

ESCAPING THE VERIFIER: LEARNING TO REASON VIA DEMONSTRATIONS

Locke Cai
Together AI
Massachusetts Institute of Technology
lcai12@mit.edu

Ivan Provilkov
Together AI
ivan@together.ai

ABSTRACT

Training Large Language Models (LLMs) to reason often relies on Reinforcement Learning (RL) with task-specific verifiers. However, many real-world reasoning-intensive tasks lack verifiers, despite offering abundant expert demonstrations that remain under-utilized for reasoning-focused training. We introduce **RARO** (Relativistic Adversarial Reasoning Optimization) that learns strong reasoning capabilities from only expert demonstrations via **Inverse Reinforcement Learning**. Our method sets up an adversarial interaction between a **policy** (generator) and a **relativistic critic** (discriminator): the policy learns to mimic expert answers, while the critic learns to compare and distinguish between policy and expert answers. Our method trains both the policy and the critic jointly and continuously via RL, and we identify the key stabilization techniques required for robust learning. Empirically, RARO significantly outperforms strong verifier-free baselines on all of our evaluation tasks — Countdown, DeepMath, and Poetry Writing — and enjoys the same robust scaling trends as RL on verifiable tasks. These results demonstrate that our method effectively elicits strong reasoning performance from expert demonstrations alone, enabling robust reasoning learning even when task-specific verifiers are unavailable.

1 INTRODUCTION

Recent advances in Large Language Models (LLMs) have been driven substantially by improvements in their *reasoning* abilities. Reasoning enables LLMs to perform deliberate intermediate computations before producing answers to the user queries, proposing candidate solutions and self-corrections. Much of this progress has been enabled via Reinforcement Learning (RL) on *verifiable* tasks such as mathematics and competitive programming (DeepSeek-AI et al., 2025; Yang et al., 2025a; Shao et al., 2024; Luo et al., 2025). Notably, recent work has demonstrated that RL with Verifiable Rewards (RLVR) can enable LLMs to develop robust reasoning capabilities without any additional supervision (DeepSeek-AI et al., 2025). A growing body of work further improves the efficiency and stability of such RL algorithms on verifiable tasks, such as DAPO (Yu et al., 2025) and GSPO (Zheng et al., 2025). However, comparatively little attention has been paid to developing reasoning abilities on *non-verifiable* tasks, where task-specific verifiers are unavailable.

Yet, in many impactful and challenging tasks — such as analytical writing, open-ended research, or financial analysis — LLM outputs are not directly verifiable due to hard-to-specify criteria, wide variation among acceptable answers, and other practical constraints. A popular approach in these settings is Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022; Rafailov et al., 2023), but they require collecting human preferences beyond demonstration data, which is often a time-consuming and expensive process.

Without preference data, the typical approach to improving LLM performance in these domains is to conduct Supervised Fine-Tuning (SFT) on expert demonstration data via the next-token prediction objective. However, such methods, even if the data are further annotated with reasoning traces, does not encourage the same reasoning behaviors elicited from large-scale RL training on verifiable tasks (Chu et al., 2025). Additionally, naive next-token prediction objective induces training-inference distribution mismatch: during training, the model conditions only on the dataset contexts, whereas

at inference, it conditions on self-sampled contexts. Training on self-sampled contexts, as occurs during RL, yields lower training-inference mismatch, leading to better performance at test time (Ross et al., 2011). Thus, we hypothesize that leveraging expert demonstrations in conjunction with RL could cultivate robust reasoning abilities, leading to substantially improved performance on downstream tasks and offering a new pathway for developing reasoning capabilities in non-verifiable domains.

To this end, we introduce **RARO** (Relativistic Adversarial Reasoning Optimization), a robust RL algorithm that trains LLMs to reason using only expert demonstrations *without* task-specific verifiers or human preferences.

The key contributions of our work are as follows:

- We propose a novel perspective on training reasoning models via **Inverse Reinforcement Learning** (Ng & Russell, 2000). With this perspective, we develop a principled method, RARO, that enables training reasoning models using demonstration data only.
- We evaluate RARO on a controlled toy reasoning task, **Countdown**, where it not only significantly outperforms SOTA baselines without verification (Zhou et al., 2025), but it nearly matches the performance of RLVR, demonstrating the effectiveness of RARO on inducing reasoning behaviors.
- Next, we further stress test RARO’s reasoning elicitation capability by scaling it on the general domain of math problems via the **DeepMath** dataset (He et al., 2025), where RARO again outperforms baselines without verification and exhibits similar scaling trends as RLVR, demonstrating the *scalability* of RARO.
- Finally, we demonstrate that RARO’s superior performance generalizes well to non-verifiable domains by evaluating it on **Poetry Writing**, where it substantially outperforms all baselines, underscoring its effectiveness in open-ended tasks without verification.

2 PRELIMINARY

2.1 LLM REASONING

Reasoning in LLMs has been a central focus of recent work, with numerous approaches proposed to enhance it, including Chain-of-Thought (CoT) prompting (Wei et al., 2022), Tree of Thoughts (ToT) (Yao et al., 2023), and Buffer of Thoughts (BoT) (Yang et al., 2024).

CoT prompting is a simple yet effective technique: it enables LLMs to generate intermediate reasoning tokens that steer them toward correct answers without additional training. CoT also pairs naturally with Test-Time Scaling, critical for further performance gains (Snell et al., 2024). Recently, *reasoning LLMs* operationalize this idea by explicitly training via RL with verifiable rewards to produce long CoT reasonings before outputting the final response, yielding substantially higher-quality answers (DeepSeek-AI et al., 2025; Yang et al., 2025a).

While CoT reasonings provide useful guidance, the ultimate objective is to improve the quality of the final answers. Thus, following prior work (Phan et al., 2023), we take the perspective that an LLM can be modeled as a joint distribution over prompts, CoT reasonings, and answers. This perspective induces a *conditional latent-variable model* where the prompt is the conditioning variable, the answer is observed, while the CoT reasonings are latent variables.

2.2 REINFORCEMENT LEARNING WITH VERIFIABLE REWARDS

Reinforcement Learning with Verifiable Rewards (RLVR) is a family of methods designed to train LLMs to reason on verifiable tasks, such as mathematics and competitive programming, enabling recent SOTA open-source models to achieve expert-level performance on relevant benchmarks (DeepSeek-AI et al., 2025; Yang et al., 2025b).

The dominant method in this line of work is Group Relative Policy Optimization (GRPO) (Shao et al., 2024), which builds upon the Proximal Policy Optimization (PPO) (Schulman et al., 2017) algorithm by replacing the advantage function with a sample average computed from rollout groups.

Since the introduction of GRPO, numerous works have been proposed to further improve its training efficiency and stability, such as DAPO (Yu et al., 2025) and GSPO (Zheng et al., 2025).

2.3 GENERAL REASONING LEARNING

While RLVR is effective for training LLMs to reason on readily verifiable tasks, it does not directly extend to the broader setting of learning reasoning on real-world domains with no verifiers, yet many of these tasks could still benefit from explicit reasoning (Zhou et al., 2025).

Although no consensus method exists for general reasoning learning to our knowledge, several recent efforts make early progress. Zhou et al. (2025) propose to train LLMs to reason with reward derived from the model’s own logits on expert answers rather than from an external verifier. Jia et al. (2025) propose a pairwise generative reward model with a PPO-style objective for non-verifiable writing tasks, achieving sizable gains without external training signals. Ma et al. (2025) distill a model-based verifier from a strong teacher and leverage it as a reward model to train a general reasoner without verifiers. Li et al. (2025) investigate large-scale multi-task RLVR, hypothesizing that breadth across many tasks induces stronger general reasoning. We build on this line of work while adopting a complementary perspective based on Inverse Reinforcement Learning.

2.4 INVERSE REINFORCEMENT LEARNING

Inverse Reinforcement Learning (IRL) (Ng & Russell, 2000) studies the task of recovering a reward function for which an observed expert policy is near-optimal. A seminal application is robust imitation learning, most notably Generative Adversarial Imitation Learning (GAIL) (Ho & Ermon, 2016), casting imitation as an adversarial game between a policy and a discriminator.

Recently, IRL has been applied to LLMs. Sun & van der Schaar (2025) propose to align LLMs to human demonstrations via IRL without preference labels or explicit feedback, showing that IRL is effective for aligning LLMs to human preferences without preference data. However, their applications remain limited in scope, and with the rise of RL for reasoning, it remains open whether IRL-based methods can effectively train reasoning-focused LLMs.

3 METHOD

We study the general setting where we are given an expert Question–Answer (QA) dataset, and we aim to train a LLM policy to produce expert-level answers via explicit reasoning. We adopt this setting because verifiable tasks are relatively scarce, whereas expert demonstration data are abundant for many non-verifiable domains (e.g., highly upvoted Stack Exchange answers).

To approach this task, we propose a novel inverse reinforcement learning framework that sets up an adversarial interaction between a policy and a relativistic critic: the policy learns to mimic expert answers, while the critic learns to discriminate between policy and expert answers via pairwise comparison. By jointly training both the policy and the critic to reason via RL, we enable the emergence of strong reasoning capabilities from demonstrations alone, without requiring task-specific verifiers.

3.1 FROM MAXIMUM LIKELIHOOD TO REWARD GRADIENT

Setup. Let $D = \{(q_i, a_i)\}_{i=1}^n$ denote the expert QA dataset. We parameterize our LLM policy as $\pi_\theta(a, z \mid q)$, a distribution over answers a and Chain-of-Thought (CoT) reasonings z conditioned on a question q . We let $\hat{p}_q(\cdot)$ denote the empirical distribution of questions in D , $\hat{p}_{a|q}(\cdot \mid \cdot)$ denote the empirical distribution of expert answers conditioned on a question, and the joint $\hat{p}_D = \hat{p}_{a|q}\hat{p}_q$ denote the empirical distribution of dataset pairs (q, a) .

A natural baseline for producing expert-quality answers is the *maximum likelihood (ML)* objective on expert demonstrations: $\arg \max_\theta \mathbb{E}_{(q,a) \sim \hat{p}_D} [\log \pi_\theta(a \mid q)]$.

However, for models that perform CoT reasoning before producing an answer, each (q, a) is associated with many possible CoT traces. Thus, the marginal likelihood required by the ML objective, $\pi_\theta(a \mid q) = \sum_z \pi_\theta(a, z \mid q)$, involves summing over a combinatorially large (often effectively unbounded) set of traces, rendering exact computation and its gradients computationally impractical.

Inverse Reinforcement Learning. To address this intractability, we adopt the perspective of Inverse Reinforcement Learning (IRL). Rather than maximizing the marginal likelihood directly, we learn a *parameterized reward* $r_\phi(a, q)$ over QA pairs such that optimizing a policy $\pi_\theta(a | q)$ with respect to r_ϕ yields a “near-optimal” policy that approximately maximizes the ML objective.

We formalize “near-optimality” via the KL-regularized reward-maximization objective, and under this objective, it can be shown (Peng et al., 2019) that the optimal policy has the following *closed-form solution*:

$$\pi_{\theta^*(\phi)}(a | q) = \frac{1}{Z_{\theta^*(\phi)}(q)} \pi_{\text{ref}}(a | q) \exp \left\{ \frac{1}{\beta} r_\phi(a, q) \right\},$$

where $Z_{\theta^*(\phi)}(q)$ is the partition function, π_{ref} is a fixed reference policy, and $\beta > 0$ controls the strength of the KL-regularization. See Appendix A.1 for the proof.

Reward Gradient. With the closed-form expression for the optimal policy under the reward model r_ϕ , we can derive the corresponding gradient needed to optimize it by differentiating the negative ML loss with respect to ϕ :

$$\nabla_\phi \mathcal{L}(\phi) = \frac{1}{\beta} \left(\underbrace{\mathbb{E}_{(q,a) \sim \hat{p}_D} [\nabla_\phi r_\phi(a, q)]}_{\text{expert answers}} - \underbrace{\mathbb{E}_{q \sim \hat{p}_q} \mathbb{E}_{a' \sim \pi_{\theta^*(\phi)}(\cdot | q)} [\nabla_\phi r_\phi(a', q)]}_{\text{policy answers}} \right).$$

See Appendix A.2 for the proof.

Intuitively, the gradient shapes the reward signal by increasing $r_\phi(a, q)$ on expert answers and decreasing it on policy answers, thus nudging the policy toward the expert distribution. A concrete algorithm for the alternating optimization is given in Algorithm 4 in Appendix E.

3.2 REASONING REWARD MODEL

While Algorithm 4 provides a concrete method for optimization, we have yet to decide on an appropriate architecture for the reward model $r_\phi(\cdot)$. Our setting targets difficult QA tasks that benefit from reasoning. Consequently, to reliably separate expert from policy answers, we expect that the reward model should be at least as capable as the policy. Thus, a natural instantiation is therefore a *reasoning LLM*. Specifically, we reparametrize the reward model with a *binary classification* setup:

$$r_\phi(a, q) = c_\phi(\ell = \text{expert} | a, q) - c_\phi(\ell = \text{policy} | a, q)$$

where $c_\phi(\cdot)$ is a reasoning *critic* that classifies whether an answer is from the expert or the policy.

Under this parameterization, as shown in Appendix A.3, the gradient $\nabla_\phi \mathcal{L}$ corresponds to the standard *policy gradient*, and we can further derive an unbiased estimator for $r_\phi(a, q)$, resulting in two simple reward functions for the critic and policy.

Reward for Critic:

$$R_{\text{critic}}(\ell, a, q) = \mathbb{1}_{\ell \text{ is correct}} - \mathbb{1}_{\ell \text{ is incorrect}}.$$

Reward for Policy:

$$R_{\text{policy}}(a, q) = \mathbb{1}_{\ell=\text{expert}} - \mathbb{1}_{\ell=\text{policy}}, \quad \ell \sim c_\phi(\cdot | a, q).$$

This allows us to optimize both the critic and policy using any reward-maximization algorithm (e.g., GRPO). Intuitively, such reward formulation creates an *adversarial game* between the critic and policy: the critic is rewarded when it correctly classifies answer as coming from the expert or policy, while the policy is rewarded when the critic *incorrectly* classifies its answer as an expert answer.

Limitations. Despite the theoretical soundness, this binary classification setup poses challenges for critic learning. As policy approaches the expert distribution, the classification task becomes much more difficult due to a lack of reference answer for the critic to compare against. In addition, with an optimal policy, the critic effectively degenerates to random guessing, providing high-variance, uninformative gradients to the policy, leading to training instability as observed in our initial experiments (see Appendix D.2).

