Harnessing Negative Signals: Reinforcement Distillation from Teacher Data for LLM Reasoning

Shuyao Xu[2,1,*] **Cheng Peng**[1] **Jiangxuan Long**[1,*] **Weidi Xu**[1] **Wei Chu**[1] **Yuan Qi**[1,3]
[1]INFLY TECH (Shanghai) Co., Ltd. [2]National University of Singapore
[3]AI[3] Institute of Fudan University
shuyao@u.nus.edu, wdxu@inftech.ai

Code and models: `https://github.com/Tim-Siu/reinforcement-distillation`

Abstract

Recent advances in model distillation demonstrate that data from advanced reasoning models (e.g., DeepSeek-R1, OpenAI's o1) can effectively transfer complex reasoning abilities to smaller, efficient student models. However, standard practices employ rejection sampling, discarding incorrect reasoning examples—valuable, yet often underutilized data. This paper addresses the critical question: *How can both positive and negative distilled reasoning traces be effectively leveraged to maximize LLM reasoning performance in an offline setting?* To this end, We propose **Re**inforcement **Di**stillation (REDI), a two-stage framework. Stage 1 learns from positive traces via Supervised Fine-Tuning (SFT). Stage 2 further refines the model using both positive and negative traces through our proposed REDI objective. This novel objective is a simple, reference-free loss function that outperforms established methods like DPO and SimPO in this distillation context. Our empirical evaluations demonstrate REDI's superiority over baseline Rejection Sampling SFT or SFT combined with DPO/SimPO on mathematical reasoning tasks. Notably, the Qwen-REDI-1.5B model, post-trained on just 131k positive and negative examples from the open Open-R1 dataset, achieves an 83.1% score on MATH-500 (pass@1). Its performance matches or surpasses that of DeepSeek-R1-Distill-Qwen-1.5B (a model post-trained on 800k proprietary data) across various mathematical reasoning benchmarks, establishing a new state-of-the-art for 1.5B models post-trained offline with openly available data.

1 Introduction

Recent breakthroughs with large reasoning models, such as DeepSeek-R1 and OpenAI's o1, have demonstrated remarkable capabilities in complex reasoning tasks [DeepSeek-AI et al., 2025, OpenAI et al., 2024]. Techniques like test-time scaling facilitate longer Chain-of-Thought (CoT) processes and induce sophisticated reasoning behaviors, enhancing model performance in domains like mathematics. For base models initially lacking such advanced reasoning, two primary methods are employed to cultivate these abilities. The first, large-scale reinforcement learning (RL), directly applies RL algorithms to the base model, continually optimizing it through online exploration [DeepSeek-AI et al., 2025, Pan et al., 2025, Zeng et al., 2025]. However, RL approaches typically demand strong base models to achieve their full potential and are computationally intensive [Yue et al., 2025, DeepSeek-AI et al., 2025]. In contrast, distillation—learning from reasoning traces (e.g., CoT) generated by large "teacher" models—emerges as an attractive alternative for smaller, more efficient student models. This approach offers a practical and cost-effective pathway to extend their reasoning capabilities [Team, 2025, DeepSeek-AI et al., 2025]. Benefiting from open datasets distilled from

*Work done during an internship at INF AI (INFLY TECH (Shanghai) Co., Ltd).

Preprint. Under review.

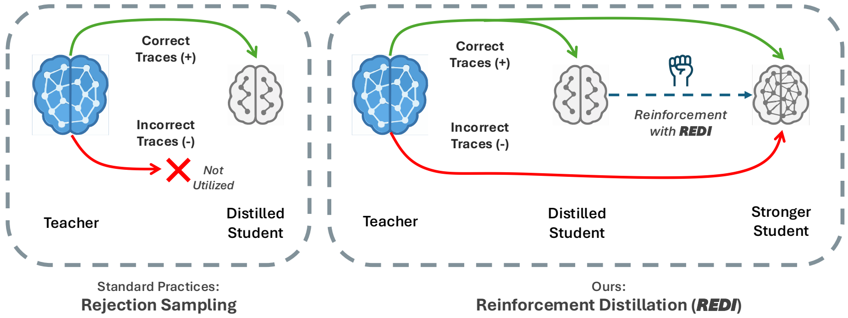

Figure 1: **Standard distillation practices via Rejection Sampling *vs.* our proposed Reinforcement Distillation (REDI).** Our REDI framework is able to utilize previously discarded incorrect reasoning traces generated by the teacher and yield stronger distilled models.

Table 1: **Model Performance Comparison (pass@1 over 16 samples) across reasoning benchmarks.** Our `Qwen-REDI-1.5B`, trained with the REDI framework on just 131k open data points, achieves the highest average score. This performance surpasses `DeepSeek-R1-Distill-Qwen-1.5B` (trained on 800k proprietary data) [DeepSeek-AI et al., 2025], demonstrating REDI's **remarkable data efficiency**. REDI enhances reasoning by effectively utilizing both positive and negative distilled examples. Values in **bold** indicate the best performance in each column. *Officially reported pass@1 results.

Model	MATH-500	AIME24	AMC23	Minerva	OlympiadBench	Avg.
Qwen2.5-Math-1.5B-Instruct*	75.8	10.0	60.0	**29.4**	38.1	42.7
DeepSeek-R1-Distill-Qwen-1.5B	**83.2** $\pm$**0.2**	**28.3** $\pm$**1.0**	62.1 $\pm$0.8	26.0 $\pm$0.4	43.1 $\pm$0.3	48.6 $\pm$0.3
Qwen-SFT-1.5B-5ep (SFT Baseline)	80.4 $\pm$0.3	21.9 $\pm$1.1	57.5 $\pm$0.6	27.5 $\pm$0.4	41.5 $\pm$0.3	45.8 $\pm$0.3
Qwen-REDI-1.5B	83.1 $\pm$0.3	28.1 $\pm$1.1	**62.4** $\pm$**0.6**	28.8 $\pm$0.3	**45.2** $\pm$**0.2**	**49.5** $\pm$**0.3**

powerful reasoning models like DeepSeek-R1 [Team, 2025, Labs, 2025, Face, 2025], openly post-trained models have shown strong performance [Team, 2025, Ye et al., 2025, Muennighoff et al., 2025, Wen et al., 2025], although a performance gap remains compared to their closed-data counterparts.

However, current distillation methodologies predominantly rely on rejection sampling, which involves leveraging only positive reasoning examples—those whose final answers are verified. This practice means that negative examples, despite the significant computational effort invested in their generation, are typically underutilized. We hypothesize that these negative traces contain vital insights into common pitfalls and nuanced errors from which smaller models could learn, thereby further unlocking the potential of open distilled data. This leads to the central research question we address:

How can we effectively leverage both positive and negative distilled reasoning traces to maximize LLM reasoning performance with a fixed distilled open dataset?

To address this challenge, we propose **Re**inforcement **Di**stillation (REDI), a two-stage offline training framework designed to explicitly utilize both positive and negative distilled reasoning traces. In Stage 1, we perform standard supervised fine-tuning (SFT) using only positive traces to establish a robust reasoning foundation. Stage 2 further refines this model by incorporating negative traces, effectively transforming "mistakes" into valuable learning signals. Our preliminary investigations explored established off-policy preference optimization methods, such as Direct Preference Optimization (DPO) [Rafailov et al., 2024] and SimPO [Meng et al., 2024]. A key empirical finding from these studies is that the implicit Kullback-Leibler (KL) divergence penalty β, while beneficial for stabilizing offline training and permitting larger gradient step sizes, often caps test-time performance. Motivated by this observation, we explored alternative training objectives that eliminate such regularization terms. We found that a simple reference-free objective—analogous to the $\beta \to 0$ limit of DPO/SimPO objectives, which directly maximizes the likelihood of positive traces while simultaneously minimizing the likelihood of negative traces—can yield higher peak performance. Building upon this, REDI introduces an asymmetric weighting strategy. By down-weighting the gradient contribution of negative samples, our framework achieves both enhanced stability and superior test-time performance.

Our key contributions are:

1. We propose Reinforcement Distillation (REDI), a cost-effective and data-efficient two-stage training framework designed to enhance reasoning model distillation by incorporating negative traces. Central to REDI is our novel, asymmetrically weighted, reference-free objective that

effectively utilizes previously discarded negative signals, leading to performance gains without requiring costly online interaction.

2. We provide a detailed analysis highlighting the trade-off between performance and stability associated with KL regularization in DPO when applied to off-policy distilled data in the reasoning domain. This analysis motivates the design of our REDI objective.
3. We empirically demonstrate that REDI consistently outperforms Rejection Sampling SFT and SFT combined with DPO/SimPO on mathematical reasoning tasks. Our `Qwen-REDI-1.5B` model, as highlighted in Table 1, achieves state-of-the-art results among 1.5B models post-trained offline with openly available data, showcasing REDI's effectiveness.

The remainder of this paper is organized as follows: Section 2 details the Reinforcement Distillation methodology and the REDI objective. Section 3 describes the experimental setup. Section 4 presents our results and analysis. Section 5 discusses related work, and Section 6 concludes the paper.

2 Methodology: Reinforcement Distillation (REDI)

2.1 Problem Setting and Data

We operate in an offline distillation setting with a fixed dataset collected via a common distillation pipeline. The dataset originates from a set of problems, each denoted by x. For each problem x, a capable "teacher" model is employed to generate reasoning traces. The generation process for a specific problem x continues until a correct reasoning trace, y_w, is successfully produced. During these attempts, incorrect traces, y_l, might also be generated before y_w is obtained.

From these generated traces, we construct two distinct datasets for our two-stage training framework:

1. **Positive Traces Dataset ($\mathcal{D}_{\text{SFT}}$):** This dataset comprises all pairs (x, y_w), where y_w is a correct reasoning trace generated by the teacher for problem x.
2. **Preference Pairs Dataset ($\mathcal{D}_{\text{Pref}}$):** This dataset is constructed from the subset of problems x for which at least one incorrect trace was generated before the correct trace y_w was obtained. For each such problem x, we form a preference tuple (x, y_w, y_l) by pairing its correct trace y_w with *one* selected incorrect trace y_l generated for the same problem. This selection strategy is adopted for simplicity and aligns with observations from datasets like Open-R1 [Face, 2025], where most problems with negative examples feature only one such instance.

Our overall objective is to train a student LLM, π_θ, to maximize its reasoning performance by effectively leveraging all information within the pre-collected $\mathcal{D}_{\text{SFT}}$ and $\mathcal{D}_{\text{Pref}}$ datasets.

2.2 The Reinforcement Distillation (REDI) Framework

2.2.1 Stage 1: Supervised Fine-Tuning (SFT) on Positive Traces

The first stage involves standard Supervised Fine-Tuning (SFT) of the base LLM on the $\mathcal{D}_{\text{SFT}}$ dataset, which contains only positive (correct) reasoning traces (x, y_w). The SFT objective is to maximize the likelihood of generating the correct trace y_w given the problem x:

$$\mathcal{L}_{\text{SFT}}(\theta) = - \mathop{\mathbb{E}}_{(x, y_w) \sim \mathcal{D}_{\text{SFT}}} \left[\log \pi_\theta(y_w | x) \right]. \tag{1}$$

This initial SFT stage serves several key purposes. Primarily, it adapts the base model to the specific style and format of the reasoning traces. Furthermore, it provides a strong initial policy, denoted as π_{SFT}, which can subsequently serve as a reference for methods like DPO or as the starting point for our REDI objective in the second stage. Finally, this stage establishes a baseline performance comparable to traditional SFT-only pipelines (i.e., training solely on positive examples), allowing us to quantify the gains achieved by later incorporating negative examples.

