0</b>

760

761

## F ABLATIONS ON SLICE SEGMENTATION DESIGN

### F.1 SEGMENTATION STRATEGY

In this section, we clarify the rationale behind adopting the proposed segmentation method based on **explicit delimiters** combined with a **token-length range**. Chain-of-thought (CoT) generated by contemporary reasoning models (e.g., DeepSeek-R1, Gemini) is typically organized into coherent fragments separated by line breaks (“\n”) and discourse cues such as “Wait,” “Since,” and “Therefore.” These natural markers provide reliable boundaries for forming semantically self-contained slices. Applying an additional token-length constraint prevents segments from being excessively short or long, while keeping computation and implementation overhead minimal.

Table 11 compares this rule-based method with two alternatives: (1) pure fixed-length token windows and (2) LLM-based semantic segmentation, where a model is prompted to automatically partition the CoT. The results show that the alternatives either disrupt coherent reasoning steps or require substantially higher training cost, without yielding performance improvements over our method.

Table 11: Ablation on different slice segmentation strategies.

Method	AIME24	Training Time (hours)
Pure fixed-length token windows	58.7	19
LLM-based semantic segmentation	61.6	35
<b>Ours</b>	<b>61.3</b>	19

### F.2 SENSITIVITY TO SLICE LENGTH

We further add a sensitivity analysis over slice length. As shown in Table 12, model performance is highest and most stable when slices contain approximately **320–560 tokens**. Shorter slices often contain no explicit reasoning error, making it difficult for the discriminator to learn informative supervision signals. In contrast, very long slices typically include at least one flaw, causing most segments to be labeled as “incorrect” and reducing label diversity, which in turn weakens the discriminator’s effectiveness.

Table 12: Ablation over slice length (tokens).

Slice Length	160	320	480	560	800	960	1120	1440
AIME24	57.4	61.3	61.5	61.4	61.0	59.3	56.5	56.8

## G ABLATIONS ON DISCRIMINATOR SIZE

In the Qwen configuration, we use a 1.5B discriminator because, after supervised fine-tuning (SFT), even this smaller model becomes a strong discriminator. As shown in Table 13, replacing the 1.5B discriminator with a 7B one yields better performance but also introduces substantially higher training cost. Due to compute constraints, we therefore adopt the 1.5B model as the default discriminator.

810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863

Table 13: Ablation on discriminator size for the Qwen setup.

Discriminator Size	AIME24	AIME25
1.5B	61.3	44.3
7B	62.5	45.2

For Llama, the **8B** model is the smallest publicly available Llama checkpoint with R1-style reasoning ability. Consequently, we do not have access to a smaller, compatible backbone for the discriminator. Therefore, for the Llama setup, we use the 8B model as both the generator and the discriminator.

## H REASONING CORRECTNESS UNDER CONTROLLED STYLE

Our “real slices” are reference CoTs sampled from strong reasoning models (DeepSeek-R1 or Gemini). While they naturally contain stylistic regularities, our reward design prevents the policy from exploiting style alone. In particular, the discriminator reward  $R^d$  is always paired with (i) task-level correctness rewards and (ii) KL regularization, ensuring that purely stylistic imitation is penalized whenever it harms correctness or formatting. Furthermore, within the discriminator,  $R^d$  is combined with the alignment reward  $R^a$ , anchoring slice judgments to correctness rather than surface patterns.

A central question is whether the discriminator learns *reasoning correctness* rather than merely *style*. To directly isolate this factor, we introduce a style-controlled evaluation (Table 14): for each reference CoT, we keep the overall writing style, structure, and tone fixed, while perturbing only intermediate numbers or local conclusions—producing logically incorrect but stylistically indistinguishable slices.

As shown in Table 14, the discriminator reliably differentiates logically correct from incorrect slices even under matched style, indicating that it learns to detect reasoning errors rather than mimic surface patterns.

Table 14: Style-controlled discrimination accuracy. Even when reasoning style and tone are held constant, the discriminator correctly identifies logical errors.