3.3 RELATIVISTIC CRITIC

To address the lack of reference in the binary classification setup, we adopt a *relativistic* formulation: the critic takes a triplet (q, a, a^*) consisting of one policy answer and one expert answer, and outputs which is better or `tie` if they are equal in quality. This resolves the degeneracy where the critic is forced to differentiate even when the policy is optimal. We empirically show that the `tie` option is crucial for better performance (see Appendix D.2).

Formally, the *relativistic critic* c_ϕ takes a question q and two candidate answers $(a^{(1)}, a^{(2)})$ and returns a label $\ell \in \{1, 2, \text{tie}\}$. Assuming one expert and one policy answer, we can define:

Reward for Critic:

$$R_{\text{critic}}(q, a^{(1)}, a^{(2)}) = \mathbb{1}_{\ell \text{ is expert}} + \tau_{\text{crit}} \cdot \mathbb{1}_{\ell=\text{tie}}, \quad \tau_{\text{crit}} \in [0, 1].$$

Reward for Policy:

$$R_{\text{policy}}(q, a^{(1)}, a^{(2)}) = \mathbb{1}_{\ell \text{ is policy}} + \tau_{\text{pol}} \cdot \mathbb{1}_{\ell=\text{tie}}, \quad \tau_{\text{pol}} \in [0, 1].$$

where τ_{crit} and τ_{pol} are *tie rewards*, new hyperparameters introduced to handle the `tie` label.

Intuitively, unlike the binary classification setup, the relativistic critic is now given a *pairwise comparison* task: the critic is rewarded when it correctly identifies the expert answer, and the policy is rewarded when the critic mistakenly identifies its answer as the expert answer, with additional `tie` rewards to ensure non-degeneracy and stable learning. Algorithm 1 describes the full training process and see Appendix E for example critic outputs.

Algorithm 1 Relativistic Critic with KL-Regularized Policy

Inputs: Dataset $D = \{(q_i, a_i)\}$; Tie reward $\tau_{\text{pol}}, \tau_{\text{crit}}$; Batch B ; Rollout K_π, K_c ;

Models: Policy $\pi_\theta(z, a \mid q)$; Relativistic critic $c_\phi(q, a^{(1)}, a^{(2)})$.

```

1: Initialize  $\theta, \phi$ 
2: for  $t = 1, \dots, T$  do
3:   Draw  $\{(q_i, a_i^E)\}_{i=1}^B \sim D$ 
4:   for  $i = 1..B, k = 1..K_\pi$  do
       Sample  $(z_{i,k}^\pi, a_{i,k}^P) \sim \pi_\theta(z, a \mid q_i)$ ; build pair  $(a_i^E, a_{i,k}^P)$ 
5:    $\ell_{i,k} \leftarrow c_\phi(q_i, a_i^E, a_{i,k}^P)$ 
        $R_{i,k}^{\text{pol}} \leftarrow \mathbb{1}_{[\ell \text{ is policy}]} + \tau_{\text{pol}} \mathbb{1}_{[\ell=\text{tie}]}$ 
6:   end for
7:   for  $i = 1..B, j = 1..K_c$  do
       Form pair  $(a_{i,j}^{(1)}, a_{i,j}^{(2)})$  for  $q_i$ , query critic  $\ell_{i,j} \leftarrow c_\phi(q_i, a_{i,j}^{(1)}, a_{i,j}^{(2)})$ 
8:    $R_{i,j}^{\text{crit}} \leftarrow \mathbb{1}_{[\ell \text{ is expert}]} + \tau_{\text{crit}} \mathbb{1}_{[\ell=\text{tie}]}$ 
9:   end for
10:  GRPO step on  $\theta$  to maximize  $\mathbb{E}[R_{i,k}^{\text{pol}}] - \beta D_{\text{KL}}(\pi_\theta \parallel \pi_{\text{ref}})$ .
11:  GRPO step on  $\phi$  to maximize  $\mathbb{E}[R_{i,j}^{\text{crit}}]$ .
12: end for
13: return  $\theta, \phi$ 

```

3.4 RARO: RELATIVISTIC ADVERSARIAL REASONING OPTIMIZATION

To ensure stable and efficient learning, we implement several optimizations. First, we use a *shared LLM* for both the critic and the policy, which reduces memory usage and promotes generalization. This allows us to employ *data mixing*, where policy and critic rollouts are combined in a single batch, simplifying the training loop. To prevent the critic from suffering from catastrophic forgetting, we utilize a *replay buffer* that mixes past policy rollouts with current ones. Finally, we incorporate

Algorithm 2 RARO (Relativistic Adversarial Reasoning Optimization)

Inputs: Dataset $D = \{(q_i, a_i)\}$; Tie reward $\tau_{\text{pol}}, \tau_{\text{crit}}$; Loss weight $\lambda_{\text{pol}}, \lambda_{\text{crit}}$ Batch B ; Rollout K .
Model: Shared $\theta \rightarrow \pi_\theta, c_\theta$. Replay buffer \mathcal{R} .

```
1: Initialize  $\theta, \mathcal{R} \leftarrow \emptyset$ ;  
2: for  $t = 1, \dots, T$  do  
3:    $\mathcal{R}_{\text{new}} \leftarrow \emptyset$   
4:   Draw  $\{(q_i, a_i^E)\}_{i=1}^B \sim D$   
5:   for  $i = 1 \dots B, k = 1 \dots K$  do  
6:      $(z_{i,k}^P, a_{i,k}^P) \sim \pi_\theta(\cdot | q_i)$ ; build pair  $(a_i^E, a_{i,k}^P)$   
      $\ell_{i,k} \sim c_\theta(\cdot | q_i, a_i^E, a_{i,k}^P)$   
      $R_{i,k}^{\text{pol}} \leftarrow \mathbb{I}[\ell_{i,k} \text{ is policy}] + \tau_{\text{pol}} \mathbb{I}[\ell_{i,k} = \text{tie}]$   
      $\mathcal{R}_{\text{new}} \leftarrow \mathcal{R}_{\text{new}} \cup \{(q_i, a_i^E, a_{i,k}^P)\}$   
7:   end for  
8:    $\mathcal{C} \leftarrow \text{Mix}(\mathcal{R}_{\text{new}}, \mathcal{R})$   
9:    $\mathcal{R} \leftarrow \mathcal{R} \cup \mathcal{R}_{\text{new}}$   
10:  for  $(q_j, a_j^E, a_j^P) \in \mathcal{C}$  do  
11:     $\ell_j \sim c_\theta(\cdot | q_j, a_j^E, a_j^P)$   
     $R_j^{\text{crit}} \leftarrow \mathbb{I}[\ell_j \text{ is expert}] + \tau_{\text{crit}} \mathbb{I}[\ell_j = \text{tie}]$   
12:  end for  
13:  GRPO step on  $\theta$  to maximize:  $\lambda_{\text{pol}} J_{\text{pol}}(\theta) + \lambda_{\text{crit}} J_{\text{crit}}(\theta) - \beta D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}})$   
14: end for  
15: return  $\theta$ 
```

} **Policy rollouts**

} **Critic rollouts**

several practical improvements to the GRPO algorithm, such as over-length filtering and removing advantage/length normalization. For full implementation details, please refer to Appendix C.1.

Incorporating all of these optimizations into a concrete algorithm, we arrive at our final algorithm, *RARO (Relativistic Adversarial Reasoning Optimization)*, shown in Algorithm 2.

4 EXPERIMENTS

4.1 TASKS & DATASETS

We evaluate RARO on three diverse reasoning tasks that probe complementary aspects of reasoning. See Appendix C.2 for more details on the datasets.

Countdown. First, we evaluate our method on the Countdown task, a controlled toy reasoning task where answer verification is much simpler than answer generation. We use a 24-style variant where the goal is to combine four integers to obtain 24 (see Appendix C.2 for details). Through this task, we aim to study the effectiveness of our method on reasoning capabilities in a controlled environment where answer checking is much easier than solution search.

DeepMath. Then, we evaluate our method on the domain of general math reasoning problems using the DeepMath dataset (He et al., 2025). Compared to Countdown, answer verification in the general math domain is significantly more challenging, often requiring reproduction of the derivation. Through this task, we aim to stress test our method on difficult general reasoning environments where verification is as difficult as generation.

Poetry Writing. Finally, we extend our method to its intended setting of non-verifiable, open-ended reasoning tasks using a custom Poetry Writing dataset. Unlike the math tasks, poetry writing does not admit an objective verifier. Thus, for evaluation, we use GPT-5 (OpenAI, 2025) as a judge to evaluate poems in both isolation and in comparison to the expert poem (see Appendix C.2

for details). This task represents the non-verifiable regime that our method aims to handle, where explicit reasoning could significantly improve quality.

4.2 BASELINES

We compare RARO against several strong post-training baselines under the same dataset, training, and evaluation setup.

Supervised Fine-Tuning (SFT). The SFT baseline trains the base models to directly maximize the conditional log-likelihood of the expert answer given the question, representing the standard use of demonstration data.

Rationalization. Following prior work on self-rationalizing techniques (Zelikman et al., 2022), we construct a rationalization baseline that augments each expert answer with an explicit CoT. Concretely, we prompt the base model to annotate the expert demonstrations with free-form *rationale*, then perform SFT on the concatenated (question, rationale, answer) sequences. This baseline attempts to incentivize the base model to learn to reason before producing the final answer.

Iterative Direct Preference Optimization (DPO). A natural way to match the policy’s output distribution to the expert is to apply Iterative DPO (Rafailov et al., 2024). Inspired by Iterative Reasoning Preference Optimization (Pang et al., 2024), we perform 3 rounds of DPO iteratively: in each round, we sample one response per question to form preference pairs favoring the expert. We initialize from the SFT checkpoint to mitigate distribution mismatch and report the best performance across rounds.

RL from logit-based reward (RL-Logit). Recent work has proposed training reasoning LLMs via RL where the reward is derived from the model’s own logits on expert answers rather than from an external verifier (Zhou et al., 2025; Gurung & Lapata, 2025). We implement two variants of such *logit-based* rewards (see Appendix C.3 for details):

- a *log-probability reward*, which uses the log-probability of the expert answer a^* given the question q and generated reasoning tokens z as the scalar reward $\log \pi_\theta(a^* | q, z)$; and
- a *perplexity reward*, which instead maximizes the negative perplexity of the expert answer under the same conditional distribution.

In our evaluation, we report the metrics from the best performing variant.

RL with Verifiable Reward (RLVR). For Countdown and DeepMath, where ground-truth verifiers are available, we additionally include a RLVR baseline trained with GRPO on binary rewards given by the verifier. This corresponds to the standard RLVR setting, and serves as an upper-bound for our method on tasks where verification is accessible.

4.3 TRAINING & EVALUATION SETUP

We evaluate our method and baselines on the Qwen2.5 (Qwen et al., 2025) family of models, and to focus on improving reasoning performance rather than language understanding, we initialize from the instruction-tuned checkpoints instead of the pretrained model checkpoints. We select the Qwen2.5 family they are popular *non-reasoning* LLMs, allowing us to study the effectiveness of our method on eliciting reasoning behaviors in a controlled manner.

Countdown and DeepMath are evaluated with a ground-truth verifier, while Poetry Writing is evaluated with GPT-5 as a judge in two fashions: a *scalar score* normalized to 0-100 and a *win-rate* against the expert poem. See Appendix C.2 for further details.

Each dataset is split into train, validation, and test sets, and we select our checkpoints based on the highest validation performance. For each dataset and model size, we match dataset splits, rollout budgets, hyperparameters, and sampling configurations when possible to ensure a fair comparison. Unless otherwise specified, all methods are trained and evaluated with a reasoning budget of 2048 tokens. Full implementation details and hyperparameters are provided in Appendix C.

4.4 MAIN RESULTS

We present our experimental results structured by task: Countdown, DeepMath, and Poetry Writing. Across these domains, we observe that our method significantly and consistently outperforms all baselines, scaling effectively with both reasoning budget and model size.

4.4.1 COUNTDOWN

We first evaluate RARO on the Countdown task, a controlled toy reasoning task where answer verification is much simpler than answer generation. For this task, we focus our investigation at the 1.5B model size and further ablate our method and baselines with respect to both the training and test-time reasoning token budget. We do not ablate along model size as Countdown is a straightforward task where the reasoning budget is the primary bottleneck rather than model capacity (see Appendix D.1 for additional details).

Superior Performance at Fixed Budget. At a fixed reasoning budget of 2048 tokens, RARO achieves 54.4% accuracy, significantly outperforming the best verifier-free baseline (SFT, 40.7%) by 13.7% and nearly matching the oracle RLVR baseline (57.7%) (Table 1). We also notice that RL-Logit (2.2%) and Rationalization (12.5%) perform rather poorly, and we hypothesize that it is likely due to the base model’s inability to produce high-quality rationalizations or informative logits. The strong performance of RARO demonstrates that our learned critic provides a signal comparable to verification rewards.

Emergence of Self-Correcting Search. A key qualitative finding is the emergence of explicit search behaviors. As shown in Figure 5, our model learns to explore the solution space dynamically proposing combinations, verifying them, and backtracking when they are incorrect (e.g., “too high”). This self-correction mechanism acts as an internal verifier, allowing the model to recover from errors. Such behavior is absent in the SFT baseline, as it is trained to directly output a candidate answer without any explicit reasoning.

Scaling with Reasoning Budget. Finally, we examine the scalability of RARO with respect to both training and test-time reasoning token budget. Figure 1 illustrates a clear trend: while the SFT baseline’s performance plateaus at 40.7% regardless of the token budget, our method exhibits continuous improvement as the budget increases, rising from 33.1% at 256 tokens to 61.3% at 4096 tokens. Notably, the result at 4096 tokens is achieved by a model trained with a 2048-token budget, demonstrating that our method can extrapolate to longer reasoning chains at test time without additional training. This scaling behavior confirms that RARO successfully transforms reasoning budget into better performance, a hallmark of effective reasoning.

4.4.2 DEEPMATH

Next, we evaluate RARO on the DeepMath dataset, a collection of general math problems. For the DeepMath task, we focus on scaling our method and baselines with respect to model size instead of reasoning budget, as it is a much more general setting where model capacity is a real bottleneck in performance.

Method	Countdown accuracy (%) \uparrow
RLVR (<i>with verifier</i>)	57.7 ± 1.6
Base	2.0 ± 0.4
SFT	40.7 ± 1.6
Rationalization	12.5 ± 1.0
Iterative DPO	40.4 ± 1.5
RL-Logit	2.2 ± 0.4
RARO	54.4 ± 1.5

Table 1: **Main Countdown Results.** RARO against baselines at a fixed reasoning budget of 2048 tokens.