2.2.2 Stage 2: Reinforcement with Positive and Negative Traces

The second stage aims to further refine the model obtained from Stage 1 by leveraging the negative signals encoded in $\mathcal{D}_{\text{Pref}}$, which contains pairs of positive (y_w) and negative (y_l) traces for the same problem x.

Preliminary study. To contextualize our REDI objective, we first briefly review established preference optimization methods such as DPO [Rafailov et al., 2024] and SimPO [Meng et al., 2024].

DPO optimizes the policy π_θ to align with human or model preferences while regularizing its deviation from a reference policy π_{ref} (typically π_{SFT} from Stage 1). Its loss function is:

$$\mathcal{L}_{\text{DPO}}(\theta; \pi_{\text{ref}}) = -\mathop{\mathbb{E}}_{(x,y_w,y_l)\sim\mathcal{D}_{\text{Pref}}}\left[\log\sigma\left(\beta\left(\log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \log\frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right)\right)\right], \tag{2}$$

where $\sigma(\cdot)$ is the sigmoid function. The hyperparameter β controls the strength of an implicit KL divergence penalty against π_{ref}, where larger β values imply stronger regularization.

SimPO offers a reference-free alternative that incorporates sequence length normalization and an explicit margin γ:

$$\mathcal{L}_{\text{SimPO}}(\theta) = -\mathop{\mathbb{E}}_{(x,y_w,y_l)\sim\mathcal{D}_{\text{Pref}}}\left[\log\sigma\left(\beta\left(\frac{\log\pi_\theta(y_w|x)}{|y_w|} - \frac{\log\pi_\theta(y_l|x)}{|y_l|}\right) - \gamma\right)\right]. \tag{3}$$

Here, $|y|$ denotes the length (e.g., number of tokens) of sequence y. In SimPO, we also observe that β influences learning dynamics with a regularizing effect. Higher β corresponds to higher stability.

As empirically demonstrated in Section 4.2, while stronger regularization (e.g., higher β in DPO or SimPO) can enhance training stability and permit larger gradient steps, it often results in lower peak model performance.

Towards a regularization-free objective. The observed trade-off between performance and stability associated with β in methods like DPO and SimPO motivates exploring objectives that minimize or eliminate such explicit regularization. As detailed in Appendix B, considering the $\beta \to 0$ limit of preference optimization objectives like SimPO yields the following simplified, regularization-free loss function (to be minimized):

$$\mathcal{L}_{\text{symm}}(\theta) = \mathop{\mathbb{E}}_{(x,y_w,y_l)\sim\mathcal{D}_{\text{Pref}}}\left(-\frac{\log\pi_\theta(y_w|x)}{|y_w|} + \frac{\log\pi_\theta(y_l|x)}{|y_l|}\right). \tag{4}$$

As empirically demonstrated in Section 4.3, this symmetric, reference-free objective (Eq. (4)) can achieve performance comparable to meticulously tuned DPO or SimPO, offering reduced hyperparameter tuning. Nevertheless, the tension between performance and stability persists: careful learning rate tuning remains crucial, as larger learning rates, while potentially accelerating learning and improving transient performance, often lead to training collapse.

The REDI objective: asymmetric weighting for stability and performance. During experiments with DPO, SimPO, and the symmetric objective $\mathcal{L}_{\text{symm}}$ (Section 4), we observed frequent training collapses when learning rates were inadequately tuned. Collapse manifests as a rapid decrease in the likelihood of both positive (y_w) and negative (y_l) responses, accompanied by declining task accuracy. Recent studies attribute this instability to unintended side effects of off-policy gradients [Yan et al., 2025, Razin et al., 2025, Ren and Sutherland, 2025]. Specifically, gradient updates penalizing negative responses may inadvertently suppress semantically similar *positive* responses, leading to degenerate solutions. Heuristic mitigations include auxiliary SFT losses or asymmetric β tuning [Pang et al., 2024, Yan et al., 2025].

Inspired by these insights, we propose **asymmetric weighting** for the simplified objective (Eq. (4)). By down-weighting gradients from negative traces, we preserve stability while maximizing peak performance.

The **REDI objective**, central to the second stage of our framework, refines the model using an asymmetrically weighted, reference-free loss. The REDI loss to be minimized is defined as:

$$\mathcal{L}_{\text{REDI}}(\theta) = \mathop{\mathbb{E}}_{(x,y_w,y_l)\sim\mathcal{D}_{\text{Pref}}}\left[-\frac{\log\pi_\theta(y_w|x)}{|y_w|} + \alpha\cdot\frac{\log\pi_\theta(y_l|x)}{|y_l|}\right], \tag{5}$$

where $\alpha \in [0, 1]$ controls the penalty strength for negative traces:

- $\alpha = 0$: Reduces to SFT on positive traces (ignores negatives).
- $\alpha = 1$: Recovers the symmetric objective (Eq. (4)).

The REDI objective, when optimized using gradient descent with an appropriate learning rate schedule (such as the one in Section 3), is amenable to standard convergence analysis. Under typical L-smoothness assumptions for the loss function, this optimization process is guaranteed to converge to a stationary point. Further details and a formal proof are provided in Appendix A. The asymmetric weighting ($\alpha < 1$) moderates gradient contributions from positive and negative samples, preventing collapse while maintaining aggressive learning dynamics.

3 Experimental Setup

3.1 Data Curation

Following the data pipeline described in Section 2.1, we derived two datasets from the OpenR1-Math-Raw corpus [Face, 2025]; the `cn_k12` subset was excluded due to its lower relative difficulty. The OpenR1-Math-Raw corpus provides two labels for correctness: one from the Llama judge and one from Math-Verify [Kydlíček]. A response was considered correct if both labels were "True"; otherwise, it was considered incorrect. More details are discussed in Appendix C.2.

The two datasets were constructed as follows:

- **Positive Traces Dataset ($\mathcal{D}_{\text{SFT}}$):** This dataset contains 78k problem-solution pairs (x, y_w), where y_w represents a correct reasoning trace. It was used for SFT in Stage 1.
- **Preference Pairs Dataset ($\mathcal{D}_{\text{Pref}}$):** This dataset consists of 53k triplets (x, y_w, y_l), where y_w is a correct trace and y_l is an incorrect trace for the same problem x. It was utilized in Stage 2.

3.2 Training Configuration

Stage 1 Configuration: In the first stage, we establish strong SFT baselines by fine-tuning the base `Qwen2.5-Math-1.5B` model on the $\mathcal{D}_{\text{SFT}}$ dataset. Two SFT baselines were prepared:

- `Qwen-SFT-1.5B-3ep`: This model was trained for 3 epochs on $\mathcal{D}_{\text{SFT}}$. It served as the initial checkpoint for our comparative studies involving DPO, SimPO, and various REDI configurations.
- `Qwen-SFT-1.5B-5ep`: Observing continued SFT performance improvement beyond 3 epochs (as shown in Section 4.1), this model was trained for 5 epochs on $\mathcal{D}_{\text{SFT}}$. This stronger SFT variant was used as the starting point for training our final `Qwen-REDI-1.5B` model.

For this SFT stage, all models were trained using the AdamW optimizer [Loshchilov and Hutter, 2019] with a batch size of 128. The learning rate schedule featured a linear warmup for the initial 10% of total training steps, followed by a linear decay to zero.

Stage 2 Configuration: The second stage involves further refining the SFT-tuned models using the $\mathcal{D}_{\text{Pref}}$ dataset, which contains preference pairs (x, y_w, y_l). We applied DPO, SimPO, and our proposed REDI objective to the SFT checkpoints obtained from Stage 1. All preference tuning methods were trained for 1 epoch over the $\mathcal{D}_{\text{Pref}}$ dataset. Similar to Stage 1, the AdamW optimizer and the same learning rate schedule (10% warmup, then linear decay) were used. The batch size for this stage was 32. Specific hyperparameter settings for DPO (e.g., β values, learning rates), SimPO (e.g., β, γ values, learning rates), and REDI (e.g., α values, learning rates) were carefully tuned, with detailed ranges and chosen values provided in Appendix C.4.

3.3 Evaluation Protocol

During all evaluations, generated samples were decoded using a temperature of 0.6, Top P sampling with $p = 0.95$, and a maximum generation length of $32,768$ tokens.

Protocols: We utilized two distinct configurations for evaluating model performance:

- *Intermediate Evaluations:* These evaluations, used for hyperparameter tuning, performance plotting, and ablation studies, were conducted using LightEval [Habib et al., 2023] on the MATH-500 benchmark [Lightman et al., 2023]. Performance was measured as pass@1, averaged over 4 generated samples per problem.
- *Final Model Evaluations:* These evaluations, presented in comparison tables (e.g., Table 1), were performed using the DeepScaleR/rllm [Luo et al., 2025] framework on the mathematics

benchmarks MATH-500, AIME24, AMC23, Minerva and out-of-distribution STEM benchmark OlympiadBench [Lewkowycz et al., 2022, He et al., 2024]. Performance was measured as pass@1 (averaged over 16 generated samples) per problem for Tables 1 and 2, and pass@16 for discussions in Section 4.6.

Reporting and SEM Calculation: The pass@k benchmark scores represent the proportion of problems for which at least one of k generated samples is correct. When reporting pass@1 averaged over multiple attempts for main results, we also include standard error of the mean (SEM). See Appendix C.5 for the calculation of SEM.

4 Results and Analysis

4.1 Performance Limits of SFT-Only Training

We first establish the performance achievable using only positive distilled data via Supervised Fine-Tuning (SFT). As illustrated by Figure 2, performance increases for approximately 5 epochs before eventually plateauing. This observation highlights the limitations of learning solely from positive traces and motivates the utilization of negative signals.

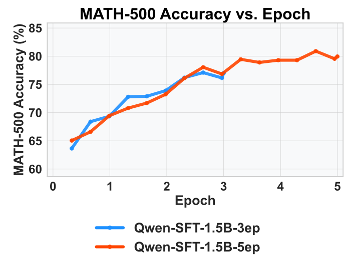

Figure 2: SFT MATH-500 accuracy vs. training epochs.

4.2 Performance-Stability Tradeoff in DPO

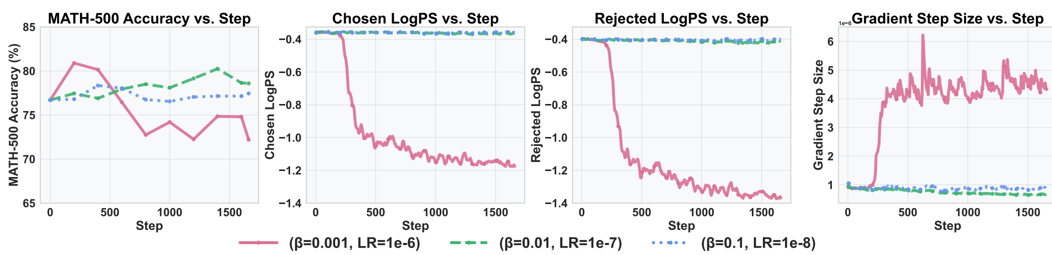

Figure 3: **DPO training dynamics with respect to** β, when initial gradient step sizes are controlled to be similar. LogPS visualizes the average per-token log probability of the model generating the chosen or rejected response. Gradient Step Size refers to the norm of the parameter update.