	Predicted Correct	Predicted Wrong
Actually correct	84.3%	15.7%
Actually wrong	23.3%	76.7%

## I LIMITATIONS AND FUTURE WORK

The proposed GAR has a few remaining limitations. First, it remains challenging to balance the discriminator’s reasoning depth with compute efficiency. In this work, we reformulate the think–answer format into an analysis–score–rationale format to make the discriminator more efficient while preserving its reasoning ability. Nevertheless, there is room to further improve compute-efficient reasoning. Promising directions include adaptive early-exit mechanisms, dynamic truncation of analysis conditioned on confidence, etc. Second, although we compute slice-level dense rewards, the final objective aggregates them into a single trajectory-level signal by averaging, which can dilute local credit assignment and increase variance. Better ways to leverage slice-wise information could improve training efficiency and stability.

## J ETHICS STATEMENT

This work seeks to improve model reasoning and training efficiency, but we acknowledge risks including bias amplification, spurious correlations, reward hacking, and potential misuse for generating harmful or deceptive content. We rely on publicly available datasets (Math-220K) and adhere to their licenses, avoid collecting or exposing personal data, and assess safety and fairness where

864 applicable, documenting observed failure modes. We will not release artifacts that materially in-  
865 crease misuse risk and provide usage guidelines to support responsible deployment. Downstream  
866 use should include human oversight, continuous monitoring, and domain-specific risk assessments.  
867

## 868 K REPRODUCIBILITY STATEMENT 869

870 We will release code, pretrained checkpoints, and detailed scripts to reproduce all results, including  
871 data preparation, training, and evaluation. The repository will provide exact configurations (model  
872 architectures, the analysis–score–rationale discriminator format, slice-level reward computation and  
873 aggregation, loss functions, optimizers, schedules), dataset splits and preprocessing, and the eval-  
874 uation harness with metrics. We document software and hardware environments (package versions,  
875 CUDA/cuDNN, GPU types), supply containerized setups (Conda/Docker), and fix random seeds;  
876 results are reported as mean over 30 runs. The implementation details are also provided in Sec. 4.1.  
877

## 878 L LLM USAGE 879

880 LLM is only used for limited writing assistance (clarity edits and grammar).  
881

882  
883  
884  
885  
886  
887  
888  
889  
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

919

920 **Table 7: Training-time examples of LLM reasoning slices and our discriminator judgments.**  
 921 We underline the discriminator’s key analysis leading to its judgment. Our model provides localized,  
 922 interpretable justifications for each reasoning slice.