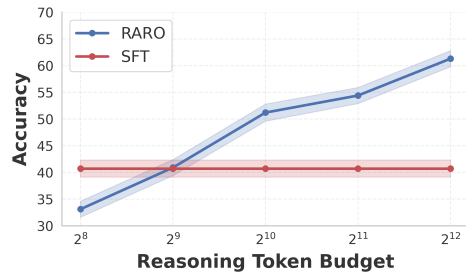


Figure 1: **Reasoning Budget Scaling on Countdown.** Our method scales effectively with both training and test-time token budget, unlike SFT (best baseline). See Table 7 in Appendix E for detailed data.

Method	DeepMath accuracy (%) \uparrow	Poetry score (0-100) \uparrow	Poetry win-rate (%) \uparrow
1.5B			
RLVR (<i>with verifier</i>)	50.9 ± 1.9	N/A	N/A
Base	29.6 ± 1.9	35.0 ± 0.9	0.0 ± 0.0
SFT	35.7 ± 1.8	53.7 ± 1.0	2.3 ± 1.0
Rationalization	34.5 ± 2.0	35.6 ± 1.6	0.8 ± 0.5
Iterative DPO	33.0 ± 1.9	48.6 ± 0.9	0.0 ± 0.0
RL-Logit	37.7 ± 1.9	36.4 ± 0.7	0.0 ± 0.0
RARO	41.3 ± 1.9	67.8 ± 0.8	7.8 ± 1.7
3B			
RLVR (<i>with verifier</i>)	55.8 ± 2.0	N/A	N/A
Base	39.4 ± 1.9	46.5 ± 0.9	0.0 ± 0.0
SFT	39.0 ± 1.9	57.4 ± 1.0	2.3 ± 1.0
Rationalization	32.3 ± 1.9	30.8 ± 1.9	0.4 ± 0.4
Iterative DPO	34.2 ± 1.9	69.8 ± 0.8	6.6 ± 1.5
RL-Logit	43.1 ± 2.0	46.9 ± 0.8	0.4 ± 0.4
RARO	49.1 ± 1.9	71.9 ± 0.8	17.2 ± 2.4
7B			
RLVR (<i>with verifier</i>)	66.2 ± 1.9	N/A	N/A
Base	44.2 ± 2.1	54.0 ± 0.9	1.2 ± 0.7
SFT	42.3 ± 1.9	65.4 ± 1.0	5.9 ± 1.4
Rationalization	48.6 ± 1.9	57.7 ± 1.2	5.1 ± 1.3
Iterative DPO	36.9 ± 2.0	66.5 ± 0.9	5.1 ± 1.4
RL-Logit	49.3 ± 2.0	55.4 ± 0.8	3.9 ± 1.2
RARO	57.5 ± 2.0	77.3 ± 0.8	25.0 ± 2.6

Table 2: **Main results for DeepMath and Poetry.** We report results for RARO against baselines on DeepMath and Poetry Writing across model scales with a reasoning budget of 2048 tokens. For Iterative DPO, we report the max of the 3 rounds. For RL-Logit, we report the best over the 2 variants. See Table 12 in Appendix E for full data.

Significant Improvement over Baselines.

As reported in Table 2, RARO consistently outperforms all verifier-free baselines across model scales. With the 1.5B model, we achieve 41.3% accuracy compared to 37.7% for the best baseline (RL-Logit), an improvement of 3.6%. This advantage grows with model size: at 3B, our method (49.1%) surpasses the best baseline (RL-Logit, 43.1%) by 6.0%, and at 7B, it reaches 57.5%, beating the best baseline (RL-Logit, 49.3%) by 8.2%. These results demonstrate that our adversarial learning framework provides a strong signal for reasoning that outperforms not only purely supervised approaches like SFT or Rationalization but also RL-based approaches like RL-Logit.

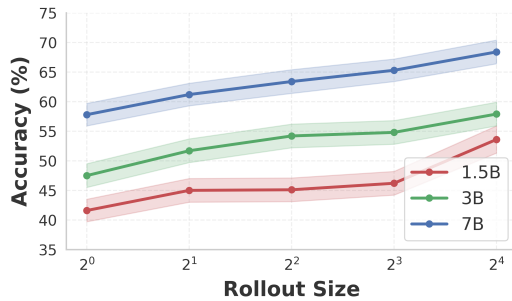


Figure 2: **Test-time Scaling (TTS) on DeepMath.** Performance improves as the number of rollouts (N) increases for all model sizes. See Table 8 in Appendix E for detailed data.

Stable Training Dynamics. We further analyze the training dynamics of RARO. As shown in Figure 4 and Figure 7, our coupled training objective maintains a robust equilibrium, allowing the policy to steadily improve its reasoning capabilities and response length without collapsing. This stability confirms the robustness of our optimization procedure.

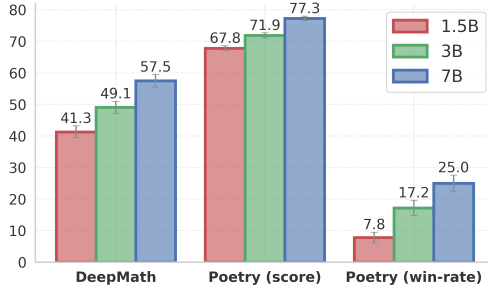


Figure 3: **Performance scaling.** RARO consistently improves with model size (1.5B to 7B) across both DeepMath and Poetry Writing.

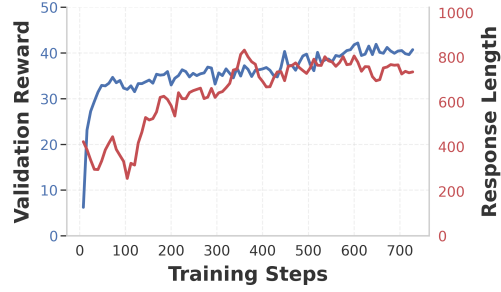


Figure 4: **Stable Reward and Length Growth.** The validation reward and response length of RARO on DeepMath (1.5B) continuously grows over time, indicating a stable dynamic.

Effective Test-Time Scaling. Another key advantage of RARO is that our learned critic enables natural Test-Time Scaling (TTS) to further improve the policy’s performance. Specifically, our critic’s pairwise comparison setup allows for a *single-elimination tournament* with the critic as the judge (see Algorithm 3), enabling further policy improvements with additional rollouts. As shown in Figure 2 (and detailed in Table 8), increasing the number of rollouts from 1 to 16 consistently improves performance. Notably, with 16 rollouts, RARO achieves 53.6% on the 1.5B model and 57.9% on the 3B model. When comparing against the RLVR baseline with the same TTS strategy (Table 9), we observe that RARO achieves a similar rate of improvement. This result highlights that RARO, when combined with test-time search, can scale effectively, matching the scaling trends of models trained with oracle verifiers.

4.4.3 POETRY WRITING

Finally, we study RARO on Poetry Writing, an open-ended, un-verifiable domain that benefits from specialized reasoning capabilities. For this task, similar to DeepMath, we study RARO across a range of model sizes.

Surpassing Supervised Baselines. Table 12 reveals a striking performance gap between RARO and baselines. While SFT and Rationalization achieve modest win-rates against expert poetry (peaking at 5.9% with the 7B model), RARO reaches 25.0%, a four-fold improvement. This advantage is also reflected in the scoring evaluation, where RARO consistently surpasses baselines (e.g., 67.8 vs. 53.7 for SFT at 1.5B). Notably, RL-Logit, leading baseline for DeepMath, fails to produce competitive results, yielding near-zero improvement over the base model (0.364 vs. 0.350 at 1.5B) for both the win-rate and scoring evaluation. This suggests that standard preference optimization methods and RL-based approaches struggle to generalize in open-ended domains.

Scaling Creative Capabilities. A key result is the scalability of RARO with model size in the creative domain. As we increase model capacity from 1.5B to 7B, the win-rate against expert human poems grows substantially, from 7.8% to 25.0%. The scoring evaluation similarly improves from 67.8 to 77.3. This trend shows that just like verifiable domains, RARO continues to effectively scale with model size in open-ended domains.

Emergent Qualitative Reasoning. Qualitatively, RARO induces explicit planning and reasoning behaviors even in open-ended domains. As shown in Figure 5 and fully detailed in Figure 17, the model learns to decompose the prompt into key themes (e.g., “disillusionment”, “transience of power”) and stylistic constraints (e.g., “flowing, rhythmic yet contemplative style”) before generating the poem. This demonstrates that RARO effectively elicits reasonings that align the model’s output to creative poems while adhering to the prompt’s nuanced requirements.

<p>Countdown Input</p> <p>Use the four numbers 5, 33, 40, and 26 to create an expression that equals 24.</p>	<p>DeepMath Input</p> <p>Find a positive integer $n > 1$ that is both a sum of two squares ($n = a^2 + b^2$) and a perfect cube ($n = z^3$), where $a \neq b$. Determine the non-trivial solutions to the equation $a^2 + b^2 = z^3$.</p>	<p>Poetry Input</p> <p>A short free-verse poem about the ironic pageantry of war glory turning to dust at sunset</p>
<p>Countdown Reasoning & Answer</p> <p>To solve this problem, we need to use the numbers 5, 33, 40, and 26 to create an expression that equals 24. Let's explore different combinations and find one that meets the requirements.</p> <p>1. $(40 - 26) * (33 - 5) = 14 * 28 = 392$ (too high) 2. $(40 - 5) * (33 - 26) = 35 * 7 = 245$ (too high) ... 16. $(33 + 5) - (40 - 26) = 38 - 14 = 24$ (just right! This is the correct solution.)</p> <hr/> <p>Answer: $(33 + 5) - (40 - 26)$</p>	<p>DeepMath Reasoning & Answer</p> <p>To find a positive integer $n > 1$ that is both a sum of two squares and a perfect cube, where $a \neq b$, we start by considering the general form of such numbers.</p> <p>For $z = 2$, we need $n = 2^3 = 8$. We check if 8 can be expressed as $a^2 + b^2$ with $a \neq b$...</p> <p>... For $z = 5$, we need $n = 5^3 = 125$. The possible pairs (a, b) are $(3, 4)$ and $(4, 3)$, and since $a \neq b$, this pair works. Therefore, $125 = 3^2 + 4^2$.</p> <hr/> <p>Answer: 125</p>	<p>Poetry Reasoning & Answer</p> <p>To write a free-verse poem about war glory, I should first consider the themes of disillusionment, the transience of power, and what aspects of war can be attributed to pageantry.</p> <p>... These should be woven into a flowing, rhythmic yet contemplative style, reflective of an artist's perspective on war.</p> <hr/> <p>Answer: The Decoration (1942)</p> <p>Nestled, red-glazed in a rail, Is the body of Brigadier Sir Thomas Hassall. ... His very gold is dust.</p>

Figure 5: **Qualitative examples for Countdown, DeepMath, and Poetry Writing at 1.5B.** For each task, we show the input (top) and the the truncated reasoning and answer sampled from 1.5B models trained with RARO (bottom). See Appendix E for full examples.

5 CONCLUSION & FUTURE WORK

We introduced RARO (Relativistic Adversarial Reasoning Optimization), a novel approach to training reasoning LLMs using only expert demonstrations, thereby bypassing the need for task-specific verifiers or expensive preference annotations. By formulating the problem as Inverse Reinforcement Learning and incorporating a relativistic critic setup, we obtain a principled and stable adversarial training algorithm that yields strong reasoning capabilities.

Our experiments demonstrate the effectiveness of RARO: (i) on the controlled Countdown task, it not only outperforms verifier-free baselines but also nearly matches the performance of RLVR; (ii) on the general math domain, it exhibits similar scalability trends to RLVR while outperforming baselines without verification; and (iii) on the open-ended Poetry Writing task, it successfully elicits emerging specialized reasoning capabilities and significantly surpasses all baselines. Together, these findings suggest that RARO is a promising and practical approach for training reasoning models without reliance on explicit verifiers.

Future work includes: (i) extending the framework to more generalized adversarial setups that stabilize training across diverse domains; (ii) improving sample efficiency; (iii) scaling the method to larger, state-of-the-art model sizes; and (iv) developing an alternative critic setup that enables better reward interpretability. See Appendix B for more details.

REFERENCES

- Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training, 2025. URL <https://arxiv.org/abs/2501.17161>.
- DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyi Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding,

-
- Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, and S. S. Li. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948, January 2025. URL <https://doi.org/10.48550/arXiv.2501.12948>.
- Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. URL <https://arxiv.org/abs/1406.2661>.
- Alexander Gurung and Mirella Lapata. Learning to reason for long-form story generation, 2025. URL <https://arxiv.org/abs/2503.22828>.
- Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning, 2025. URL <https://arxiv.org/abs/2504.11456>.
- Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning, 2016. URL <https://arxiv.org/abs/1606.03476>.
- Ruipeng Jia, Yunyi Yang, Yongbo Gai, Kai Luo, Shihao Huang, Jianhe Lin, Xiaoxi Jiang, and Guanjin Jiang. Writing-zero: Bridge the gap between non-verifiable tasks and verifiable rewards, 2025. URL <https://arxiv.org/abs/2506.00103>.
- Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data, 2020. URL <https://arxiv.org/abs/2006.06676>.
- Peiji Li, Jiasheng Ye, Yongkang Chen, Yichuan Ma, Zijie Yu, Kedi Chen, Ganqu Cui, Haozhan Li, Jiacheng Chen, Chengqi Lyu, Wenwei Zhang, Linyang Li, Qipeng Guo, Dahua Lin, Bowen Zhou, and Kai Chen. Internbootcamp technical report: Boosting llm reasoning with verifiable task scaling, 2025. URL <https://arxiv.org/abs/2508.08636>.
- KJ Liang and L Carin. Generative adversarial network training is a continual learning problem. *arXiv preprint arXiv:1811.11083*, 2018.
- Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding rl-zero-like training: A critical perspective. In *Conference on Language Modeling (COLM)*, 2025.
- Michael Luo, Sijun Tan, Roy Huang, Ameen Patel, Alpay Ariyak, Qingyang Wu, Xiaoxiang Shi, Rachel Xin, Colin Cai, Maurice Weber, Ce Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepcoder: A fully open-source 14b coder at o3-mini level. <https://pretty-radio-b75.notion.site/DeepCoder-A-Fully-Open-Source-14B-Coder-at-O3-mini-Level-1cf81902c14680b3bee5eb349>, 2025. Notion Blog.
- Xueguang Ma, Qian Liu, Dongfu Jiang, Ge Zhang, Zejun Ma, and Wenhua Chen. General-reasoner: Advancing llm reasoning across all domains, 2025. URL <https://arxiv.org/abs/2505.14652>.
- Andrew Y. Ng and Stuart J. Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning, ICML '00*, pp. 663–670, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1558607072.
- OpenAI. Gpt-5, 2025. URL <https://openai.com>. Technical report, unreleased.