DPO dynamics with varying β and similar initial gradient step sizes. Figure 3 illustrates DPO training dynamics for three configurations: $(\beta = 0.001, \text{LR} = 1\times10^{-6})$, $(\beta = 0.01, \text{LR} = 1\times10^{-7})$, and $(\beta = 0.1, \text{LR} = 1 \times 10^{-8})$. The learning rates were selected such that the initial gradient step sizes were comparable across these runs, as indicated in the "Gradient Step Size vs. Step" subplot. The subsequent dynamics revealed a trade-off:

- The lowest β setting (0.001) achieved the highest peak accuracy (approximately 80.9% on MATH-500) but subsequently experienced training collapse. This collapse in accuracy was accompanied by a sharp drop in chosen and rejected LogPS and a surge in gradient step size.
- Higher β values (0.01, 0.1) maintained stability throughout training but achieved lower peak accuracies (approximately 80.3% and 78.3%, respectively).

This exploration suggests that when initial gradient step sizes are matched, stronger KL regularization (higher β) yields more stable training, but performance can be constrained.

Optimizing learning rates for different β values. To further investigate whether the performance ceiling observed with higher β values is an inherent limitation, we conducted learning rate (LR) sweeps for fixed β values of 0.001 and 0.01 (Figure 4). This allows for a fairer comparison, as stronger regularization (higher β) can often accommodate larger gradient steps.

- For $\beta = 0.001$, an LR of 2×10^{-7} yielded the best peak performance at step 1000, reaching approximately 82.3% on MATH-500.
- For $\beta = 0.01$, an LR of 2×10^{-7} achieved the best peak for this β value at step 1600, at approximately 81.2%.

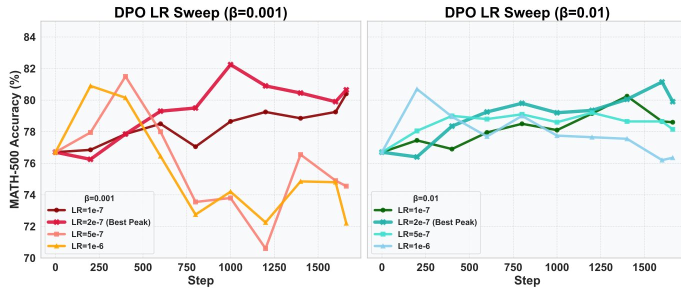

Figure 4: DPO MATH-500 accuracy with learning rate sweeps for $\beta = 0.001$ and $\beta = 0.01$.

Comparing the best-tuned runs from Figure 4, the configuration with the lower $\beta = 0.001$ still achieved a significantly higher peak accuracy.

Observation: Stability and Peak Performance Trade-off in DPO with KL Regularization

DPO's β parameter, which controls KL regularization, presents a critical trade-off. Higher β values enhance training stability, often allowing for more aggressive learning rates. However, our experiments suggest that even with tuned LRs, **higher β may restrict peak performance**. Conversely, lower β values can unlock higher peak accuracies.

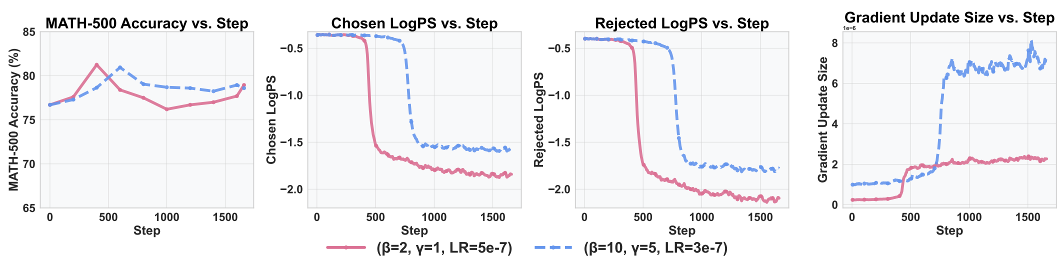

Figure 5: SimPO training dynamics.

Similar performance-stability tradeoff observed for SimPO. Preliminary experiments were also conducted with SimPO (Figure 5). We found that with a fixed γ/β ratio (0.5 in our tests), higher β values correspond to stronger regularization effects. We experimented with $(\beta = 2, \gamma = 1, \text{LR} = 5 \times 10^{-7})$ and $(\beta = 10, \gamma = 5, \text{LR} = 3 \times 10^{-7})$. The $\beta = 10$ run had a larger initial gradient update size and demonstrated greater stability (i.e., it "collapsed" later than the $\beta = 2$ run). However, its peak performance on MATH-500 was slightly lower than that of the $\beta = 2$ run before its collapse. This reinforces the observation of a trade-off between stability and attainable peak performance.

4.3 REDI: Achieving Stability and Performance with Asymmetric Weighting

Our REDI method directly optimizes log-likelihoods without KL regularization against a reference model, relying instead on asymmetric weighting to manage stability.

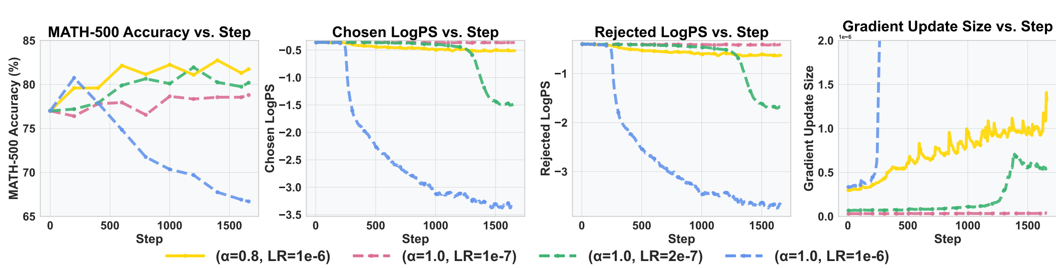

Figure 6: Comparison of Symmetric REDI ($\alpha = 1.0$) and Asymmetric REDI ($\alpha = 0.8$).

Symmetric REDI ($\alpha = 1.0$). Figure 6 shows that the Symmetric REDI objective exhibits dynamics similar to DPO with low β. A high LR (1×10^{-6}) leads to rapid learning (peaking around 80.8% MATH-500 accuracy) but then collapses, evidenced by sharp drops in chosen and rejected LogPS, as well as accuracy. However, reducing the learning rate significantly improves training stability. The ablation table (Table 2) further shows that a more stable symmetric REDI run ($\alpha = 1.0, \mathrm{LR} = 2 \times 10^{-7}$) achieves 81.7% on MATH-500, comparable to the best-tuned DPO result (81.3%). This suggests that *a simpler, regularization-free objective can indeed match DPO's performance when its LR is carefully tuned*. Nevertheless, the trade-off between performance and stability persists. For instance, the stable $\mathrm{LR} = 1 \times 10^{-7}$ run, while avoiding LogPS collapse, achieves a lower peak accuracy than the unstable $\mathrm{LR} = 2 \times 10^{-7}$ run. This trade-off is particularly evident if we focus on the first 200 steps, where the least stable run with $\mathrm{LR} = 1 \times 10^{-6}$ achieves the highest accuracy (learns the fastest) before collapsing.

Asymmetric weighting ($\alpha < 1.0$) is key for REDI. Figure 6 (yellow solid line) demonstrates that REDI with $\alpha = 0.8$ and a high LR of 1×10^{-6} achieves rapid learning, comparable to the symmetric $\alpha = 1.0$ high-LR run, but crucially, it *does not collapse*. It reaches a high peak performance and maintains it. The chosen and rejected LogPS do not suffer from collapse, and the gradient update size remains controlled.

Insight: Asymmetric REDI ($\alpha < 1.0$) Balances Stability and Performance

Asymmetric weighting in the REDI objective (specifically, $\alpha < 1.0$, with $\alpha = 0.8$ proving effective in our experiments) greatly impacts the training dynamics. In our experiments, it allows the use of more aggressive learning rates, which leads to faster learning and the ability to achieve higher peak performance, while simultaneously preventing the training collapse often observed in unregularized or symmetrically weighted objectives when using high LRs.

4.4 Tuning the Asymmetric Weighting Factor α in REDI

We studied $\alpha \in \{0.2, 0.5, 0.8\}$ and found that $\alpha = 0.8$ provided the best balance for achieving strong test-time performance while maintaining stability. Lowering α further (e.g., to 0.5 or 0.2) tended to degrade peak performance. This is intuitive, as lower α values make the objective more similar to SFT on positive examples only, which we have shown to plateau earlier. We advocate setting α to a value like 0.8, which is close to 1.0, to benefit from enhanced stability without a significant sacrifice in peak performance. Refer to Appendix D.1 for detailed ablation on α.

4.5 Summary of Ablation and Final Model Performance

Comparative analysis of REDI against established objectives. Table 2 summarizes the optimal outcomes from our ablation studies across key reasoning benchmarks (pass@1 over 16 samples), with all configurations initialized from the `Qwen-SFT-1.5B-3ep` model. Our REDI objective ($\alpha = 0.8$, $\mathrm{LR} = 1 \times 10^{-6}$) consistently surpasses the SFT baseline and optimized DPO, SimPO, and symmetric REDI configurations across all metrics, achieving a benchmark average of 48.3%.

Table 2: Ablation Study: Model Performance Comparison (pass@1, 16 samples). SEM reported. We chose the best checkpoint for each configuration.

Model Configuration	MATH-500	AIME24	AMC23	Minerva	OlympiadBench	Avg.
Qwen-SFT-1.5B-3ep (Start)	76.7 ±0.3	18.1 ±1.1	52.8 ±0.6	24.6 ±0.5	37.5 ±0.3	41.9 ±0.3
DPO ($\beta = 0.001, \mathrm{LR} = 2e-7$)	81.3 ±0.2	24.6 ±1.7	58.5 ±0.6	28.7 ±0.4	43.1 ±0.3	47.2 ±0.4
SimPO ($\beta = 2, \gamma = 1, \mathrm{LR} = 5e-7$)	81.1 ±0.3	24.8 ±1.9	58.8 ±0.6	29.1 ±0.2	42.2 ±0.3	47.2 ±0.4
Symmetric REDI ($\alpha = 1.0, \mathrm{LR} = 2e-7$)	**81.7 ±0.2**	25.8 ±1.3	**59.5 ±1.1**	29.3 ±0.4	42.1 ±0.2	47.7 ±0.4
REDI ($\alpha = 0.8, \mathrm{LR} = 1e-6$)	**81.7 ±0.2**	**27.3 ±1.4**	58.8 ±0.8	**30.4 ±0.5**	**43.4 ±0.3**	**48.3 ±0.3**

Advancing openly post-trained models through enhanced data efficiency. When REDI stage 2 training is applied to the stronger `Qwen-SFT-1.5B-5ep` baseline, our final `Qwen-REDI-1.5B` model ($\alpha = 0.8$, $\mathrm{LR} = 1 \times 10^{-6}$) attains state-of-the-art results as shown in Table 1. Remarkably, `Qwen-REDI-1.5B`—post-trained on merely 131k openly available data points—outperforms `DeepSeek-R1-Distill-Qwen-1.5B` (trained on 800k proprietary samples). This underscores the

exceptional data efficiency of our Reinforcement Distillation framework, achieved through systematic utilization of previously discarded negative traces.