923 LLM Reasoning Slices	924 Our Full Discriminator Judgments
<p>925 1 Case 1: <math>p</math> divides <math>2^s</math>. Since <math>2^s</math> is a power of 2, <math>p</math>    926 must be 2.    927 Case 2: <math>p</math> divides <math>q</math>. Since <math>q</math> is a prime, <math>p = q</math>.    928 Let’s explore each case.    929 **Case 1: <math>p = 2</math>**    930 Substituting <math>p = 2</math> into the equation, we get:  <math display="block">2^s q = 2^y - 1</math> <p>931 Rearranging:</p> <math display="block">2^y = 2^s q + 1</math> <p>932 Since <math>2^y</math> is even, the right-hand side must also be even. However, <math>2^s q</math> is even if <math>s \geq 1</math>, so <math>2^s q + 1</math> is odd. This is a contradiction because an even number cannot equal an odd number. Therefore, <math>p = 2</math> is not a valid solution.</p> </p>	<p>933 The reasoning examines the case where the prime <math>p</math> divides <math>2^s</math>. <u>It correctly concludes <math>p = 2</math>, substitutes into the equation <math>2^s q = 2^y - 1</math>, and argues that the left side <math>2^y</math> is even while the right side <math>2^s q + 1</math> is odd, leading to a contradiction.</u>    934 **YES**    935 The reasoning is rigorous and accurate. It systematically considers the case <math>p = 2</math>, substitutes it into the equation, and rearranges to <math>2^y = 2^s q + 1</math>. <u>It correctly observes that <math>2^y</math> is even, while <math>2^s q + 1</math> is odd (since <math>2^s q</math> is even for <math>s \geq 1</math>), leading to a contradiction.</u> The logic is clear, step-by-step, and based on fundamental parity properties, leaving no room for ambiguity. This is a high-quality analysis for this specific case.</p>
<p>936 2 Let me start with the first part. I know that a wise    937 number can be written as <math>a^2 - b^2</math> where <math>a</math> and <math>b</math> are positive integers. I can factor this expression as <math>(a-b)(a+b)</math>. So, any wise number can be expressed as the product of two integers <math>(a-b)</math> and <math>(a+b)</math>. Since <math>a</math> and <math>b</math> are positive integers, <math>a+b</math> must be greater than <math>a-b</math>, and both <math>a+b</math> and <math>a-b</math> must be positive integers. Moreover, both <math>a+b</math> and <math>a-b</math> must have the same parity because <math>a+b + (a-b) = 2a</math> must be even, meaning their sum is even. Therefore, both <math>a+b</math> and <math>a-b</math> must be either both even or both odd. This implies that their product <math>(a-b)(a+b)</math> must be either odd or divisible by 4. So, if a number is of the form <math>4k+2</math>, it cannot be expressed as the product of two integers with the same parity. For example, a number like 2 cannot be written as the difference of two squares because 2 is of the form <math>4k+2</math> (where <math>k = 0</math>), and there are no two integers <math>a</math> and <math>b</math> that would satisfy <math>a^2 - b^2 = 2</math>.</p>	<p>938 The reasoning shows that <math>a^2 - b^2 = (a-b)(a+b)</math> requires both factors to have the same parity, <u>so the product is either odd or divisible by 4</u>. Thus, <u>numbers of form <math>4k+2</math> (like 2) cannot be wise numbers.</u>    939 **YES**    940 It correctly identifies that a wise number is of the form <math>a^2 - b^2 = (a-b)(a+b)</math>, and notes that <math>a+b</math> and <math>a-b</math> must be positive integers with the same parity (both even or both odd) because their sum is <math>2a</math> (even). This leads to the key insight: <u>the product <math>(a-b)(a+b)</math> must be either odd or divisible by 4</u>. <u>Consequently, numbers of the form <math>4k+2</math> (which are even but not divisible by 4) cannot be expressed as such a product</u>. The example of 2 (which is <math>4 \times 0 + 2</math>) solidifies the argument. The logic is clear, step-by-step, and based on sound number theory principles.</p>
<p>941 3 But since this is an approximation, I need to check if 2686 or 2687 is a wise number. Let me check if 2686 is a wise number.    942 2686 divided by 2 is 1343, which is odd. So, 2686 is 2 times an odd number, which is not a multiple of 4. Therefore, 2686 is not a wise number because it cannot be expressed as the difference of squares.    943 Next, check 2687. Since 2687 is odd, it can be expressed as a difference of squares. So, 2687 is a wise number.    944 Therefore, the 2015th wise number is 2687.    945 Wait, but let me make sure. Let’s verify how many wise numbers are less than or equal to 2686 and 2687.</p>	<p>946 The reasoning checks if 2686 and 2687 are wise numbers, correctly using the parity/divisibility rule. <u>It then concludes 2687 is the 2015th wise number without verifying the count.</u>    947 **NO**    948 While it correctly identifies that 2686 (even but not divisible by 4) cannot be a wise number, and that 2687 (odd) can be a wise number, <u>it fails to verify whether 2687 is actually the 2015th wise number</u>. The reasoning jumps to a conclusion without calculating how many wise numbers exist up to 2686 and 2687. Specifically, <u>it does not account for the fact that not all odd numbers are necessarily wise (e.g., 1 is odd and wise, but other constraints might exist)</u> and, more importantly, <u>it does not compute the cumulative count of wise numbers to confirm the index</u>. This lack of systematic counting or verification makes the reasoning incomplete and potentially incorrect.</p>

971

972  
 973 **Table 8: Training-time examples of LLM reasoning slices and our discriminator judgments.**  
 974 We underline the discriminator’s key analysis leading to its judgment. Our model provides localized,  
 975 interpretable justifications for each reasoning slice.