-
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.
- Richard Yuanzhe Pang, Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston. Iterative reasoning preference optimization, 2024. URL <https://arxiv.org/abs/2404.19733>.
- Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning, 2019. URL <https://arxiv.org/abs/1910.00177>.
- Du Phan, Matthew D. Hoffman, David Dohan, Sholto Douglas, Tuan Anh Le, Aaron Parisi, Pavel Sountsov, Charles Sutton, Sharad Vikram, and Rif A. Saurous. Training chain-of-thought via latent-variable inference, 2023. URL <https://arxiv.org/abs/2312.02179>.
- Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL <https://arxiv.org/abs/2412.15115>.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL <https://openreview.net/forum?id=HPuSIXJaa9>.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2024. URL <https://arxiv.org/abs/2305.18290>.
- Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík (eds.), *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL <https://proceedings.mlr.press/v15/ross11a.html>.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL <https://arxiv.org/abs/1707.06347>.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL <https://arxiv.org/abs/2402.03300>.
- Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.
- Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters, 2024. URL <https://arxiv.org/abs/2408.03314>.

-
- Hao Sun and Mihaela van der Schaar. Inverse-rllignment: Large language model alignment from demonstrations through inverse reinforcement learning, 2025. URL <https://arxiv.org/abs/2405.15624>.
- Hoang Thanh-Tung and T. Tran. Catastrophic forgetting and mode collapse in gans. *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–10, 2020. URL <https://api.semanticscholar.org/CorpusID:221659882>.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiayi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025a. URL <https://arxiv.org/abs/2505.09388>.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiayi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025b. URL <https://arxiv.org/abs/2505.09388>.
- Ling Yang, Zhaochen Yu, Tianjun Zhang, Shiyi Cao, Minkai Xu, Wentao Zhang, Joseph E. Gonzalez, and Bin Cui. Buffer of thoughts: Thought-augmented reasoning with large language models, 2024. URL <https://arxiv.org/abs/2406.04271>.
- Feng Yao, Liyuan Liu, Dinghui Zhang, Chengyu Dong, Jingbo Shang, and Jianfeng Gao. Your efficient rl framework secretly brings you off-policy rl training, August 2025. URL <https://fengyao.notion.site/off-policy-rl>.
- Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822, 2023.
- Qiyang Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. Dapo: An open-source llm reinforcement learning system at scale, 2025. URL <https://arxiv.org/abs/2503.14476>.
- Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. Star: Bootstrapping reasoning with reasoning, 2022. URL <https://arxiv.org/abs/2203.14465>.
- Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization, 2025. URL <https://arxiv.org/abs/2507.18071>.
- Xiangxin Zhou, Zichen Liu, Anya Sims, Haonan Wang, Tianyu Pang, Chongxuan Li, Liang Wang, Min Lin, and Chao Du. Reinforcing general reasoning without verifiers. *arXiv preprint arXiv:2505.21493*, 2025.

A DERIVATIONS

A.1 DERIVATION OF CLOSED-FORM OPTIMAL POLICY

Proposition A.1. *Consider the KL-regularized reward-maximization objective:*

$$\theta^*(\phi) = \arg \max_{\theta} \mathbb{E}_{(q,a) \sim \hat{p}_D} \left[r_{\phi}(a, q) - \beta D_{\text{KL}}(\pi_{\theta}(\cdot | q) \| \pi_{\text{ref}}(\cdot | q)) \right].$$

The optimal policy has the following closed-form solution:

$$\pi_{\theta^*(\phi)}(a | q) = \frac{1}{Z_{\theta^*(\phi)}(q)} \pi_{\text{ref}}(a | q) \exp \left\{ \frac{1}{\beta} r_{\phi}(a, q) \right\},$$

where $Z_{\theta^*(\phi)}(q)$ is the partition function ensuring normalization.

Proof. We derive the closed-form solution for the KL-regularized reward maximization objective. Consider the objective function for a single question q :

$$\mathcal{J}(\pi) = \mathbb{E}_{a \sim \pi(\cdot | q)} [r_{\phi}(a, q)] - \beta D_{\text{KL}}(\pi(\cdot | q) \| \pi_{\text{ref}}(\cdot | q)). \quad (1)$$

Expanding the KL divergence term:

$$\begin{aligned} D_{\text{KL}}(\pi \| \pi_{\text{ref}}) &= \mathbb{E}_{a \sim \pi(\cdot | q)} \left[\log \frac{\pi(a | q)}{\pi_{\text{ref}}(a | q)} \right] \\ &= \mathbb{E}_{a \sim \pi(\cdot | q)} [\log \pi(a | q) - \log \pi_{\text{ref}}(a | q)]. \end{aligned}$$

Substituting this back into the objective:

$$\begin{aligned} \mathcal{J}(\pi) &= \mathbb{E}_{a \sim \pi(\cdot | q)} [r_{\phi}(a, q) - \beta \log \pi(a | q) + \beta \log \pi_{\text{ref}}(a | q)] \\ &= \beta \mathbb{E}_{a \sim \pi(\cdot | q)} \left[\frac{1}{\beta} r_{\phi}(a, q) + \log \pi_{\text{ref}}(a | q) - \log \pi(a | q) \right] \\ &= -\beta \mathbb{E}_{a \sim \pi(\cdot | q)} \left[\log \pi(a | q) - \left(\log \pi_{\text{ref}}(a | q) + \frac{1}{\beta} r_{\phi}(a, q) \right) \right]. \end{aligned}$$

Let us define the normalized Gibbs distribution:

$$\pi^*(a | q) = \frac{1}{Z(q)} \pi_{\text{ref}}(a | q) \exp \left(\frac{1}{\beta} r_{\phi}(a, q) \right), \quad (2)$$

where $Z(q) = \int \pi_{\text{ref}}(a' | q) \exp \left(\frac{1}{\beta} r_{\phi}(a', q) \right) da'$ is the partition function. Taking the logarithm of π^* :

$$\log \pi^*(a | q) = \log \pi_{\text{ref}}(a | q) + \frac{1}{\beta} r_{\phi}(a, q) - \log Z(q). \quad (3)$$

Substituting $\log \pi_{\text{ref}}(a | q) + \frac{1}{\beta} r_{\phi}(a, q) = \log \pi^*(a | q) + \log Z(q)$ into the objective:

$$\begin{aligned} \mathcal{J}(\pi) &= -\beta \mathbb{E}_{a \sim \pi(\cdot | q)} [\log \pi(a | q) - (\log \pi^*(a | q) + \log Z(q))] \\ &= -\beta \left(\mathbb{E}_{a \sim \pi(\cdot | q)} \left[\log \frac{\pi(a | q)}{\pi^*(a | q)} \right] - \log Z(q) \right) \\ &= -\beta D_{\text{KL}}(\pi \| \pi^*) + \beta \log Z(q). \end{aligned}$$

Since $\beta > 0$ and $\log Z(q)$ does not depend on π , maximizing $\mathcal{J}(\pi)$ is equivalent to minimizing the KL divergence $D_{\text{KL}}(\pi \| \pi^*)$. By Gibbs' inequality, $D_{\text{KL}}(\pi \| \pi^*) \geq 0$, with equality if and only if $\pi = \pi^*$ almost everywhere. Thus, the optimal policy is given by:

$$\pi_{\theta^*(\phi)}(a | q) = \pi^*(a | q) = \frac{1}{Z(q)} \pi_{\text{ref}}(a | q) \exp \left(\frac{1}{\beta} r_{\phi}(a, q) \right). \quad (4)$$

□

A.2 PROOF OF REWARD GRADIENT

Proposition A.2. *Using the closed-form policy, the gradient of the data log-likelihood $\mathcal{L}(\phi) = \mathbb{E}_{(q,a) \sim \hat{p}_D} [\log \pi_{\theta^*(\phi)}(a | q)]$ with respect to ϕ is:*

$$\nabla_{\phi} \mathcal{L}(\phi) = \frac{1}{\beta} \left(\mathbb{E}_{(q,a) \sim \hat{p}_D} [\nabla_{\phi} r_{\phi}(a, q)] - \mathbb{E}_{q \sim \hat{p}_q} \mathbb{E}_{a' \sim \pi_{\theta^*(\phi)}(\cdot | q)} [\nabla_{\phi} r_{\phi}(a', q)] \right).$$

Proof. We aim to derive the gradient of the data log-likelihood objective with respect to the reward parameters ϕ . Recall the objective:

$$\mathcal{L}(\phi) = \mathbb{E}_{(q,a) \sim \hat{p}_D} [\log \pi_{\theta^*(\phi)}(a | q)]. \quad (5)$$

The optimal policy $\pi_{\theta^*(\phi)}$ takes the closed-form solution:

$$\pi_{\theta^*(\phi)}(a | q) = \frac{1}{Z_{\theta^*(\phi)}(q)} \pi_{\text{ref}}(a | q) \exp \left(\frac{1}{\beta} r_{\phi}(a, q) \right), \quad (6)$$

where $Z_{\theta^*(\phi)}(q) = \int \pi_{\text{ref}}(a' | q) \exp \left(\frac{1}{\beta} r_{\phi}(a', q) \right) da'$ is the partition function.

Substituting the policy expression into the log-likelihood:

$$\begin{aligned} \log \pi_{\theta^*(\phi)}(a | q) &= \log \left(\frac{\pi_{\text{ref}}(a | q) \exp \left(\frac{1}{\beta} r_{\phi}(a, q) \right)}{Z_{\theta^*(\phi)}(q)} \right) \\ &= \log \pi_{\text{ref}}(a | q) + \frac{1}{\beta} r_{\phi}(a, q) - \log Z_{\theta^*(\phi)}(q). \end{aligned}$$

Since π_{ref} does not depend on ϕ , the gradient is:

$$\begin{aligned} \nabla_{\phi} \log \pi_{\theta^*(\phi)}(a | q) &= \nabla_{\phi} \left(\frac{1}{\beta} r_{\phi}(a, q) - \log Z_{\theta^*(\phi)}(q) \right) \\ &= \frac{1}{\beta} \nabla_{\phi} r_{\phi}(a, q) - \frac{\nabla_{\phi} Z_{\theta^*(\phi)}(q)}{Z_{\theta^*(\phi)}(q)}. \end{aligned}$$

We now compute the gradient of the partition function $Z_{\theta^*(\phi)}(q)$ using the Leibniz integral rule (interchanging gradient and integral):

$$\begin{aligned} \nabla_{\phi} Z_{\theta^*(\phi)}(q) &= \nabla_{\phi} \int \pi_{\text{ref}}(a' | q) \exp \left(\frac{1}{\beta} r_{\phi}(a', q) \right) da' \\ &= \int \pi_{\text{ref}}(a' | q) \nabla_{\phi} \exp \left(\frac{1}{\beta} r_{\phi}(a', q) \right) da' \\ &= \int \pi_{\text{ref}}(a' | q) \exp \left(\frac{1}{\beta} r_{\phi}(a', q) \right) \left(\frac{1}{\beta} \nabla_{\phi} r_{\phi}(a', q) \right) da'. \end{aligned}$$

Substituting this back into the gradient term for $\log Z_{\theta^*(\phi)}(q)$:

$$\begin{aligned} \frac{\nabla_{\phi} Z_{\theta^*(\phi)}(q)}{Z_{\theta^*(\phi)}(q)} &= \int \frac{\pi_{\text{ref}}(a' | q) \exp \left(\frac{1}{\beta} r_{\phi}(a', q) \right)}{Z_{\theta^*(\phi)}(q)} \left(\frac{1}{\beta} \nabla_{\phi} r_{\phi}(a', q) \right) da' \\ &= \int \pi_{\theta^*(\phi)}(a' | q) \left(\frac{1}{\beta} \nabla_{\phi} r_{\phi}(a', q) \right) da' \\ &= \mathbb{E}_{a' \sim \pi_{\theta^*(\phi)}(\cdot | q)} \left[\frac{1}{\beta} \nabla_{\phi} r_{\phi}(a', q) \right]. \end{aligned}$$

Finally, averaging over the dataset $(q, a) \sim \hat{p}_D$:

$$\begin{aligned} \nabla_{\phi} \mathcal{L}(\phi) &= \mathbb{E}_{(q,a) \sim \hat{p}_D} \left[\frac{1}{\beta} \nabla_{\phi} r_{\phi}(a, q) - \mathbb{E}_{a' \sim \pi_{\theta^*(\phi)}(\cdot | q)} \left[\frac{1}{\beta} \nabla_{\phi} r_{\phi}(a', q) \right] \right] \\ &= \frac{1}{\beta} \left(\mathbb{E}_{(q,a) \sim \hat{p}_D} [\nabla_{\phi} r_{\phi}(a, q)] - \mathbb{E}_{q \sim \hat{p}_q} \mathbb{E}_{a' \sim \pi_{\theta^*(\phi)}(\cdot | q)} [\nabla_{\phi} r_{\phi}(a', q)] \right). \end{aligned}$$

This completes the derivation. \square

A.3 DERIVATION OF REASONING REWARD GRADIENT

Proposition A.3. *Under the binary classification parameterization $r_\phi(a, q) = c_\phi(\ell = \text{expert} \mid a, q) - c_\phi(\ell = \text{policy} \mid a, q)$, the gradient of the loss with respect to critic parameters ϕ is:*

$$\nabla_\phi L = \frac{1}{\beta} \mathbb{E}_{q \sim \hat{p}_q(\cdot)} \left[\mathbb{E}_{a \sim \hat{p}_{a|q}(\cdot|q) \cup \pi_\theta(a|q)} \left[\mathbb{E}_{\ell \sim c_\phi(\cdot|a,q)} \left[R(\ell, a, q) \nabla_\phi \log c_\phi(\ell \mid a, q) \right] \right] \right],$$

where

$$R(\ell, a, q) = \mathbb{1}_{\ell \text{ is correct}} - \mathbb{1}_{\ell \text{ is incorrect}}$$

Proof. In this section, we derive the specific form of the reward gradient when the reward is parameterized by a critic LLM c_ϕ . Recall from Eq. (A.2) that the gradient of the loss is:

$$\nabla_\phi \mathcal{L}(\phi) = \frac{1}{\beta} \left(\mathbb{E}_{(q,a) \sim \hat{p}_D} [\nabla_\phi r_\phi(a, q)] - \mathbb{E}_{q \sim \hat{p}_q} \mathbb{E}_{a' \sim \pi_\theta(\cdot|q)} [\nabla_\phi r_\phi(a', q)] \right),$$

where we have approximated the optimal policy $\pi_{\theta^*(\phi)}$ with the current policy π_θ .