4.6 REDI Improves Performance Without Harming Potential for Future Online RL

A key consideration is whether REDI enhances performance (like pass@1) by simply reinforcing the model's existing high-probability solution paths, or if it genuinely broadens its problem-solving abilities. Online Reinforcement Learning (RL) often works by refining and amplifying the knowledge already present within a model [Shao et al., 2024, Yue et al., 2025]. Therefore, it's important that an offline method like REDI doesn't narrow the model's underlying knowledge base.

A model's ability to find a correct answer given multiple attempts (e.g., pass@k for larger k, like $k = 16$) can serve as an indicator of the breadth of its existing knowledge. If REDI maintains or improves these pass@k scores, it suggests that while it refines certain solution strategies, it doesn't do so at the expense of the model's diverse underlying capabilities. This would mean the model remains a strong candidate for subsequent online RL.

We investigate this by examining pass@16 performance, as presented in Tables 3 and 4.

Table 3 shows that for models initialized from `Qwen-SFT-1.5B-3ep`, REDI (with $\alpha = 0.8$) not only improves pass@1 (Table 2) but also sustains or improves pass@16 scores across several benchmarks (e.g., AIME24, Minerva, OlympiadBench) compared to both the SFT baseline and other preference optimization methods. For instance, it achieved the best pass@16 on AIME24 and Minerva among the preference-tuned models.

Furthermore, Table 4 indicates that our final `Qwen-REDI-1.5B` model (initialized from the stronger `Qwen-SFT-1.5B-5ep`) maintains robust pass@16 performance. It achieves the highest pass@16 on AIME24 and matches or surpasses the SFT baseline and the `DeepSeek-R1-Distill-Qwen-1.5B` model on Minerva and OlympiadBench.

The consistent maintenance or improvement in pass@16 scores suggests that REDI's offline refinement does not merely over-optimize for a narrow set of high-probability solutions from the SFT model. Rather, these pass@16 results indicate that by learning from both the teacher's successful and unsuccessful solution attempts, REDI genuinely improves the model's overall problem-solving abilities. It appears to build these skills without causing the model to "forget" or narrow down the range of solutions it could already generate. This is encouraging, as it suggests that REDI-trained models are well-prepared, and potentially even better suited, for subsequent performance gains through online RL.

Table 3: **Pass@16** Performance Comparison for Models Initialized from `Qwen-SFT-1.5B-3ep`.

Model Configuration	MATH-500	AIME24	AMC23	Minerva	OlympiadBench
Qwen-SFT-1.5B-3ep (Start)	94.6	53.3	86.7	57.7	63.1
DPO ($\beta = 0.001, \text{LR} = 2e-7$)	94.6	63.3	86.7	58.8	**64.7**
SimPO ($\beta = 2, \gamma = 1, \text{LR} = 5e-7$)	94.8	60.0	89.2	58.5	63.3
Symmetric REDI ($\alpha = 1.0, \text{LR} = 2e-7$)	**95.0**	63.3	**90.4**	58.8	62.5
REDI ($\alpha = 0.8, \text{LR} = 1e-6$)	**95.0**	**66.7**	81.9	**59.9**	63.9

Table 4: **Pass@16** Performance for REDI Initialized from `Qwen-SFT-1.5B-5ep`.

Model	MATH-500	AIME24	AMC23	Minerva	OlympiadBench
DeepSeek-R1-Distill-Qwen-1.5B	**95.6**	63.3	**92.8**	56.6	**65.8**
Qwen-SFT-1.5B-5ep (SFT baseline)	**95.6**	56.7	86.7	56.3	64.3
Qwen-REDI-1.5B ($\alpha = 0.8, \text{LR} = 1e-6$)	95.0	**66.7**	90.4	**57.0**	**65.8**

5 Related Work

Eliciting reasoning in LLMs through Online Reinforcement Learning. Online Reinforcement Learning (RL) has emerged as a powerful paradigm for enhancing the reasoning abilities of LLMs. Pioneering works like DeepSeek-R1 [DeepSeek-AI et al., 2025] and other research efforts [Zeng et al., 2025, Pan et al., 2025] demonstrate its effectiveness in optimizing LLMs for complex reasoning tasks. These approaches typically employ an iterative online process: the model generates reasoning traces, receives feedback (e.g., from a reward model or verifier), and updates its policy accordingly.

However, online RL methods are often computationally demanding due to the slow, auto-regressive nature of response generation (rollouts). Moreover, the performance improvements from online RL are often seen as amplifying existing capabilities within the base model, rather than imparting entirely new knowledge, thus its effectiveness can be bounded by the base model's inherent potential [Yue et al., 2025]. REDI, in contrast, operates in an offline setting, focusing on efficiently leveraging pre-collected distilled data to instill reasoning abilities from a teacher model.

Enhancing reasoning in LLMs with distillation. An alternative, more cost-effective strategy for smaller, efficient student models is distillation, where reasoning traces (e.g., Chain-of-Thought) generated by powerful "teacher" models are used for training [Team, 2025, Face, 2025]. This approach has led to strong performance in openly post-trained models tuned with datasets like Open-R1 [Face, 2025, Labs, 2025, Ye et al., 2025, Muennighoff et al., 2025, Wen et al., 2025]. However, a common limitation of current distillation methodologies is their predominant reliance on rejection sampling, which only utilizes positive (correct) reasoning examples. This practice discards valuable negative examples, despite the significant computational effort invested in their generation. Our work addresses this gap by proposing a framework that effectively leverages both positive and negative distilled reasoning traces to maximize LLM reasoning performance in an offline setting.

Offline preference optimization methods in LLMs. The field of aligning LLMs with human preferences has seen significant advancements through offline preference optimization methods. Methods like DPO, SimPO, and IPO [Rafailov et al., 2024, Meng et al., 2024, Gheshlaghi Azar et al., 2024] simplifies the RLHF pipeline by directly optimizing a policy to satisfy preferences without explicit reward modeling or on-policy data collection. These methods are primarily designed for general preference alignment tasks. More recently, some of these techniques have been adapted for reasoning tasks, often in conjunction with online data collection or iterative refinement, such as in Iterative Reasoning Preference Optimization [Pang et al., 2024], Online-DPO-R1 [Zhang et al.] and Light-R1 [Wen et al., 2025]. Our work distinguishes itself by focusing on the *offline* utilization of positive and negative reasoning traces, specifically for distillation, and by proposing a novel, reference-free objective tailored to this context.

Learning dynamics and stability in LLM post-training. The training dynamics of LLMs, particularly when subjected to reinforcement learning or preference optimization, are complex and can be prone to instability. Phenomena like "training collapse," characterized by a rapid decrease in model performance and degenerate outputs, have been observed in methods like DPO, especially when using aggressive learning rates or insufficient regularization [Yan et al., 2025, Razin et al., 2025, Ren and Sutherland, 2025]. This instability is often attributed to unintended side effects of off-policy gradients, where penalizing negative responses might inadvertently suppress semantically similar positive responses. Heuristic mitigations, such as auxiliary SFT losses or asymmetric β tuning, have been explored to address these issues [Pang et al., 2024, Yan et al., 2025]. Inspired by these discussions, our proposed Reinforcement Distillation (REDI) framework tackles this challenge by introducing an asymmetrically weighted, reference-free objective. This design choice is also motivated by the observed trade-off between stability and peak performance in existing methods, aiming to achieve both by carefully controlling the influence of negative gradients.

6 Conclusion

This paper introduced Reinforcement Distillation (REDI), a two-stage offline post-training framework designed to maximize the reasoning performance of LLMs by effectively utilizing both positive and negative distilled reasoning traces. Addressing the underutilization of negative examples in standard rejection sampling, REDI first builds a strong foundation via SFT on positive traces, then refines the model using our novel, asymmetrically weighted, reference-free REDI objective. This objective was motivated by our analysis of the performance-stability trade-off inherent in the regularization terms of established methods like DPO and SimPO.

Our empirical evaluations demonstrate REDI's superiority over SFT-only approaches and SFT combined with DPO/SimPO. Notably, the Qwen-REDI-1.5B model, trained on a modest 131k open examples, achieved state-of-the-art performance (83.1% on MATH-500 pass@1) for 1.5B models post-trained offline with openly available data, matching or surpassing models trained on significantly larger, closed datasets. Furthermore, REDI enhances model capabilities without diminishing potential for future online RL. By transforming previously discarded negative traces into valuable learning signals, REDI offers a more data-efficient and effective pathway for distilling complex reasoning abilities into smaller student models.

7 Acknowledgment

This work was partially supported by the National Natural Science Foundation of China (82394432 & 92249302). We thank Chao Qu and Haozhe Wang for their insightful discussions.

References

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu 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, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL `https://arxiv.org/abs/2501.12948`.

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

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. A general theoretical paradigm to understand learning from human preferences. In Sanjoy Dasgupta, Stephan Mandt, and Yingzhen Li, editors, *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 4447–4455. PMLR, 02–04 May 2024. URL `https://proceedings.mlr.press/v238/gheshlaghi-azar24a.html`.

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

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. OlympiadBench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3828–3850, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.211. URL `https://aclanthology.org/2024.acl-long.211/`.

Hynek Kydlíček. Math-Verify: Math Verification Library. URL `https://github.com/huggingface/math-verify`.

Bespoke Labs. Bespoke-stratos: The unreasonable effectiveness of reasoning distillation. https://www.bespokelabs.ai/blog/bespoke-stratos-the-unreasonable-effectiveness-of-reasoning-distillation, 2025. Accessed: 2025-01-22.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models, 2022. URL `https://arxiv.org/abs/2206.14858`.

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

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL `https://arxiv.org/abs/1711.05101`.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl, 2025. Notion Blog.

Yu Meng, Mengzhou Xia, and Danqi Chen. SimPO: Simple preference optimization with a reference-free reward. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL `https://openreview.net/forum?id=3Tzcot1LKb`.