976	977 LLM Reasoning Slices	978 Our Full Discriminator Judgments
979	1 But wait, I should double-check if this is indeed the minimum. Maybe I made a mistake in applying the Lagrange multipliers. Alternatively, perhaps there’s a geometric interpretation that can confirm this result.  Thinking geometrically, as I considered earlier, the expression represents the sum of distances from points on the coordinate axes to the point (12, 9). The minimal total distance should be the straight line distance from the origin to (12, 9), which is $\sqrt{12^2 + 9^2} = \sqrt{144 + 81} = \sqrt{225} = 15$ . That matches the result I got using calculus, so it seems correct.  So, the minimal value is indeed 15.	980 The reasoning cross-checks a calculus result (Lagrange multipliers) with a geometric interpretation: <u>the sum of distances from points on the axes to (12,9)</u> is minimized by the straight-line distance from the origin to (12,9), which is 15. The results match. **YES** The reasoning is rigorous and accurate. It cross-validates the result obtained from calculus (Lagrange multipliers) with a geometric interpretation: <u>the expression</u> $\sqrt{x^2 + 144} + \sqrt{(15 - x)^2 + 81}$ <u>represents the sum of distances from a point on the x-axis to fixed points, and the minimal path is indeed a straight line from the origin to (12,9), yielding a distance of 15</u> . This geometric insight is sound and confirms the calculus result, demonstrating thoroughness and multiple perspectives. The logic is clear and well-supported.
981	2 From this table, I notice a pattern. It seems that $f(n)$ alternates between 1 and 0, but it’s not straightforward. Let me try to see if there’s a base-3 pattern or something related to the recursive formula involving division by 3.  Looking at the recursive formula:	982 The reasoning observes a pattern in $f(n)$ and recalls the recursive definition $f(n) = 1 - f(n - 1) \cdot f(\lfloor n/3 \rfloor)$ . <u>It tests a simple hypothesis (based on divisibility by 3) but immediately notes counterexamples (like f(6)=1, f(9)=1), showing the pattern is more complex.</u> **NO** The reasoning is not rigorous or accurate. It attempts to identify a pattern (e.g., alternation or modulo 3 behavior) but quickly contradicts itself with counterexamples (e.g., $f(3) = 0$ , $f(6) = 1$ , $f(9) = 1$ ). <u>The analysis is superficial and does not deeply engage with the recursive formula. It mentions considering the recursion but does not systematically compute values or derive properties.</u>
983	3 In a geometric sequence, each term is obtained by multiplying the previous term by a constant ratio $r$ . Given $a_2 = 2$ , we have:  $a_2 = a_1 \times r \Rightarrow 2 = a_1 \times r \quad (1)$ Given $a_5 = \frac{1}{4}$ , we have:  $a_5 = a_1 \times r^4 \Rightarrow \frac{1}{4} = a_1 \times r^4 \quad (2)$ Dividing equation (2) by equation (1):  $\frac{\frac{1}{4}}{2} = \frac{a_1 \times r^4}{a_1 \times r} \Rightarrow \frac{1}{8} = r^3 \Rightarrow r = \sqrt[3]{\frac{1}{8}} = \frac{1}{2}$ Substituting $r = \frac{1}{2}$ into equation (1):  $2 = a_1 \times \frac{1}{2} \Rightarrow a_1 = 4$	984 The reasoning uses the geometric sequence definition to set up equations for $a_2$ and $a_5$ , solves for the ratio $r$ by dividing the equations, finds $r = 1/2$ , then solves for $a_1 = 4$ , and derives the general formula $a_n = 2^{3-n}$ . **YES** The reasoning is rigorous and accurate. It <u>correctly uses the definition of a geometric sequence</u> , sets up equations based on the given terms ( $a_2 = 2$ and $a_5 = 1/4$ ), and solves for the common ratio $r$ by dividing the equations. <u>The calculation</u> $\frac{\frac{1}{4}}{2} = \frac{1}{8} = r^3$ <u>is correct</u> , leading to $r = \frac{1}{2}$ . Then, it substitutes back to find $a_1 = 4$ . Finally, it expresses the general term $a_n$ in both exponential and simplified form ( $2^{3-n}$ ). <u>The steps are logical, clear, and mathematically sound, with no errors or gaps</u> . This is a high-quality analysis.