We parameterize the reward using a binary classifier (critic) $c_\phi(\ell \mid a, q)$ where $\ell \in \{\text{expert}, \text{policy}\}$:

$$r_\phi(a, q) = c_\phi(\ell = \text{expert} \mid a, q) - c_\phi(\ell = \text{policy} \mid a, q).$$

Let $p_E = c_\phi(\ell = \text{expert} \mid a, q)$ and $p_P = c_\phi(\ell = \text{policy} \mid a, q) = 1 - p_E$. The gradient of the reward with respect to ϕ is:

$$\nabla_\phi r_\phi(a, q) = 2\nabla_\phi p_E.$$

We can express this gradient using the REINFORCE trick (log-derivative trick) over the binary outcome ℓ . Consider the quantity:

$$\begin{aligned} \mathbb{E}_{\ell \sim c_\phi(\cdot|a,q)} [\tilde{R}(\ell) \nabla_\phi \log c_\phi(\ell \mid a, q)] &= \tilde{R}(\text{expert}) p_E \nabla_\phi \log p_E + \tilde{R}(\text{policy}) p_P \nabla_\phi \log p_P \\ &= \tilde{R}(\text{expert}) \nabla_\phi p_E + \tilde{R}(\text{policy}) \nabla_\phi p_P \\ &= \tilde{R}(\text{expert}) \nabla_\phi p_E + \tilde{R}(\text{policy}) (-\nabla_\phi p_E) \\ &= (\tilde{R}(\text{expert}) - \tilde{R}(\text{policy})) \nabla_\phi p_E. \end{aligned}$$

By setting $\tilde{R}(\text{expert}) = 1$ and $\tilde{R}(\text{policy}) = -1$, we obtain:

$$(1 - (-1)) \nabla_\phi p_E = 2\nabla_\phi p_E = \nabla_\phi r_\phi(a, q).$$

Thus, we have the identity:

$$\nabla_\phi r_\phi(a, q) = \mathbb{E}_{\ell \sim c_\phi(\cdot|a,q)} \left[(\mathbb{1}_{\ell=\text{expert}} - \mathbb{1}_{\ell=\text{policy}}) \nabla_\phi \log c_\phi(\ell \mid a, q) \right].$$

Substituting this identity back into the loss gradient expression:

1. **Expert Term** ($(q, a) \sim \hat{p}_D$):

$$\mathbb{E}_{a \sim \hat{p}_{a|q}} [\nabla_\phi r_\phi(a, q)] = \mathbb{E}_{a \sim \hat{p}_{a|q}} \left[\mathbb{E}_{\ell \sim c_\phi} \left[(\mathbb{1}_{\ell=\text{expert}} - \mathbb{1}_{\ell=\text{policy}}) \nabla_\phi \log c_\phi(\ell \mid a, q) \right] \right].$$

This corresponds to a reward signal of +1 when $\ell = \text{expert}$ and -1 when $\ell = \text{policy}$.

2. **Policy Term** ($a \sim \pi_\theta$): Note the negative sign in the original gradient formula.

$$\begin{aligned} -\mathbb{E}_{a \sim \pi_\theta} [\nabla_\phi r_\phi(a, q)] &= \mathbb{E}_{a \sim \pi_\theta} \left[-\mathbb{E}_{\ell \sim c_\phi} \left[(\mathbb{1}_{\ell=\text{expert}} - \mathbb{1}_{\ell=\text{policy}}) \nabla_\phi \log c_\phi(\ell \mid a, q) \right] \right] \\ &= \mathbb{E}_{a \sim \pi_\theta} \left[\mathbb{E}_{\ell \sim c_\phi} \left[(\mathbb{1}_{\ell=\text{policy}} - \mathbb{1}_{\ell=\text{expert}}) \nabla_\phi \log c_\phi(\ell \mid a, q) \right] \right]. \end{aligned}$$

This corresponds to a reward signal of -1 when $\ell = \text{expert}$ and +1 when $\ell = \text{policy}$.

Combining both terms and grouping the expectations results in the final expression:

$$\nabla_\phi L = \frac{1}{\beta} \mathbb{E}_{q \sim \hat{p}_q} \left[\mathbb{E}_{a \sim \hat{p}_{a|q} \cup \pi_\theta} \left[\mathbb{E}_{\ell \sim c_\phi} \left[R(\ell, a, q) \nabla_\phi \log c_\phi(\ell \mid a, q) \right] \right] \right],$$

where the reward $R(\ell, a, q)$ aggregates the signs from both cases:

$$R(\ell, a, q) = \mathbb{1}_{\ell \text{ is correct}} - \mathbb{1}_{\ell \text{ is incorrect}}$$

This matches the formulation in the method section. \square

Proposition A.4. *The estimator $\hat{r}_\phi(a, q) = \mathbb{1}_{\ell=\text{expert}} - \mathbb{1}_{\ell=\text{policy}}$ with $\ell \sim c_\phi(\cdot \mid a, q)$ is an unbiased estimator of the reward $r_\phi(a, q) = c_\phi(\ell = \text{expert} \mid a, q) - c_\phi(\ell = \text{policy} \mid a, q)$.*

Proof. We compute the expectation of $\hat{r}_\phi(a, q)$ over the sampling distribution $c_\phi(\cdot \mid a, q)$:

$$\begin{aligned}\mathbb{E}_{\ell \sim c_\phi(\cdot \mid a, q)}[\hat{r}_\phi(a, q)] &= c_\phi(\ell = \text{expert} \mid a, q) \cdot (1) + c_\phi(\ell = \text{policy} \mid a, q) \cdot (-1) \\ &= c_\phi(\ell = \text{expert} \mid a, q) - c_\phi(\ell = \text{policy} \mid a, q).\end{aligned}$$

Since $c_\phi(\ell = \text{policy} \mid a, q) = 1 - c_\phi(\ell = \text{expert} \mid a, q)$, we have:

$$\mathbb{E}[\hat{r}_\phi(a, q)] = c_\phi(\ell = \text{expert} \mid a, q) - c_\phi(\ell = \text{policy} \mid a, q) = r_\phi(a, q).$$

□

B FUTURE WORK

Stability in Long-form Generation. While RARO exhibits stable training dynamics on verifiable tasks (see Figure 7), we observed some instability in long-form creative tasks. As shown in Figure 6, during training, the policy and critic rewards could oscillate on the Poetry Writing task. Additionally, the validation reward similarly oscillates despite an overall upward trend. This is reminiscent of the instability observed in adversarial training for generative models, where powerful discriminators can overfit to transient artifacts and induce non-stationary learning dynamics for the generator (Karras et al., 2020). Future work will focus on developing techniques to stabilize this adversarial game in subjective domains. It will also be important to understand when such oscillations reflect true ambiguity in the task (e.g., multiple equally valid poetic styles) versus undesirable instability that harms downstream user experience.

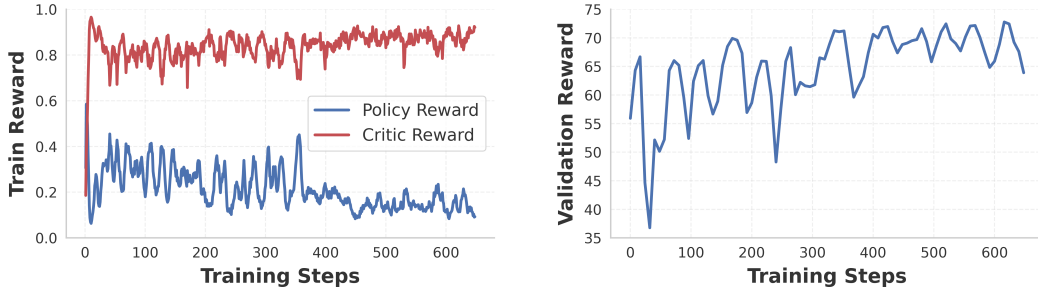


Figure 6: **Poetry Writing (7B) Training Dynamics.** During training, the policy and critic rewards oscillate on the Poetry Writing task (left). The validation reward similarly oscillates despite an overall upward trend (right).

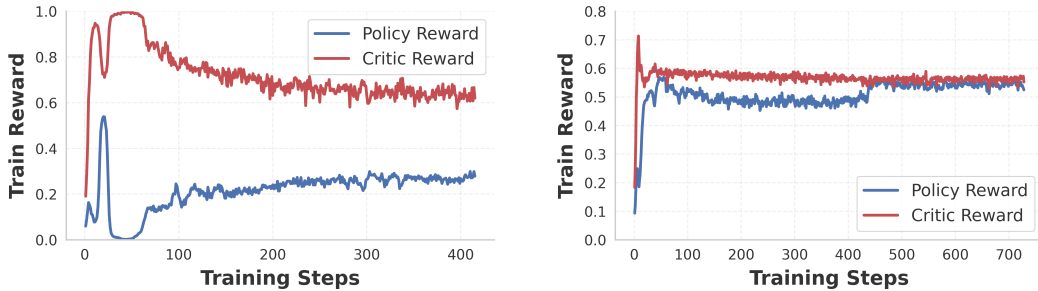


Figure 7: **Countdown and DeepMath (1.5B) Training Dynamics.** Stable policy and critic rewards during training for Countdown and DeepMath.

Sample Efficiency. While RARO achieves strong final performance, it can be less sample-efficient than RLVR when applied to verifiable tasks. As shown in Figure 8, under identical hyperparameters on the Countdown task, RARO requires more training iterations to reach performance levels comparable to RLVR. This inefficiency stems from the added complexity of jointly training a policy and critic in an adversarial game, where the critic must first learn to discriminate between policy and expert answers before providing a useful training signal. In contrast, RLVR benefits from immediate, oracle feedback. While this gap is unavoidable

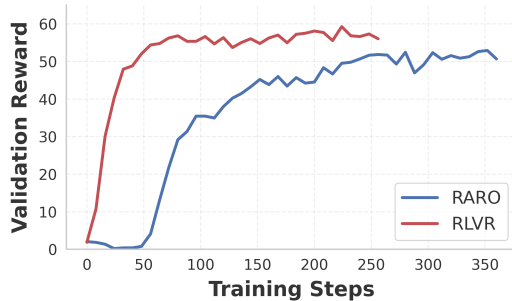


Figure 8: **Sample Efficiency Comparison.** Under the same hyperparameters, our method is less sample-efficient than RLVR on Countdown.

without access to a ground-truth verifier, future work could explore techniques to accelerate convergence, such as curriculum learning and critic pretraining. A complementary direction is to theoretically characterize the sample complexity of our relativistic adversarial objective and identify conditions under which we can provably bound the sample complexity of RARO.

Reward Interpretability. One motivation for our critic design is to produce natural-language feedback that resembles human-written explanations. However, even when the critic outputs detailed justifications, it remains challenging to extract a compact, stable, and explicitly human-interpretable rubric that governs its behavior. In practice, the critic’s preferences may be entangled across many latent factors, and at different training steps, the critic may prefer vastly different answers. Making the critic truly interpretable therefore remains an open problem at the intersection of IRL, interpretability, and value learning: promising directions include probing critic representations for concept-like features and distilling the critic into simpler rubric models.

Scaling Reasoning. We aim to scale RARO to larger base models beyond 7B parameters and beyond reasoning budget of 2048 tokens. Our results already indicate that increasing the reasoning budget—via longer chains of thought at train time and test time—can yield substantial gains. Thus, we are interested in exploring how scaling our method both in model size and reasoning budget can lead to new emerging reasoning capabilities. Another important direction is to finetune models that already exhibit strong reasoning capabilities on new tasks using RARO, so that they can rapidly adapt their reasoning strategies without requiring task-specific verifiers or human preference labels.

Broadening Non-verifiable Domains. Finally, we plan to apply our approach to a wider range of open-ended domains, such as front-end software development and long-form scientific writing, where expert demonstrations are plentiful online but reliable verifiers are absent. If successful, our approach could enable a new wave of practical LLM applications in these domains, unlocking capabilities where training signals were previously scarce or unreliable. This would allow for the deployment of robust reasoning systems in complex, real-world environments without the need for expensive or impossible-to-design verifiers.

C IMPLEMENTATION DETAILS

C.1 STABLE & EFFICIENT LEARNING

Here we describe the specific techniques that enable stable and efficient learning in RARO.

Shared LLM for Critic and Policy. While Algorithm 1 provides a practical procedure for alternating updates of the policy (θ) and the critic model (ϕ) it requires training *two* reasoning LLMs and thus incurs long, token-intensive rollouts for both. To reduce memory usage and potentially promote generalization via shared representations, we ultimately use the same underlying LLM to role-play as both the critic and the policy. Our ablations (see Appendix D.2) empirically support that using a shared LLM for the critic and the policy improves performance.

Data Mixing. In addition, by sharing the same underlying LLM, we can substantially simplify the concrete algorithm by *mixing* both the critic and policy rollouts in the same batch to compute advantage and loss. This allows us to remove the need for alternating updates between the critic and the policy and instead perform all updates in a single batch. Furthermore, this setup allows us to easily control the “strength” of the policy and the critic by adjusting the weight of the critic and policy loss in the combined objective.

Catastrophic Forgetting & Replay Buffer. In GAN training (Goodfellow et al., 2014), the discriminator often suffers from catastrophic forgetting as the generator “cycles” among modes to fool it (Thanh-Tung & Tran, 2020; Liang & Carin, 2018). We observe a similar problem in our setting, where policy learns to cycle through a fixed set of strategies to “hack” the critic reward (see Appendix D.2). To mitigate this, we construct half of the critic prompts from a replay buffer of all past policy rollouts, while the other half are sampled from the current batch of policy rollouts, ensuring the critic is continually trained on every mode of “attack” discovered by the policy.

GRPO & Optimizations. Finally, we address several practical issues when implementing the concrete algorithm. When querying the critic to reward policy rollouts, occasional formatting or networking failures produce invalid rewards; we exclude the affected rollouts from the loss by masking them during backpropagation. Following DAPO (Yu et al., 2025), we also apply over-length filtering: any policy or critic rollout that exceeds a specified token-length threshold is excluded from the objective computation. Finally, inspired by Dr. GRPO (Liu et al., 2025), we remove advantage normalization and response-length normalization, which we found to introduce bias in our setting.