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

OpenAI, :, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett, Ananya Kumar, Andre Saraiva, Andrea Vallone, Andrew Duberstein, Andrew Kondrich, Andrey Mishchenko, Andy Applebaum, Angela Jiang, Ashvin Nair, Barret Zoph, Behrooz Ghorbani, Ben Rossen, Benjamin Sokolowsky, Boaz Barak, Bob McGrew, Borys Minaiev, Botao Hao, Bowen Baker, Brandon Houghton, Brandon McKinzie, Brydon Eastman, Camillo Lugaresi, Cary Bassin, Cary Hudson, Chak Ming Li, Charles de Bourcy, Chelsea Voss, Chen Shen, Chong Zhang, Chris Koch, Chris Orsinger, Christopher Hesse, Claudia Fischer, Clive Chan, Dan Roberts, Daniel Kappler, Daniel Levy, Daniel Selsam, David Dohan, David Farhi, David Mely, David Robinson, Dimitris Tsipras, Doug Li, Dragos Oprica, Eben Freeman, Eddie Zhang, Edmund Wong, Elizabeth Proehl, Enoch Cheung, Eric Mitchell, Eric Wallace, Erik Ritter, Evan Mays, Fan Wang, Felipe Petroski Such, Filippo Raso, Florencia Leoni, Foivos Tsimpourlas, Francis Song, Fred von Lohmann, Freddie Sulit, Geoff Salmon, Giambattista Parascandolo, Gildas Chabot, Grace Zhao, Greg Brockman, Guillaume Leclerc, Hadi Salman, Haiming Bao, Hao Sheng, Hart Andrin, Hessam Bagherinezhad, Hongyu Ren, Hunter Lightman, Hyung Won Chung, Ian Kivlichan, Ian O'Connell, Ian Osband, Ignasi Clavera Gilaberte, Ilge Akkaya, Ilya Kostrikov, Ilya Sutskever, Irina Kofman, Jakub Pachocki, James Lennon, Jason Wei, Jean Harb, Jerry Twore, Jiacheng Feng, Jiahui Yu, Jiayi Weng, Jie Tang, Jieqi Yu, Joaquin Quiñonero Candela, Joe Palermo, Joel Parish, Johannes Heidecke, John Hallman, John Rizzo, Jonathan Gordon, Jonathan Uesato, Jonathan Ward, Joost Huizinga, Julie Wang, Kai Chen, Kai Xiao, Karan Singhal, Karina Nguyen, Karl Cobbe, Katy Shi, Kayla Wood, Kendra Rimbach, Keren Gu-Lemberg, Kevin Liu, Kevin Lu, Kevin Stone, Kevin Yu, Lama Ahmad, Lauren Yang, Leo Liu, Leon Maksin, Leyton Ho, Liam Fedus, Lilian Weng, Linden Li, Lindsay McCallum, Lindsey Held, Lorenz Kuhn, Lukas Kondraciuk, Lukasz Kaiser, Luke Metz, Madelaine Boyd, Maja Trebacz, Manas Joglekar, Mark Chen, Marko Tintor, Mason Meyer, Matt Jones, Matt Kaufer, Max Schwarzer, Meghan Shah, Mehmet Yatbaz, Melody Y. Guan, Mengyuan Xu, Mengyuan Yan, Mia Glaese, Mianna Chen, Michael Lampe, Michael Malek, Michele Wang, Michelle Fradin, Mike McClay, Mikhail Pavlov, Miles Wang, Mingxuan Wang, Mira Murati, Mo Bavarian, Mostafa Rohaninejad, Nat McAleese, Neil Chowdhury, Neil Chowdhury, Nick Ryder, Nikolas Tezak, Noam Brown, Ofir Nachum, Oleg Boiko, Oleg Murk, Olivia Watkins, Patrick Chao, Paul Ashbourne, Pavel Izmailov, Peter Zhokhov, Rachel Dias, Rahul Arora, Randall Lin, Rapha Gontijo Lopes, Raz Gaon, Reah Miyara, Reimar Leike, Renny Hwang, Rhythm Garg, Robin Brown, Roshan James, Rui Shu, Ryan Cheu, Ryan Greene, Saachi

Jain, Sam Altman, Sam Toizer, Sam Toyer, Samuel Miserendino, Sandhini Agarwal, Santiago Hernandez, Sasha Baker, Scott McKinney, Scottie Yan, Shengjia Zhao, Shengli Hu, Shibani Santurkar, Shraman Ray Chaudhuri, Shuyuan Zhang, Siyuan Fu, Spencer Papay, Steph Lin, Suchir Balaji, Suvansh Sanjeev, Szymon Sidor, Tal Broda, Aidan Clark, Tao Wang, Taylor Gordon, Ted Sanders, Tejal Patwardhan, Thibault Sottiaux, Thomas Degry, Thomas Dimson, Tianhao Zheng, Timur Garipov, Tom Stasi, Trapit Bansal, Trevor Creech, Troy Peterson, Tyna Eloundou, Valerie Qi, Vineet Kosaraju, Vinnie Monaco, Vitchyr Pong, Vlad Fomenko, Weiyi Zheng, Wenda Zhou, Wes McCabe, Wojciech Zaremba, Yann Dubois, Yinghai Lu, Yining Chen, Young Cha, Yu Bai, Yuchen He, Yuchen Zhang, Yunyun Wang, Zheng Shao, and Zhuohan Li. Openai o1 system card, 2024. URL https://arxiv.org/abs/2412.16720.

Jiayi Pan, Junjie Zhang, Xingyao Wang, Lifan Yuan, Hao Peng, and Alane Suhr. Tinyzero. https://github.com/Jiayi-Pan/TinyZero, 2025. Accessed: 2025-01-24.

Richard Yuanzhe Pang, Weizhe Yuan, He He, Kyunghyun Cho, Sainbayar Sukhbaatar, and Jason E Weston. Iterative reasoning preference optimization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=4XIKfvNYvx.

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.

Noam Razin, Sadhika Malladi, Adithya Bhaskar, Danqi Chen, Sanjeev Arora, and Boris Hanin. Unintentional unalignment: Likelihood displacement in direct preference optimization. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=uaMSBJDnRv.

Yi Ren and Danica J. Sutherland. Learning dynamics of LLM finetuning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=tPNHOoZFl9.

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.

OpenThoughts Team. Open Thoughts. https://open-thoughts.ai, January 2025.

Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Lifu Tang, Xiaowei Lv, Haosheng Zou, Yongchao Deng, Shousheng Jia, and Xiangzheng Zhang. Light-r1: Curriculum sft, dpo and rl for long cot from scratch and beyond, 2025. URL https://arxiv.org/abs/2503.10460.

Yuzi Yan, Yibo Miao, Jialian Li, Yipin Zhang, Jian Xie, Zhijie Deng, and Dong Yan. 3d-properties: Identifying challenges in dpo and charting a path forward, 2025. URL https://arxiv.org/abs/2406.07327.

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

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025.

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

Hanning Zhang, Jiarui Yao, Chenlu Ye, Wei Xiong, and Tong Zhang. Online-dpo-r1: Unlocking effective reasoning without the ppo overhead, 2025. *Notion Blog*.

A Convergence Guarantee

Definition A.1 (REDI Loss)**.** *Let π_θ be the target policy model, which is parametrized by $\theta \in \mathbb{R}^d$. Let y_w be the preferred data and y_l be the not preferred data. Let N be the number of data pairs, i.e., the number of (y_w, y_l). Let α be a preset hyperparameter. The loss function is given as:*

$$\mathcal{L}(\theta) := \mathop{\mathbb{E}}_{(x,y_w,y_l)\sim\mathcal{D}_{\mathit{Pref}}}\left[-\frac{\log \pi_\theta(y_w|x)}{|y_w|} + \alpha \frac{\log \pi_\theta(y_l|x)}{|y_l|}\right]$$

Assumption A.2. *Let the loss function $\mathcal{L}(\theta)$ be defined in Definition A.1. We assume that $\nabla\mathcal{L}(\theta)$ is L-Lipschitz, i.e., for $\theta, \widehat{\theta} \in \mathbb{R}^d$, we have*

$$\|\nabla\mathcal{L}(\theta) - \nabla\mathcal{L}(\widehat{\theta})\| \leq L \cdot \|\theta - \widehat{\theta}\|.$$

Definition A.3 (Update Rule with Linear Scheme)**.** *Let the loss function $\mathcal{L}(\theta)$ be defined in Definition A.1. At training step k, for the parameter $\theta \in \mathbb{R}^d$, we have:*

$$\theta^k = \theta^{k-1} - \eta_{k-1} \cdot \nabla\mathcal{L}(\theta^{k-1}).$$

Our Training scheme involve a linear warm-up stage and a linear decay stage. The learning rate η is given by

$$\eta_k = \begin{cases} \underline{\eta} + \frac{k}{\widetilde{k}}(\overline{\eta} - \underline{\eta}), & \text{if } k \leq \widetilde{k}; \\ \overline{\eta} - \frac{k-\widetilde{k}}{K-\widetilde{k}}(\overline{\eta} - \underline{\eta}), & \text{if } k > \widetilde{k}, \end{cases}$$

where $\widetilde{k} \in [K]$ is a preset hyperparameter denoting the number of warm-up steps, $\underline{\eta}$ and $\overline{\eta}$ are the minimum value and maximum value of learning rate η respectively, i.e., $\eta \in [\underline{\eta}, \overline{\eta}]$. Specifically, we set $\overline{\eta} < 1/L$, where L is the Lipschitz constant in Assumption A.2.

Theorem A.4 (Convergence Guarantee)**.** *Let the loss function $\mathcal{L}(\theta)$ be defined in Definition A.1. For any small $\epsilon > 0$, the update iterations satisfy:*

$$\min_{k\in[K]} \mathbb{E}[\|\nabla\mathcal{L}(\theta^k)\|^2] \leq \epsilon.$$

Proof. At time step $k-1$, we analyze the expected loss and perform a Taylor expansion of $\mathcal{L}(\theta)$:

$$\begin{aligned} \mathbb{E}[\mathcal{L}(\theta^k)] \leq&\ \mathbb{E}[\mathcal{L}(\theta^{k-1}) + (\theta^k - \theta^{k-1})^\top \nabla\mathcal{L}(\theta^{k-1}) + 0.5L\|\theta^k - \theta^{k-1}\|^2] \\ \leq&\ \mathbb{E}[\mathcal{L}(\theta^{k-1}) - \eta_{k-1}\nabla\mathcal{L}(\theta^{k-1})^\top \nabla\mathcal{L}(\theta^{k-1}) + 0.5L\|\eta_{k-1}\nabla\mathcal{L}(\theta^{k-1})\|^2] \\ =&\ \mathbb{E}[\mathcal{L}(\theta^{k-1})] - \eta_{k-1}\,\mathbb{E}[\|\nabla\mathcal{L}(\theta^{k-1})\|^2] + 0.5L\eta_{k-1}^2 \cdot \mathbb{E}[\|\nabla\mathcal{L}(\theta^{k-1})\|^2], \end{aligned}$$

where the first step follows from Assumption A.2, the second step follows from Definition A.3, and the third step follows from basic algebra.

Thus, we can show that

$$\mathbb{E}[\|\nabla\mathcal{L}(\theta^{k-1})\|^2] \leq \frac{1}{\eta(1 - 0.5L\eta)}\,\mathbb{E}[\mathcal{L}(\theta^{k-1}) - \mathcal{L}(\theta^k)], \tag{6}$$

which follows from Definition A.3 and basic algebra.

Further, for the minimal value of $\mathbb{E}[\|\nabla\mathcal{L}(\theta^{k-1})\|^2]$, we have

$$\begin{aligned} \min_{k\in[K]} \mathbb{E}[\|\nabla\mathcal{L}(\theta^{k-1})\|^2] \leq&\ \frac{1}{K}\sum_{k=1}^{K} \mathbb{E}[\|\nabla\mathcal{L}(\theta^{k-1})\|^2] \\ \leq&\ \frac{1}{K}\sum_{k=1}^{K}\left(\frac{1}{\eta_{k-1}(1 - 0.5L\eta_{k-1})}\,\mathbb{E}[\mathcal{L}(\theta^{k-1}) - \mathcal{L}(\theta^k)]\right) \\ \leq&\ \frac{1}{K\underline{\eta}(1 - 0.5L\overline{\eta})}(\mathcal{L}(\theta^0) - \mathcal{L}(\theta^K)) \\ \leq&\ \frac{1}{K\underline{\eta}(1 - 0.5L\overline{\eta})}(\mathcal{L}(\theta^0) - \mathcal{L}(\theta^*)) \end{aligned}$$

where the first step follows from the minimum is always smaller than the average, the second step follows from Eq. (6), the third step follows from Definition A.3 and basic algebra, the fourth step follows from $\mathcal{L}(\theta^*) \leq \mathcal{L}(\theta^K)$.