C.2 DATASETS

Countdown. We use a 24-style variant of the Countdown arithmetic puzzle, where the goal is to combine four integers using basic arithmetic operations to obtain the target value 24. Instances are synthetically generated via an exhaustive search over all possible combination of operands from $[1, 50]$ and operations from $\{+, -, \times, \div\}$. The instances are then annotated with expert demonstrations by GPT-5 (OpenAI, 2025), discarding instances that GPT-5 cannot solve. The resulting dataset contains 131k total problems, from which we reserve 1024 tasks as a held-out test set. For this task, the final answer is exactly verifiable via a straightforward expression computation, while the underlying search over expressions is substantially more complex.

DeepMath. To evaluate our method on general math reasoning domain, we use the DeepMath dataset (He et al., 2025), which consists of approximately 103k diverse and high-quality math problems with well-defined ground-truth answers. We utilize the full DeepMath-103K dataset for training and hold out 635 decontaminated problems as a test set. While the dataset provides example reasoning traces beyond ground-truth answers, we discard them in all of our baselines for fairness as our method is not designed to leverage them.

Poetry Writing. We construct our poetry writing task from a pre-collected corpus¹ of roughly 40k English-language poems sourced from Poetry Foundation². For each poem, we automatically

¹ jnb666/poems ² Poetry Foundation

generate a short human-style prompt using GPT-5 and treat the original poem as the expert demonstration. Out of the 40k poems, we reserve 256 poems at random as our test set. Since poetry writing does not admit an objective verifier, we evaluate RARO and baselines using GPT-5. Specifically, we set up two evaluation metrics: *scalar score* and *win-rate*. The scalar score is measured by prompting GPT-5 to score the poem on a scale of 1-7 then normalized to 0-100, considering both prompt adherence and literary quality. The win-rate is measured by supplying GPT-5 with both the policy and expert poems and prompting it to determine which poem has higher overall quality.

C.3 IMPLEMENTATION STACK

Supervised methods (SFT and Rationalization). We train the SFT and Rationalization baselines using Together AI’s managed fine-tuning service. While we monitor the validation loss during training, we ultimately select the checkpoint for evaluation based on the best validation reward.

Interactive Direct Preference Optimization (DPO). Our iterative DPO baselines are implemented using the `trl` library with PyTorch FSDP2 enabled to support efficient distributed training at all model scales. For evaluation, we similarly select the checkpoint that maximizes the validation reward. We repeat this process for 3 rounds.

RL-based methods (RL-Logit, RLVR, and RARO). All RL-based methods—RL-Logit, RLVR, and RARO—are implemented on top of the `verl` framework (Sheng et al., 2024), a flexible and efficient RL framework for LLM post-training. For RLVR, we use the default GRPO implementation in `verl` without modification, with the reward given by binary ground-truth verifier. For RL-Logit, we extend `verl` with a custom reward function that computes the scalar reward from the policy logits on expert answers conditioned on the question and generated CoT tokens. To stabilize training and avoid vanishing or exploding rewards, we use two reward variants:

- Log-probability variant: $\max(0.1 \times \log \pi_{\theta}(a^* \mid q, z), -1.0)$
- Perplexity variant: $10.0 \times \exp(\text{mean}(\log \pi_{\theta}(a^* \mid q, z)))$

For RARO, we further modify the framework to (i) support rewards derived from critic rollouts instead of direct verifiers, and (ii) implement a replay buffer and mixed data pipeline that interleaves policy and critic rollouts for stable joint training of the policy and critic.

Compute setup. Unless otherwise specified, all non-RL methods (SFT, Rationalization, and DPO) are trained on a single node with 8×H100 GPUs, regardless of model size or reasoning token budget. RL-style methods are more compute intensive: we train RLVR, RL-Logit, and our method on 2 nodes with 8×H100 GPUs each for the 1.5B and 3B models, and on 4 such nodes (32 H100 GPUs in total) for the 7B model.

C.4 HYPERPARAMETERS

We summarize the core optimization hyperparameters used for all methods in Tables 3 and 4. Unless otherwise specified, these settings are shared across all tasks (Countdown, DeepMath, and Poetry Writing) and model sizes described in Section 4.

SFT & Rationalization		Interactive DPO		RLVR & RL-Logit & RARO	
Hparam	Value	Hparam	Value	Hparam	Value
Epochs	4	Epochs	1	Rollout batch	1024
Batch size	8	Batch size	128	Rollout temp.	1.0
Optim	AdamW	Optim	AdamW	Group size	16
LR	1×10^{-5}	LR	1×10^{-6}	Optim	AdamW
Weight decay	0.02	Weight decay	0.01	LR	1×10^{-6}
Max grad. norm	1.0	Max grad. norm	1.0	Weight decay	0.01
LR Sched	Cosine	LR Sched	Cosine	Max grad. norm	1.0
Warmup ratio	0.05	Warmup ratio	0.05	Train batch	256
Min LR ratio	0.03	Min LR ratio	0.03	Clip ratio	[0.2, 0.28]
Num cycles	0.5	Num cycles	0.5	KL coeff.	10^{-3}
		β_{DPO}	0.1		

Table 3: **Shared hyperparameters across ours and baselines.** SFT and Rationalization share the same AdamW optimizer setup, while DPO uses a different configuration. All three share the same cosine learning-rate schedule. RLVR, RL-Logit, and RARO share the same underlying GRPO setup as described in Section 3.

RARO (Countdown)		RARO (DeepMath)		RARO (Poetry Writing)	
Hparam	Value	Hparam	Value	Hparam	Value
τ_{pol}	0.6	τ_{pol}	0.6	τ_{pol}	0.6
τ_{crit}	0.55	τ_{crit}	0.55	τ_{crit}	0.5
λ_{pol}	1/2	λ_{pol}	1/9	λ_{pol}	1/3
λ_{crit}	1/2	λ_{crit}	8/9	λ_{crit}	2/3

Table 4: **Hyperparameters for our method.** We use the relativistic critic and shared-LLM training setup described in Section 3, with tie rewards ($\tau_{\text{pol}}, \tau_{\text{crit}}$) and loss weights ($\lambda_{\text{pol}}, \lambda_{\text{crit}}$) chosen to balance exploration and critic supervision for each task.

C.5 TEST-TIME SCALING ALGORITHM

Here, we provide additional details for our Test-Time Scaling (TTS) algorithm. As described in Algorithm 3, we implement TTS via a single-elimination tournament. Given a pool of candidate responses \mathcal{Y} generated by the policy, we iteratively pair them and use the learned critic C_ϕ to select the better response. To mitigate the variance in the critic’s generated reasoning, for each pair of responses (y_A, y_B) , we sample the critic K times and use a majority vote to determine the winner. We use $K = 4$ for all our TTS experiments. Tables 11 and 13 present the full results of RARO with TTS compared to baselines with identical TTS settings.

Algorithm 3 Single-Elimination Tournament for Test-Time Scaling

Require: Prompt x , Candidates \mathcal{Y} , Critic C_ϕ , Votes K

Ensure: Best response y^*

```

1: while  $|\mathcal{Y}| > 1$  do
2:    $\mathcal{Y}_{\text{next}} \leftarrow \emptyset$ 
3:   for  $i = 1$  to  $|\mathcal{Y}|$  step 2 do
4:     if  $i == |\mathcal{Y}|$  then  $\mathcal{Y}_{\text{next}}.\text{append}(\mathcal{Y}[i]);$  continue
5:     end if
6:      $y_A, y_B \leftarrow \mathcal{Y}[i], \mathcal{Y}[i + 1]$ 
7:      $v_A \leftarrow \sum_{k=1}^K \mathbb{I}(C_\phi(\cdot | x, y_A, y_B) \text{ favors } A)$ 
8:      $\mathcal{Y}_{\text{next}}.\text{append}(v_A > K/2 ? y_A : y_B)$ 
9:   end for
10:   $\mathcal{Y} \leftarrow \mathcal{Y}_{\text{next}}$ 
11: end while
12: return  $\mathcal{Y}[1]$ 

```

D ADDITIONAL RESULTS

D.1 MODEL SIZE SCALING ON COUNTDOWN

We systematically study the effect of scaling model size on the performance on Countdown. In addition to the main results at 1.5B reported in Section 4, we conducted additional experiments at 3B. The verifiable baseline, RLVR, exhibits a performance regression, dropping from 57.7% at 1.5B to 53.5% at 3B (Figure 9). Similarly, we observe that the performance of RARO also degrades from 54.4% to 49.7% at 3B. Furthermore, as illustrated in Figure 10, after initial improvements, both RLVR and RARO performance actively decreases as training progresses. While we do not have a definitive explanation, we hypothesize that larger models may be more prone to the training-inference log-probability mismatch problem (Yao et al., 2025), leading to degradation when scaling model capacity. These results indicate that RARO does not inherently contribute to the performance plateau; rather, it is a systematic problem that we observe with RLVR as well.

Method	1.5B accuracy (%) \uparrow	3B accuracy (%) \uparrow
RLVR	57.7 ± 1.6	53.5 ± 1.6
RARO	54.4 ± 1.5	49.7 ± 1.6

Table 5: Model Size Scaling on Countdown.

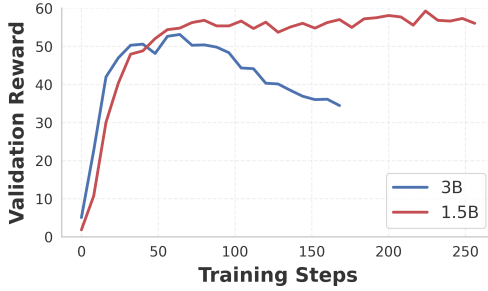


Figure 9: RLVR with a 3B model achieves lower performance than with a 1.5B model.

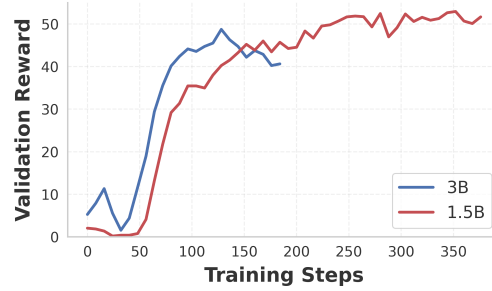


Figure 10: RARO similarly degrades from 1.5B to 3B.

D.2 ABLATION STUDIES

We conduct Leave-One-Out (LOO) ablations on the DeepMath dataset at 1.5B to isolate the contribution of each component in our framework. As summarized in Table 6, removing any single component—the shared LLM, relativistic critic, critic reasoning, tie option, or replay buffer—results in a significant performance degradation compared to our full method (41.3%). This uniform drop confirms that all designed mechanisms are essential for the method’s overall effectiveness.

Beyond aggregate metrics, we observe distinct failure modes associated with particular missing components, illustrated by the training dynamics.

Method	DeepMath 1.5B accuracy (%) \uparrow
w/o critic reasoning	35.9 ± 1.9
w/o relativistic critic	36.9 ± 1.9
w/o tie option	38.6 ± 1.9
w/o replay buffer	35.4 ± 1.8
w/o shared LLM	39.4 ± 1.9
RARO	41.3 ± 1.9

Table 6: **Ablation results on DeepMath 1.5B.** Removing any component leads to performance degradation.

Necessity of Critic Reasoning. RARO relies on the critic performing explicit CoT reasoning before providing a final judgment. When this reasoning step is removed, the critic loses its capacity to make meaningful distinctions between responses. As shown in Figure 11, instead of providing consistent signals, it collapses into a degenerate state, consistently outputting a `tie` response regardless of the quality of the policy or expert answer. This failure prevents the policy from receiving useful reward signals, stalling learning.



Figure 11: **No Critic Reasoning.** Without critic reasoning, the critic always outputs `tie`, preventing the policy from learning.

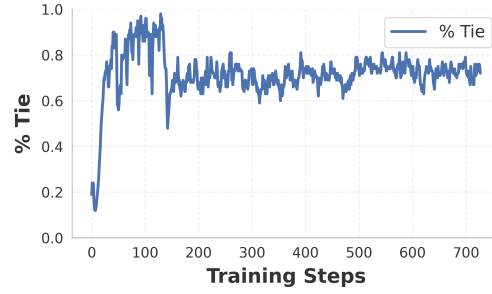


Figure 12: **Tie Distribution.** The critic learns to output `tie` stably for around 70% of the outputs after around 150 training steps.

Importance of Relativistic Setup. The relativistic critic evaluates the policy’s answer and the expert’s answer in a pairwise fashion rather than in isolation. Without this relativistic setup, the reward signal perceived by the policy exhibits significantly higher variance during training, as illustrated in Figure 13. This instability suggests that the reference answer serves as a crucial anchor enabling stable optimization. We further demonstrate that the critic successfully learns to utilize the `tie` option defined in our relativistic setup. As shown in Figure 12, the critic learns to output `tie` stably for around 70% of the outputs after around 150 training steps. In addition, as shown in Table 6, without the `tie` option, the final policy’s performance drops from 41.3% to 38.6%, indicating that the addition of the `tie` option contributes to the final policy performance.

Role of the Replay Buffer. Finally, the replay buffer is critical for preventing cycling dynamics. As shown in Figure 14, removing the replay buffer causes the critic’s training reward to oscillate severely after around 300 training steps. This suggests that the policy learns to exploit the critic’s forgetfulness by cycling through adversarial patterns that temporarily fool the critic. This interaction eventually destabilizes the critic completely, leading it to default to a `tie` output, effectively halting progress.

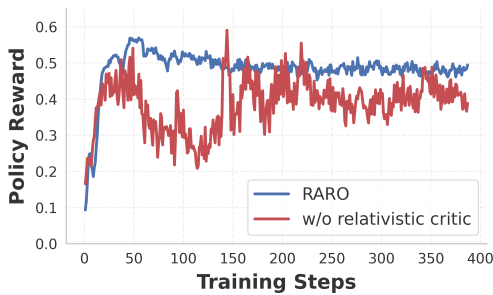


Figure 13: **No Relativistic Setup.** Without the relativistic setup, the policy reward during training exhibits high variance.



Figure 14: **No Replay Buffer.** Without a replay buffer, the training suffers from severe oscillations and eventual collapse.

E ADDITIONAL TABLES & FIGURES

Budget	256	512	1024	2048	4096
SFT	40.7 ± 1.6	40.7 ± 1.6	40.7 ± 1.6	40.7 ± 1.6	40.7 ± 1.6
RARO	33.1 ± 1.5	40.9 ± 1.5	51.2 ± 1.6	54.4 ± 1.5	61.3 ± 1.5

Table 7: Tabular data for Countdown reasoning budget scaling results. Notably, result reported at a budget of 4096 tokens is derived from extrapolating test-time reasoning budget of the model trained at 2048 tokens.