Plugging in

$$K = \frac{\mathcal{L}(\theta^0) - \mathcal{L}(\theta^*)}{\underline{\eta}(1 - 0.5L\overline{\eta})\epsilon},$$

we finish the proof. $\square$

B Relationship between SimPO and Our Loss Function

First, we restate SimPO loss Eq. (3) as follows:

$$\mathcal{L}_{\text{SimPO}}(\theta) = - \mathop{\mathbb{E}}_{(x,y_w,y_l)\sim\mathcal{D}_{\text{Pref}}} \left[\log \sigma \left(\beta \left(\frac{\log \pi_\theta(y_w|x)}{|y_w|} - \frac{\log \pi_\theta(y_l|x)}{|y_l|}\right) - \gamma\right)\right]$$

Then, we restate the gradient of SimPO, which is implicit on page 22 in Meng et al. [2024].

$$\begin{aligned}\nabla_\theta \mathcal{L}_{\text{SimPO}}(\theta) =& -\beta \mathop{\mathbb{E}}_{(x,y_w,t_l)\sim\mathcal{D}_{\mathrm{Pref}}}[\sigma(\frac{\beta}{|y_l|}\log(y_l \mid x) - \frac{\beta}{|y_w|}\log(y_w \mid x) + \gamma)\cdot \\ & (\frac{1}{|y_w|}\nabla_\theta \log(y_w \mid x) - \frac{1}{|y_l|}\nabla_\theta \log(y_l \mid x))].\end{aligned}$$

Define $R_\theta := \frac{1}{|y_w|}\log(y_w \mid x) - \frac{1}{|y_l|}\log(y_l \mid x)$, we have

$$\begin{aligned}\nabla_\theta \mathcal{L}_{\text{SimPO}}(\theta) =& -\beta \mathop{\mathbb{E}}_{(x,y_w,t_l)\sim\mathcal{D}_{\mathrm{Pref}}}[\sigma(-\beta\cdot R_\theta + \gamma)\cdot \nabla_\theta R_\theta] \\ =& - \mathop{\mathbb{E}}_{(x,y_w,t_l)\sim\mathcal{D}_{\mathrm{Pref}}}[\sigma(-\beta\cdot R_\theta + \gamma)\cdot \beta\nabla_\theta R_\theta].\end{aligned}$$

Also, we define

$$\mathcal{L}_{\text{symm}}(\theta) = \mathop{\mathbb{E}}_{(x,y_w,y_l)\sim\mathcal{D}_{\text{Pref}}} \left(-\frac{\log \pi_\theta(y_w|x)}{|y_w|} + \frac{\log \pi_\theta(y_l|x)}{|y_l|}\right).$$

Fact B.1. *the sigmoid function is Lipschitz continuous, i.e.,* $|\sigma(x) - \sigma(x')| \le 0.25|x - x'|$.

Lemma B.2. *Let R_θ be bounded by constant c_0. Let the hyperparameter $\beta > 0$ be an arbitrary small number. Let γ be a constant. Let $\epsilon = \frac{\beta\cdot c_0}{4}$ which can be arbitary small, we have*

$$|\sigma(-\beta R_\theta + \gamma) - \sigma(\gamma)| < \epsilon.$$

Proof. We can show

$$\begin{aligned}|\sigma(-\beta R + \gamma) - \sigma(\gamma)| \le& \frac{\beta\cdot R_\theta}{4} \\ \le& \frac{\beta\cdot c_0}{4},\end{aligned}$$

where the first step follows from Fact B.1, the second step follows from R_θ is bounded by constant c_0. □

Given typical learning rate η, adjusted learning rate η'. We claim the one step update over parameter θ with loss function $\mathcal{L}_{\text{SimPO}}(\theta)$ is approximately equal to $\mathcal{L}_{\text{symm}}(\theta)$.

Proposition B.3. *Let the hyperparameter $\beta > 0$ be an arbitrary small number. Let η_{SimPO} be set as an inverse multiple of β, i.e., $\eta_{\mathrm{SimPO}} = c_1/\beta$. Assume $\nabla_\theta R_\theta$ is bounded. Given the initial parameter θ_t. We can choose learning rate $\eta = c_1\cdot\sigma(\gamma)$, such that for the one step gradient decent update over $\Delta_\theta^{\mathrm{SimPO}} := \theta_{t+1} - \theta_t = -\eta_{\mathrm{SimPO}}\nabla_\theta\mathcal{L}_{\textit{SimPO}}(\theta)$ is approximately equal to the gradient decent update over $\Delta_\theta^{\mathrm{symm}} := \theta_{t+1} - \theta_t = -\eta\nabla_\theta\mathcal{L}_{\textit{symm}}(\theta)$, i.e., $|\eta_{\mathrm{SimPO}}\nabla_\theta\mathcal{L}_{\textit{SimPO}}(\theta) - \eta\nabla_\theta\mathcal{L}_{\textit{symm}}(\theta)|$ can be arbitrary small.*

Proof. We have

$$\begin{aligned}|\eta_{\mathrm{SimPO}}\nabla_\theta\mathcal{L}_{\text{SimPO}}(\theta) - \eta\nabla_\theta\mathcal{L}_{\text{symm}}(\theta)| =& |\eta_{\mathrm{SimPO}}\sigma(-\beta\cdot R_\theta+\gamma)\cdot\beta\nabla_\theta R_\theta - \eta\nabla_\theta R_\theta| \\ =& |c_1\sigma(-\beta\cdot R_\theta+\gamma)\nabla_\theta R_\theta - \eta\nabla_\theta R_\theta| \\ =& |c_1\sigma(-\beta\cdot R_\theta+\gamma)\nabla_\theta R_\theta - c_1\sigma(\gamma)\nabla_\theta R_\theta| \\ =& c_1\nabla_\theta R_\theta|\sigma(-\beta R+\gamma) - \sigma(\gamma)| \\ \le& c_0 c_1\nabla_\theta R_\theta\beta/4,\end{aligned}$$

which can be arbitrarily small. □

C Detailed Experimental Setup

This section provides a comprehensive overview of the experimental setup, including details on the base model, dataset curation, training configurations for both SFT and preference optimization stages, evaluation protocols, and computational resources. For more detailed implementation, readers may refer to the provided codebase.

C.1 Base Model and Initial SFT Checkpoints

All experiments commenced with the `Qwen2.5-Math-1.5B` model as the base LLM, chosen for its strong foundational capabilities in mathematical reasoning. Two SFT checkpoints were prepared from this base model to serve different purposes:

- `Qwen-SFT-1.5B-3ep`: This model was fine-tuned on the $\mathcal{D}_{\text{SFT}}$ dataset for 3 epochs. It served as the starting point for the ablation studies involving DPO, SimPO, and REDI, as detailed in Section 4 and Table 2.
- `Qwen-SFT-1.5B-5ep`: Fine-tuned for 5 epochs on $\mathcal{D}_{\text{SFT}}$, this model demonstrated improved SFT performance and was used as the SFT starting point for our final, best-performing `Qwen-REDI-1.5B` model presented in Table 1.

C.2 Datasets

As described in Section 3.1, the data was derived from the OpenR1-Math-Raw corpus [Face, 2025], excluding the `cn_k12` subset due to its lower relative difficulty. A response was considered correct if both the Llama judge (an LLM-based verifier) and Math-Verify [Kydlíček] (a rule-based verifier) labeled it as "True"; otherwise, it was considered incorrect.

- $\mathcal{D}_{\text{SFT}}$ **(Positive Traces Dataset):** Contained 77,629 problem-solution pairs (x, y_w) where y_w is a correct reasoning trace. This dataset was used for Stage 1 SFT.
- $\mathcal{D}_{\text{Pref}}$ **(Preference Pairs Dataset):** Consisted of 53,175 triplets (x, y_w, y_l). This dataset was derived by selecting data from $\mathcal{D}_{\text{SFT}}$ for which an incorrect response y_l (deemed incorrect by either Math-Verify or the Llama verifier) was also available for the same problem x. Each triplet comprises a problem x, a preferred correct trace y_w, and a rejected incorrect trace y_l. We further filtered out instances where queries exceeded 800 tokens, or either chosen (y_w) or rejected (y_l) responses exceeded 19,000 tokens. This dataset was used for Stage 2 preference optimization.

C.3 Stage 1: Supervised Fine-Tuning (SFT)

- **Objective:** Maximize log-likelihood of positive traces (Eq. (1)).
- **Optimizer:** AdamW [Loshchilov and Hutter, 2019] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and a weight decay of 0.0001.
- **Learning Rate Schedule:** Peak learning rate of 5×10^{-5}, with a linear warmup for the first 10% of total training steps, followed by a linear decay to zero.
- **Batch Size:** 128.
- **Epochs:** 3 epochs for `Qwen-SFT-1.5B-3ep` and 5 epochs for `Qwen-SFT-1.5B-5ep`.
- **Max Sequence Length:** 32,768 tokens.

C.4 Stage 2: Preference Optimization

All preference optimization methods (DPO, SimPO, REDI) were initialized from an SFT checkpoint (`Qwen-SFT-1.5B-3ep` for ablations, `Qwen-SFT-1.5B-5ep` for the final model). Training was conducted on the $\mathcal{D}_{\text{Pref}}$ dataset.

- **Optimizer:** AdamW with the same parameters as in Stage 1 ($\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$, weight decay 0.0001).
- **Learning Rate Schedule:** Linear warmup for the first 10% of total training steps, followed by linear decay to zero. Peak learning rates were method-specific and tuned as described below.
- **Batch Size:** 32.

- **Epochs:** 1 epoch over $\mathcal{D}_{\text{Pref}}$.
- **Max Sequence Length:** 800 tokens for queries (prompts x) and 19,000 tokens for responses (y_w, y_l).

Hyperparameter Configurations and Tuning: We meticulously tuned hyperparameters for each preference optimization method. The reference model for DPO was the `Qwen-SFT-1.5B-3ep` checkpoint.