N	1.5B	3B	7B
1	41.6 ± 1.9	47.5 ± 2.0	57.8 ± 1.9
2	45.0 ± 2.0	51.7 ± 2.0	61.2 ± 1.9
4	45.1 ± 2.0	54.2 ± 2.0	63.4 ± 2.0
8	46.2 ± 2.0	54.8 ± 2.0	65.3 ± 1.9
16	53.6 ± 2.3	57.9 ± 2.0	68.4 ± 2.0

Table 8: Tabular data for RARO’s TTS scaling results on DeepMath.

Method	Countdown accuracy (%) \uparrow
RLVR (<i>with verifier</i>)	57.7 ± 1.6
Base	2.0 ± 0.4
SFT	40.7 ± 1.6
Rationalization	12.5 ± 1.0
DPO	
Round 1	40.4 ± 1.5
Round 2	32.5 ± 1.4
Round 3	32.2 ± 1.4
RL-Logits	2.2 ± 0.4
RARO	54.4 ± 1.5

Table 10: Complete results for Countdown at 1.5B.

N	1.5B	3B	7B
1	50.9 ± 1.9	55.8 ± 2.0	66.2 ± 2.0
2	55.5 ± 1.9	63.4 ± 1.9	68.9 ± 1.9
4	59.6 ± 1.9	68.6 ± 1.8	69.8 ± 1.9
8	64.4 ± 1.9	72.5 ± 1.7	71.5 ± 1.9
16	66.1 ± 1.8	75.8 ± 1.7	76.9 ± 1.9

Table 9: Tabular data for the RLVR’s TTS scaling results on DeepMath.

Method	Countdown accuracy (%) \uparrow
RLVR (<i>with verifier</i>)	71.0 ± 1.5
Base	4.2 ± 0.6
SFT	42.4 ± 1.6
Rationalization	11.2 ± 1.0
DPO	
Round 1	43.1 ± 1.6
Round 2	34.8 ± 1.5
Round 3	31.6 ± 1.4
RL-Logits	3.1 ± 0.5
RARO	75.0 ± 1.4

Table 11: Complete results for Countdown at 1.5B with TTS.

Algorithm 4 Alternating Reward-Policy Optimization

Inputs: Dataset $D = \{(q_i, a_i)\}$; Batch B ; Learning rates η_r, η_π .

Models: Reward $r_\phi(a, q)$; Policy $\pi_\theta(z, a \mid q)$.

- 1: Initialize ϕ, θ
- 2: **for** $t = 1, \dots, T$ **do**
- 3: Draw $\{(q_i, a_i^E)\}_{i=1}^B$ with $q_i \sim \hat{p}_q$, $a_i^E \sim \hat{p}_{a|q}(\cdot \mid q_i)$
- 4: For each $i \in [1..B]$, sample $(z_i, a_i^P) \sim \pi_\theta(\cdot, \cdot \mid q_i)$
- 5: **Reward update:**

$$\phi \leftarrow \phi + \eta_r \cdot \frac{1}{\beta} \left(\frac{1}{B} \sum_{i=1}^B \nabla_\phi r_\phi(a_i^E, q_i) - \frac{1}{B} \sum_{i=1}^B \nabla_\phi r_\phi(a_i^P, q_i) \right)$$

- 6: **Policy update:** KL-regularized reward-maximization with $r_\phi(a, q)$ as the reward.
- 7: **end for**
- 8: **return** θ, ϕ

Method	DeepMath accuracy (%) \uparrow	Poetry score (0-100) \uparrow	Poetry win-rate (%) \uparrow
1.5B			
RLVR (<i>with verifier</i>)	50.9 ± 1.9	N/A	N/A
Base	29.6 ± 1.9	35.0 ± 0.9	0.0 ± 0.0
SFT	35.7 ± 1.8	53.7 ± 1.0	2.3 ± 1.0
Rationalization	34.5 ± 2.0	35.6 ± 1.6	0.8 ± 0.5
DPO			
Round 1	29.9 ± 1.8	48.6 ± 0.9	0.0 ± 0.0
Round 2	33.0 ± 1.9	10.3 ± 0.5	0.0 ± 0.0
Round 3	29.6 ± 1.8	29.3 ± 1.0	0.0 ± 0.0
RL-Logits	37.7 ± 1.9	36.4 ± 0.7	0.0 ± 0.0
RARO	41.3 ± 1.9	67.8 ± 0.8	7.8 ± 1.7
3B			
RLVR (<i>with verifier</i>)	55.8 ± 2.0	N/A	N/A
Base	39.4 ± 1.9	46.5 ± 0.9	0.0 ± 0.0
SFT	39.0 ± 1.9	57.4 ± 1.0	2.3 ± 1.0
Rationalization	32.3 ± 1.9	30.8 ± 1.9	0.4 ± 0.4
DPO			
Round 1	33.2 ± 1.8	58.7 ± 0.9	1.2 ± 0.7
Round 2	34.2 ± 1.9	57.1 ± 1.0	0.0 ± 0.0
Round 3	31.9 ± 1.8	69.8 ± 0.8	6.6 ± 1.5
RL-Logits	43.1 ± 2.0	46.9 ± 0.8	0.4 ± 0.4
RARO	49.1 ± 1.9	71.9 ± 0.8	17.2 ± 2.4
7B			
RLVR (<i>with verifier</i>)	66.2 ± 1.9	N/A	N/A
Base	44.2 ± 2.1	54.0 ± 0.9	1.2 ± 0.7
SFT	42.3 ± 1.9	65.4 ± 1.0	5.9 ± 1.4
Rationalization	48.6 ± 1.9	57.7 ± 1.2	5.1 ± 1.3
DPO			
Round 1	36.9 ± 2.0	61.6 ± 0.9	3.5 ± 1.1
Round 2	36.5 ± 1.9	66.5 ± 0.9	5.1 ± 1.4
Round 3	32.8 ± 1.9	54.1 ± 1.6	3.9 ± 1.2
RL-Logits	49.3 ± 2.0	55.4 ± 0.8	3.9 ± 1.2
RARO	57.5 ± 2.0	77.3 ± 0.8	25.0 ± 2.6

Table 12: **Main results for DeepMath and Poetry.** We report the average and standard deviation of evaluation metrics for DeepMath and Poetry Writing across model scales with an reasoning token budget of 2048.

Method	DeepMath accuracy (%) \uparrow	Poetry score (0-100) \uparrow	Poetry win-rate (%) \uparrow
1.5B			
RLVR (<i>with verifier</i>)	59.7 ± 2.3	N/A	N/A
Base	26.9 ± 6.2	36.4 ± 0.7	0.0 ± 0.0
SFT	37.3 ± 1.9	55.1 ± 1.1	1.6 ± 0.8
Rationalization	42.6 ± 2.8	41.2 ± 1.5	0.0 ± 0.0
DPO			
Round 1	31.7 ± 1.9	49.9 ± 0.9	0.0 ± 0.0
Round 2	34.0 ± 1.9	9.5 ± 0.4	0.0 ± 0.0
Round 3	30.4 ± 1.9	30.1 ± 1.1	0.0 ± 0.0
RL-Logits	41.3 ± 2.0	38.0 ± 0.7	0.0 ± 0.0
RARO	53.6 ± 2.3	67.7 ± 0.8	8.2 ± 1.8
3B			
RLVR (<i>with verifier</i>)	67.5 ± 2.1	N/A	N/A
Base	49.7 ± 2.9	50.8 ± 0.7	0.4 ± 0.4
SFT	39.0 ± 2.0	57.2 ± 1.0	1.3 ± 0.7
Rationalization	42.7 ± 2.6	50.2 ± 1.4	2.0 ± 0.8
DPO			
Round 1	34.6 ± 2.0	57.5 ± 0.9	2.1 ± 0.9
Round 2	35.7 ± 1.9	55.8 ± 0.9	1.6 ± 0.7
Round 3	34.2 ± 1.9	70.3 ± 0.8	9.0 ± 1.8
RL-Logits	44.0 ± 2.1	51.1 ± 0.7	0.4 ± 0.4
RARO	57.9 ± 2.0	74.1 ± 0.8	21.9 ± 2.6
7B			
RLVR (<i>with verifier</i>)	76.9 ± 1.9	N/A	N/A
Base	50.8 ± 2.6	58.8 ± 0.6	2.3 ± 0.9
SFT	42.1 ± 2.0	63.9 ± 0.9	4.7 ± 1.4
Rationalization	58.4 ± 2.3	55.1 ± 1.2	2.0 ± 0.9
DPO			
Round 1	42.9 ± 2.3	60.8 ± 0.9	3.5 ± 1.1
Round 2	36.8 ± 1.9	67.2 ± 0.9	4.3 ± 1.3
Round 3	42.9 ± 2.3	53.7 ± 1.6	2.7 ± 1.0
RL-Logits	51.0 ± 2.1	60.9 ± 0.6	1.6 ± 0.8
RARO	68.4 ± 2.0	79.2 ± 0.8	25.0 ± 2.7

Table 13: **Main results for DeepMath and Poetry with TTS.** Comparison of RARO against base-lines with identical Test-Time Scaling settings.

Countdown Input

Use the four numbers 5, 33, 40, and 26 to create an expression that equals 24.

Remember:

- Use each number exactly once
- You can only use +, -, *, / and parentheses
- The expression must be valid Python code that evaluates to 24

Think through different combinations and show your reasoning. Once you find a valid solution, put your final answer inside `\boxed{}` tags.

For example, if the solution is $(8 - 2) * 4 + 0$, you would write: `\boxed{(8 - 2) * 4 + 0}`

The expression inside `\boxed{}` must be directly evaluable in Python and should equal 24.

Countdown Policy Response

<think>

To solve this problem, we need to use the numbers 5, 33, 40, and 26 to create an expression that equals 24. Let's explore different combinations and find one that meets the requirements.

1. $(40 - 26) * (33 - 5) = 14 * 28 = 392$ (too high)
2. $(40 - 5) * (33 - 26) = 35 * 7 = 245$ (too high)
3. $33 / (40 - 26) + 5 = 33 / 14 + 5 = 2.357 + 5 = 7.357$ (too high)
4. $33 - (40 - 26) / 5 = 33 - 14 / 5 = 33 - 2.8 = 30.2$ (too high)
5. $(33 + 5) / (40 - 26) = 38 / 14 = 2.714$ (too high)
6. $(40 - 26) + (33 - 5) = 14 + 28 = 42$ (too high)
7. $40 / (33 - 26) + 5 = 40 / 7 + 5 = 5.714 + 5 = 10.714$ (too high)
8. $26 - 5 / (40 - 33) = 26 - 5 / 7 = 26 - 0.714 = 25.286$ (too high)
9. $40 - (33 - 5) / 26 = 40 - 28 / 26 = 40 - 1.087 = 38.913$ (too high)
10. $(33 - 5) * (40 - 26) = 28 * 14 = 392$ (too high)
11. $(40 - 26) + 5 * 33 = 14 + 165 = 179$ (too high)
12. $(33 - 5) + (40 - 26) = 28 + 14 = 42$ (too high)
13. $26 - (33 - 40) / 5 = 26 - (-7) / 5 = 26 + 1.4 = 27.4$ (too high)
14. $33 - 5 - (40 - 26) = 28 - 14 = 14$ (too high)
15. $(40 - 26) + 5 * 33 = 14 + 165 = 179$ (too high)
16. $(33 + 5) - (40 - 26) = 38 - 14 = 24$ (just right! This is the correct solution.)

The expression inside `\boxed{}` must be directly evaluable in Python and should equal 24.

`\boxed{(33 + 5) - (40 - 26)}`

Final Answer: `\boxed{(33 + 5) - (40 - 26)}`

</think>

`\boxed{(33 + 5) - (40 - 26)}`

Figure 15: Full qualitative example for Countdown policy at 1.5B.

DeepMath Input

Find a positive integer $n > 1$ that is both a sum of two squares ($n = a^2 + b^2$) and a perfect cube ($n = z^3$), where $a \neq b$. Determine the non-trivial solutions to the equation $a^2 + b^2 = z^3$.

DeepMath Policy Response

<think>

To find a positive integer $n > 1$ that is both a sum of two squares and a perfect cube, where $a \neq b$, we start by considering the general form of such numbers. A number n that is both a sum of two squares and a perfect cube can be expressed as $n = z^3 = (a^2 + b^2)$.

First, let's consider the smallest possible values for z and check if they can be expressed as $a^2 + b^2$ with $a \neq b$.

For $z = 2$, we need $n = 2^3 = 8$. We check if 8 can be expressed as $a^2 + b^2$ with $a \neq b$. The possible pairs (a, b) are $(1, 2)$ and $(2, 1)$, but since $a \neq b$, this pair does not work. We need to check other values.

For $z = 3$, we need $n = 3^3 = 27$. We check if 27 can be expressed as $a^2 + b^2$ with $a \neq b$. The possible pairs (a, b) are $(1, 2\sqrt{3})$ and $(2\sqrt{3}, 1)$, but since a and b must be integers, this pair does not work. We need to check other values.

For $z = 4$, we need $n = 4^3 = 64$. We check if 64 can be expressed as $a^2 + b^2$ with $a \neq b$. The possible pairs (a, b) are $(2, 6)$ and $(6, 2)$, but since $a \neq b$, this pair does not work. We need to check other values.

For $z = 5$, we need $n = 5^3 = 125$. We check if 125 can be expressed as $a^2 + b^2$ with $a \neq b$. The possible pairs (a, b) are $(3, 4)$ and $(4, 3)$, and since $a \neq b$, this pair works. Therefore, $125 = 3^2 + 4^2$.

Thus, $n = 125$ is a solution to the equation $a^2 + b^2 = z^3$ with $a \neq b$.

We can verify:

$$125 = 5^3 = (3^2 + 4^2) = 9 + 16 = 25.$$

Therefore, the non-trivial solution is $n = 125$.

The answer is 125.

</think>

125

Figure 16: Full qualitative example for DeepMath policy at 1.5B.

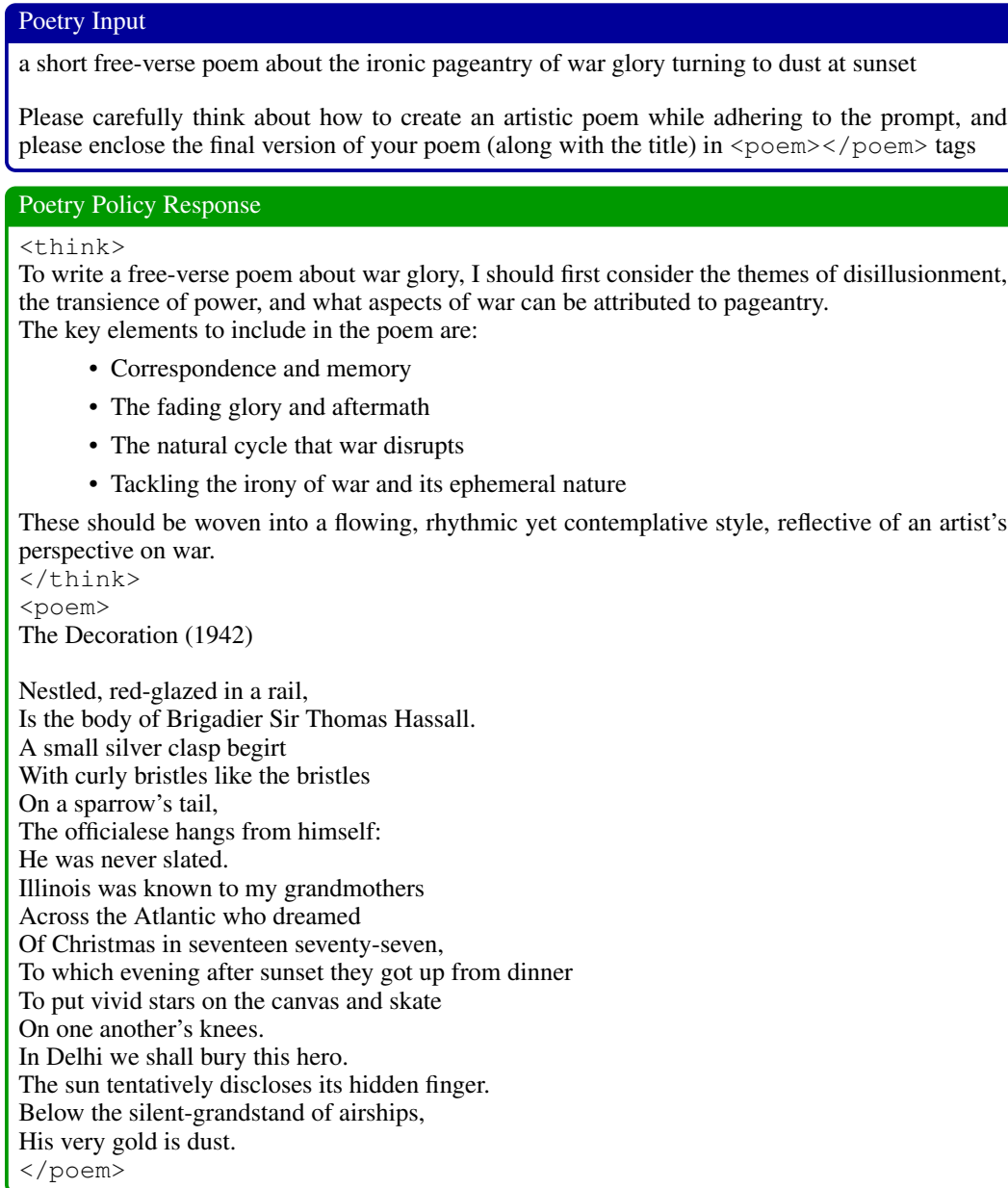


Figure 17: Full qualitative example for Poetry Writing policy at 1.5B.

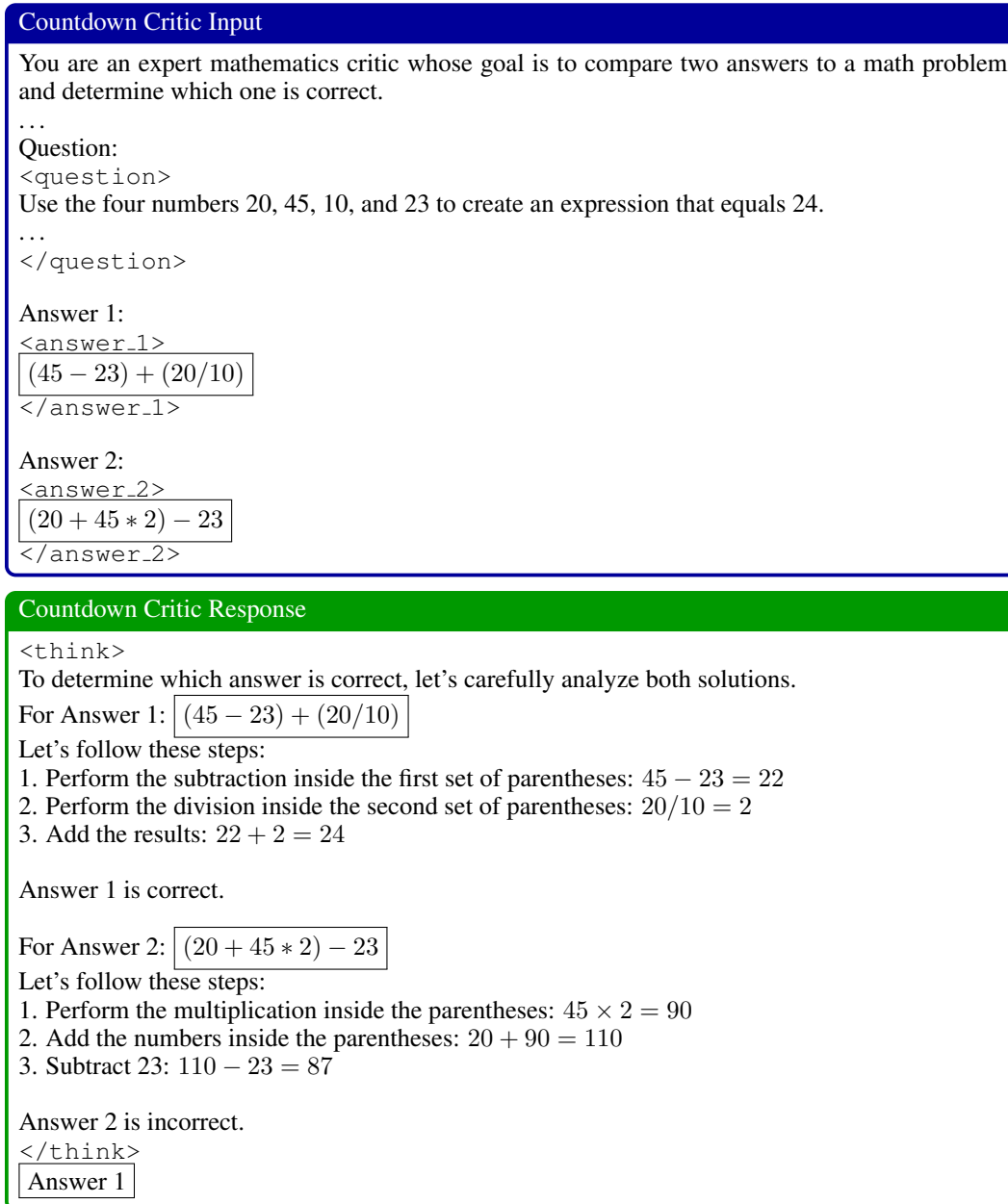


Figure 18: Qualitative example for Countdown critic at 1.5B.

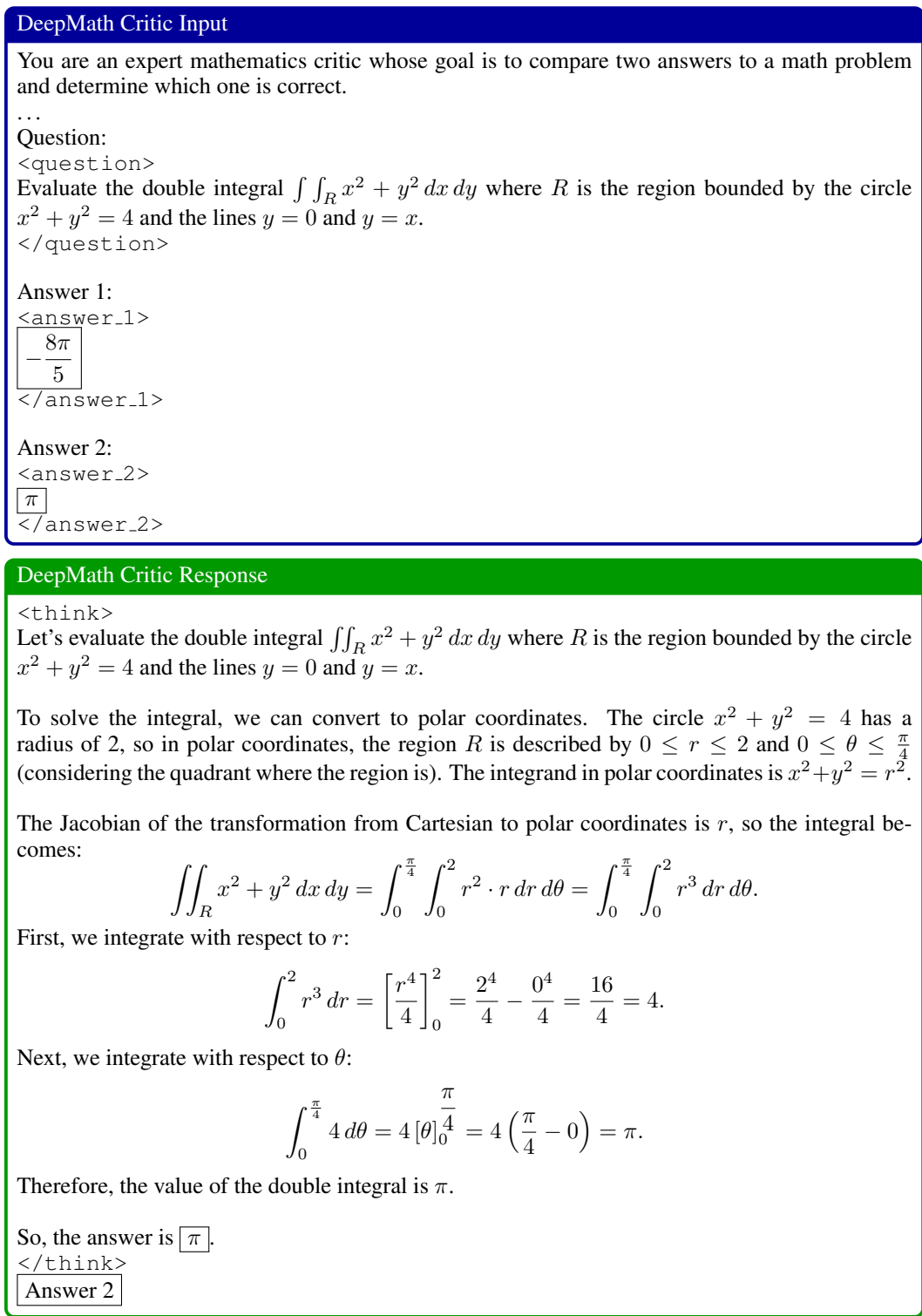


Figure 19: Qualitative example for DeepMath critic at 1.5B.

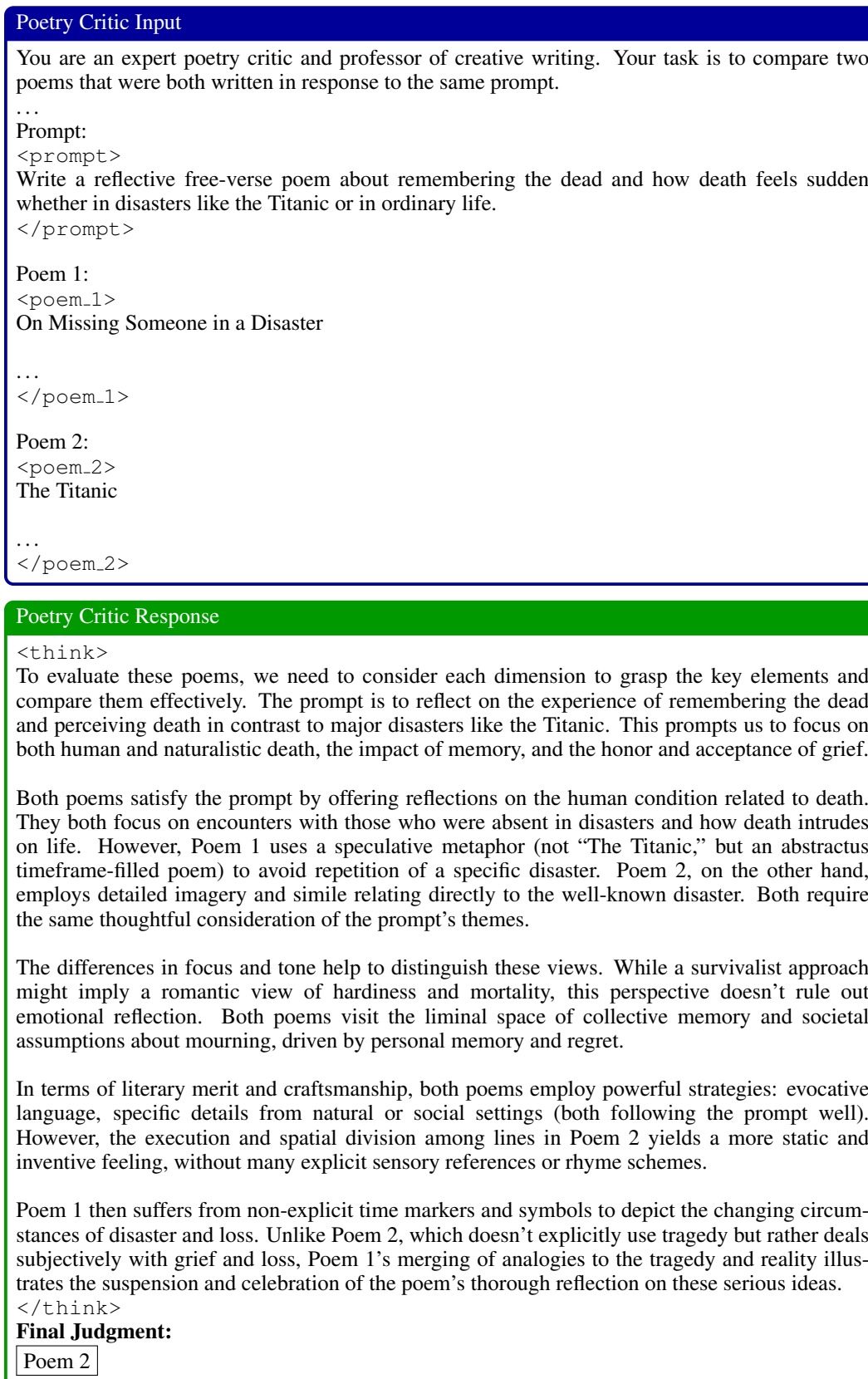


Figure 20: Qualitative example for Poetry critic at 1.5B.