- **DPO [Rafailov et al., 2024]:**
  - **β values explored:** $\{0.001, 0.01, 0.1\}$.
  - **Learning Rate (LR) exploration:** Specific LRs were tested for each β value:
    * For $\beta = 0.1$: $\{1 \times 10^{-6}, 1 \times 10^{-7}, 1 \times 10^{-8}\}$.
    * For $\beta = 0.01$: $\{1 \times 10^{-6}, 5 \times 10^{-7}, 2 \times 10^{-7}, 1 \times 10^{-7}\}$.
    * For $\beta = 0.001$: $\{1 \times 10^{-6}, 5 \times 10^{-7}, 2 \times 10^{-7}, 1 \times 10^{-7}\}$.
  - **Best Ablation Configuration (Table 2):** $\beta = 0.001$, LR $= 2 \times 10^{-7}$.
- **SimPO [Meng et al., 2024]:**
  - Two primary configurations were evaluated based on different β values, with the margin γ set to maintain a γ/β ratio of 0.5:
    * Configuration 1: $\beta = 2, \gamma = 1, \text{LR} = 5 \times 10^{-7}$.
    * Configuration 2: $\beta = 10, \gamma = 5, \text{LR} = 3 \times 10^{-7}$.
  - **Best Ablation Configuration (Table 2):** The first configuration, $\beta = 2, \gamma = 1, \text{LR} = 5 \times 10^{-7}$, yielded superior results in our ablation studies.
- **REDI (Ours):**
  - **α (asymmetric weight) values explored:** $\{0.2, 0.5, 0.8, 1.0\}$. (See Appendix D for detailed α tuning).
  - **Learning Rate (LR) exploration:**
    * For $\alpha \in \{0.2, 0.8\}$, a learning rate of 1×10^{-6} was primarily used.
    * For $\alpha = 0.5$, learning rates of 1×10^{-6} and 2×10^{-6} were tested.
    * For $\alpha = 1.0$ (Symmetric REDI), learning rates of $\{1 \times 10^{-6}, 2 \times 10^{-7}, 1 \times 10^{-7}\}$ were evaluated.
  - **Best Ablation Configuration (Table 2):** $\alpha = 0.8, \text{LR} = 1 \times 10^{-6}$.
  - **Final Model Configuration (Table 1):** For the `Qwen-REDI-1.5B` model, initialized from `Qwen-SFT-1.5B-5ep`, we used $\alpha = 0.8$ with LR $= 1 \times 10^{-6}$.

C.5 Evaluation

Decoding Parameters: During all evaluations, generated samples were decoded using the following parameters:

- Temperature: 0.6
- Top P (nucleus sampling): $p = 0.95$
- Maximum generation length: $32,768$ tokens

Evaluation Frameworks, Protocols, and Benchmarks: We utilized two distinct configurations for evaluating model performance:

- *Intermediate Evaluations:* These evaluations were used for hyperparameter tuning, generating performance plots (e.g., Figures 3, 4, 5, 6), and ablation studies. They were conducted using LightEval [Habib et al., 2023] on the MATH-500 benchmark. Performance was measured as pass@1, averaged over 4 generated samples per problem.
- *Final Model Evaluations:* These evaluations, presented in main comparison tables (Table 1,2,3,4), were performed using the DeepScaleR/rllm [Luo et al., 2025] framework. The benchmarks included MATH-500, AIME24, AMC23, Minerva, and OlympiadBench. Performance was measured as either pass@1 (averaged over 16 generated samples) or pass@16. For these evaluations, and specifically for our models, we fixed "<think>" as the first token generated by our model to align our practices with DeepSeek-R1 series of models.

SEM Calculation: The reported Standard Error of the Mean (SEM) quantifies the uncertainty of this "pass@1 over k samples" score. It is calculated as $s/\sqrt{k}$. To obtain s, we first compute k

distinct "benchmark-wide pass@1 scores." Each of these k scores (P_j, for $j = 1 \ldots k$) is determined by evaluating the model's performance across the entire benchmark using *only the j-th generated sample* for every problem. The term s is then the standard deviation of these k intermediate scores ($P_1, P_2, \ldots, P_k$). This method estimates the variability of the overall "pass@1 over k samples" metric by assessing performance consistency across the individual samples drawn for each problem.

Evaluation Prompt Format: For prompting, we followed the Open-R1 project Face [2025] and used the following template:

Prompt Template

```
Solve the following math problem efficiently and clearly.  The last line of your response should be
of the following format:  'Therefore, the final answer is:  $\boxed{{ANSWER}}$.  I hope it is correct'
(without quotes) where ANSWER is just the final number or expression that solves the problem.  Think
step by step before answering.

{Question}
```

C.6 Computational Resources

All model training and fine-tuning experiments were conducted on a distributed training cluster equipped with NVIDIA A100 80GB SXM GPUs. Each experiment was run on a node of 8 such GPUs. We utilized standard open-source libraries for large language model training, including Hugging Face Transformers for model architecture and tokenization, and Accelerate for distributed training management. DeepSpeed (ZeRO Stage 3 for DPO; ZeRO Stage 2 for SimPO and REDI) was employed to optimize memory usage and enable efficient training. Custom scripts were developed for data processing; the computational resources required for preprocessing were negligible.

Training Times: The approximate training times per run on an 8-GPU (A100 80GB) node are summarized in Table 5.

Table 5: Approximate Training Times per Run (on one 8-GPU A100 80GB node).

Task / Method	Duration per Run	Approx. GPU Hours
SFT (3 epochs on $\mathcal{D}_{\text{SFT}}$)	8 hours	64
SFT (5 epochs on $\mathcal{D}_{\text{SFT}}$)	13 hours	104
DPO (1 epoch on $\mathcal{D}_{\text{Pref}}$)	5 hours 10 mins	41.3
SimPO (1 epoch on $\mathcal{D}_{\text{Pref}}$)	4 hours 50 mins	38.7
REDI (1 epoch on $\mathcal{D}_{\text{Pref}}$)	4 hours 10 mins	33.3

The total training compute required for our final `Qwen-REDI-1.5B` model (SFT 5 epochs + REDI 1 epoch) is approximately 17 hours on an 8-GPU node (or around 136 A100 80GB GPU hours).

Evaluation Times: The approximate evaluation times on an 8-GPU (A100 80GB) node are:

- MATH-500 (pass@1 over 4 samples, LightEval): 40 minutes.
- 5 Benchmarks (pass@1 over 16 samples, DeepScaleR/rllm): 20 hours.

Total Compute for Reproducibility: The total compute needed to reproduce all results presented in this paper (including all SFT runs, hyperparameter sweeps for DPO, SimPO, and REDI, final evaluations and with buffer for debugging) is estimated to be around 350 hours on an 8-GPU node, which translates to approximately 2,800 A100 80GB GPU hours.

D Additional Results

D.1 Ablation Study on REDI Hyperparameter α

This section details the ablation study conducted to determine an effective value for the asymmetric weighting hyperparameter α in the REDI objective (Equation 5). The goal was to find an α that optimally leverages negative traces for performance improvement while maintaining training stability. We explored $\alpha \in \{0.2, 0.5, 0.8\}$ (with $\alpha = 1.0$ representing the symmetric case discussed in the main paper), using the `Qwen-SFT-1.5B-3ep` model as the starting checkpoint. Figure 7 illustrates the training dynamics for key configurations.

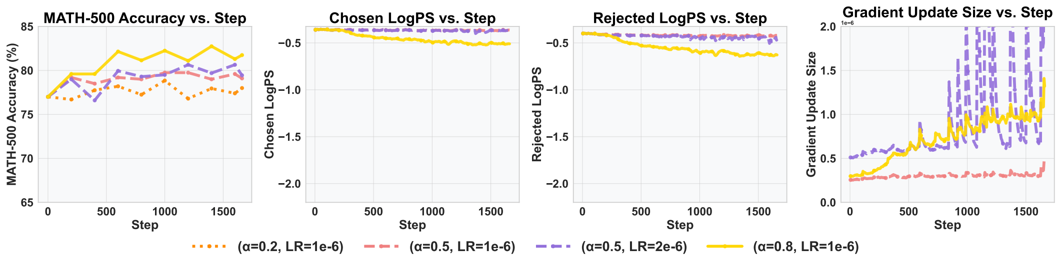

Figure 7: REDI training dynamics with varying α values and learning rates. All runs start from `Qwen-SFT-1.5B-3ep`. Metrics shown are MATH-500 Accuracy, Chosen LogPS, Rejected LogPS, and Gradient Update Size, all plotted against training steps.

Analysis of Different α Configurations:

- **Low α ($\alpha = 0.2$, LR $= 1 \times 10^{-6}$, orange dotted line):** With this configuration, the model's performance largely fluctuated around the initial SFT level. This is anticipated, as a lower α value makes the REDI objective more closely resemble the SFT loss, for which performance had already neared a plateau.
- **Moderate α ($\alpha = 0.5$, LR $= 1 \times 10^{-6}$, pink dashed line):** Increasing α to 0.5 while maintaining LR $= 1 \times 10^{-6}$ yielded improved peak accuracy (approximately 79.8%) compared to $\alpha = 0.2$. This underscores the benefit of incorporating negative samples, even moderately, over relying solely on SFT.
- **Moderate α with Higher LR** ($\alpha = 0.5$, **LR** $= 2 \times 10^{-6}$**, purple dashed line):** Testing $\alpha = 0.5$ with a more aggressive learning rate (LR $= 2 \times 10^{-6}$) showed stable LogPS values, though accompanied by intermittent spikes in the gradient update size. While its peak performance slightly surpassed the lower LR variant with $\alpha = 0.5$, it remained inferior to the $\alpha = 0.8$ run.
- **Higher** α ($\alpha = 0.8$, **LR** $= 1 \times 10^{-6}$**, yellow solid line):** This configuration achieved the highest peak MATH-500 accuracy. We note that both the Chosen LogPS and Rejected LogPS steadily decreased throughout training. This concurrent decrease, in the absence of a sudden collapse and while performance is improving, appears to be benign. It is distinct from a catastrophic collapse where both LogPS would plummet sharply alongside performance.

Importance of Update Direction over Raw Magnitude: The comparison between the ($\alpha = 0.5, \mathrm{LR} = 2 \times 10^{-6}$) and ($\alpha = 0.8, \mathrm{LR} = 1 \times 10^{-6}$) configurations is particularly insightful. The former features larger average gradient update sizes, implying a stronger raw *magnitude* of adjustment driven by the negative log-likelihood term (which scales with $\alpha \times \mathrm{LR}$). However, this did not translate to superior peak performance relative to the ($\alpha = 0.8, \mathrm{LR} = 1 \times 10^{-6}$) run.

This observation supports the view that the *direction* of the gradient, as modulated by α, is more critical than its sheer magnitude derived from negative samples. A higher α (such as 0.8) appears to provide a more qualitatively beneficial gradient signal, guiding the model more effectively. Simply increasing the learning rate for a lower α (e.g., $\alpha = 0.5$) to match or exceed the raw gradient magnitude of a higher α configuration does not necessarily yield better performance and may even compromise stability. The role of α thus extends beyond scaling the penalty; it is crucial for appropriately balancing the influence of negative examples to effectively shape the learning landscape.

Suggestions on α Tuning: Based on these ablation studies, $\alpha = 0.8$ combined with a learning rate of 1×10^{-6} demonstrated the most favorable trade-off for the `Qwen-SFT-1.5B-3ep` checkpoint, achieving the highest peak performance while maintaining robust training stability. Consequently, for applying the REDI framework to other domains or datasets, we recommend initially fixing $\alpha = 0.8$ and primarily focusing on tuning the learning rate.

E Qualitative Analysis of Model Behavior

E.1 Generation Statistics

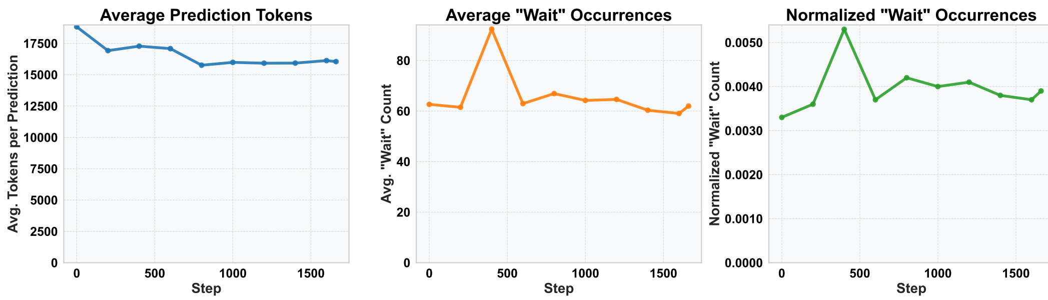

Figure 8: Generation statistics for the `REDI` model presented in Table 2 (initialized from `Qwen-SFT-1.5B-3ep`, trained with REDI using $\alpha = 0.8, \text{LR} = 1 \times 10^{-6}$) on the AIME24 test set. Metrics shown are (Left) Average Prediction Tokens, (Center) Average "Wait" Occurrences, and (Right) Normalized "Wait" Occurrences, all plotted against REDI training steps. Step 0 represents the SFT model before REDI training.

To investigate qualitative changes in reasoning style during REDI training, we monitored key generation statistics. The frequency of terms like "Wait" serves as an indicator of explicit reflective steps within the model's Chain-of-Thought (CoT), a common trait in reasoning models. The average generation length (token count per prediction) is also crucial, as complex reasoning often correlates with longer outputs. These metrics help assess how REDI influences the model's reasoning trace characteristics.

Figure 8 illustrates these trends for the REDI configuration detailed in Table 2. The average token count (left panel) shows a slight decrease from the SFT model's baseline (Step 0), stabilizing at a somewhat lower level during REDI training.

The frequency of "Wait" occurrences (center panel) exhibits a notable dynamic: a transient increase during early-to-mid REDI training, followed by a return to levels largely comparable with the SFT baseline. This pattern is mirrored by the normalized "Wait" occurrences (right panel), which accounts for token length.

Overall, while REDI training leads to a modest reduction in average generation length, the model's propensity for explicit reflection, as indicated by "Wait" counts, shows temporary fluctuations before largely realigning with the SFT base model's characteristics after the initial tuning phase.

E.2 Case Study on Model Responses

We examine model responses to AIME 2024 II Problem 3. The generation parameters were consistent with those used in our main evaluations. The problem is presented to the model as follows:

Problem Statement: AIME 2024 II Problem 3

Solve the following math problem efficiently and clearly. The last line of your response should be of the following format: 'Therefore, the final answer is: $\boxed{{ANSWER}}$. I hope it is correct' (without quotes) where ANSWER is just the final number or expression that solves the problem. Think step by step before answering.

Find the number of ways to place a digit in each cell of a 2x3 grid so that the sum of the two numbers formed by reading left to right is 999, and the sum of the three numbers formed by reading top to bottom is 99. The grid below is an example of such an arrangement because $8 + 991 = 999$ and $9 + 9 + 81 = 99$.

0	0	8
9	9	1

The ground truth answer is 45.

E.2.1 Response from `Qwen-SFT-1.5B-3ep` (SFT Baseline)

This model answered 0/4 attempts correctly for this problem. A representative incorrect response is:

SFT Baseline Response (Incorrect)

```
<think>
Okay, so I need to figure out how many ways there are to fill a 2x3 grid...
Let me denote the grid as follows:
Row 1: a b c
Row 2: d e f

The total sum of the row numbers is abc + def = 999.
(100a + 10b + c) + (100d + 10e + f) = 999
Which simplifies to:
100(a + d) + 10(b + e) + (c + f) = 999

The total sum of the column numbers is ad + be + cf = 99.
10a + d + 10b + e + 10c + f = 99
Which simplifies to:
10(a + b + c) + (d + e + f) = 99

Let me denote S1 = a + d, S2 = b + e, S3 = c + f
Then equation 1 becomes:
100*S1 + 10*S2 + S3 = 999

Equation 2 becomes:
10*(a + b + c) + (d + e + f) = 99
But a + b + c + d + e + f = S1 + S2 + S3
Therefore, substituting into equation 2:
10*(S1 + S2 + S3 - S3) + S3 = 99 [Error in reasoning: S1+S2+S3-S3 = S1+S2, not a+b
    +c]
Which simplifies to:
10*(S1 + S2) + S3 = 99

So equation 2 is 10*(S1 + S2) + S3 = 99
Equation 1 is 100*S1 + 10*S2 + S3 = 999

Let me write equation 2 as 10*S1 + 10*S2 + S3 = 99
Then subtract equation 2 from equation 1:
(100*S1 + 10*S2 + S3) - (10*S1 + 10*S2 + S3) = 999 - 99
90*S1 = 900 ⇒ S1 = 10

Then substitute back into equation 2: 10*10 + 10*S2 + S3 = 99
⇒ 100 + 10*S2 + S3 = 99 ⇒ 10*S2 + S3 = -1

But S2 and S3 are sums of digits, so they must be non-negative.
Therefore, 10*S2 + S3 = -1 is impossible. Therefore, there are no solutions.
[...]
Wait, let me check the example again. The example is:
Row 1: 0 0 8 → 8
Row 2: 9 9 1 → 991
Sum: 8 + 991 = 999
Columns:
First column: 0 9 → 09
Second column: 0 9 → 09
Third column: 8 1 → 81
Sum: 09 + 09 + 81 = 99
Yes, this is correct. So the example is a valid solution.
[...]
In the example, S1 = a + d = 0 + 9 = 9, S2 = b + e = 0 + 9 = 9, S3 = c + f = 8 + 1
     = 9.
Then equation 1: 100*9 + 10*9 + 9 = 999. Correct.
Equation 2 [Model A's version]: 10*(S1 + S2) + S3 = 10*(9 + 9) + 9 = 189. Which is
     not equal to 99.
So equation 2 is not satisfied. Therefore, the equations are inconsistent.
[...]
Therefore, according to the equations, there are no solutions. Therefore, the
    number of ways is zero.

**Final Answer**
The number of ways is \boxed{0}.
</think>
[...]
Thus, the number of ways to place the digits in the grid is \(\boxed{0}\).
```

Analysis of SFT Baseline Response: The SFT model correctly set up the initial two main equations based on row and column sums. However, it made a critical algebraic error when attempting to rewrite the second equation (sum of column numbers) in terms of intermediate sums $S_1 = a + d, S_2 = b+e, S_3 = c+f$. Specifically, it incorrectly assumed $a+b+c = S_1+S_2$. This flawed transformation led to an inconsistent system of equations, resulting in the conclusion that $10S_2 + S_3 = -1$, which is impossible for sums of digits. Consequently, it incorrectly answered 0. Even when checking the provided example, which contradicted its derived system, the model failed to identify its algebraic mistake and instead reinforced its belief in the inconsistency.

E.2.2 Response from `Qwen-SFT-1.5B-3ep + REDI` ($\alpha = 0.8$, Checkpoint 1400)

This model, taken from an intermediate stage of REDI training (step 1400 out of 1661 total steps for 1 epoch), answered 2/4 attempts correctly for this problem. A representative correct response is:

REDI-tuned Response (Correct)

```
<think>
Okay, so I need to figure out how many ways there are to fill a 2x3 grid...
Let the grid be:
  a b c
  d e f
Row 1 sum: R1 = 100a + 10b + c
Row 2 sum: R2 = 100d + 10e + f
Column 1 sum: C1 = 10a + d
Column 2 sum: C2 = 10b + e
Column 3 sum: C3 = 10c + f
Given that R1 + R2 = 999 and C1 + C2 + C3 = 99.
So, first equation: 100(a + d) + 10(b + e) + (c + f) = 999
Second equation: (10a + d) + (10b + e) + (10c + f) = 99
[...]
Let X = a + d, Y = b + e, Z = c + f.
Row sum equation: 100X + 10Y + Z = 999
Column sum equation [initial incorrect attempt by Model B]: 10X + Y + Z = 99
Subtracting the column sum equation from the row sum equation:
90X + 9Y = 900
Divide by 9: 10X + Y = 100
Then, substituting back into column sum equation:
100 + Z = 99 => Z = -1
Which is impossible. Therefore, there is a contradiction, meaning that there are
no solutions? But the problem states that there are such arrangements, so I must
have made a wrong assumption.
[...]
Wait, let's think about the addition of the two three-digit numbers R1 and R2
    resulting in 999.
100(a + d) + 10(b + e) + (c + f) = 999
So, 100X + 10Y + Z = 999, where X = a + d, Y = b + e, Z = c + f.
[...] So X can be at most 9. [...] So X must be at least 9. Therefore, X=9.
Therefore, X=9, so 100*9 + 10Y + Z = 999 => 900 + 10Y + Z = 999 => 10Y + Z = 99.
Therefore, Y=9, Z=9. So:
X = a + d = 9
Y = b + e = 9
Z = c + f = 9
Therefore, S = a + b + c + d + e + f = (a + d) + (b + e) + (c + f) = X + Y + Z = 9
     + 9 + 9 = 27
[...]
But also, C1 + C2 + C3 = 10(a + b + c) + (d + e + f) = 99
But a + b + c = 27 - (d + e + f). Let me denote D = d + e + f.
Therefore, C1 + C2 + C3 = 10*(27 - D) + D = 270 - 10D + D = 270 - 9D
But we know that C1 + C2 + C3 = 99. Therefore:
270 - 9D = 99 => -9D = -171 => D = 19
Therefore, d + e + f = 19
[...]
So, we have the following equations:
1. a + d = 9
2. b + e = 9
3. c + f = 9
4. d + e + f = 19
5. a + b + c = 8 (since 27-19=8)
6. All variables a, b, c, d, e, f are digits (0-9)
[...]
The number of non-negative integer solutions to d + e + f = 19 with d, e, f <= 9.
Using inclusion-exclusion principle.
[...]
```

```
First term: C(19 + 3 -1, 3 -1) = C(21, 2) = 210
Subtract the cases where at least one variable >=10. [...] 3*55=165.
Now, add back the cases where two variables >=10. [...] No solutions.
Therefore, total solutions=210 -165 +0 -0=45.
Therefore, 45 solutions. So there are 45 possible triples (d, e, f) that sum to
    19.
But each such triple corresponds to a unique (a, b, c) since a=9 - d, b=9 - e, c=9
     - f.
Therefore, the number of ways is 45.

But let's confirm with an example...

Therefore, the number of ways is 45. Therefore, the final answer is \boxed{45}.
**Final Answer**
\boxed{45}
</think>
[...]
Thus, the number of ways to place the digits in the grid is:
\[
\boxed{45}
\]
```

Analysis of REDI-tuned Response: The REDI-tuned model also correctly established the initial equations. Notably, it initially made a similar algebraic error when attempting to use intermediate sums, leading to a momentary contradiction ($Z = -1$). However, unlike the SFT baseline, this model demonstrated an enhanced ability to self-correct or find an alternative path. It revisited the first primary equation ($100(a + d) + 10(b + e) + (c + f) = 999$) and correctly deduced from the properties of digit sums that $a + d = 9, b + e = 9, c + f = 9$. This crucial insight allowed it to determine the total sum of all digits (27). Using this, along with the second primary equation ($10(a+b+c)+(d+e+f) = 99$), it correctly solved for the sum of digits in the top row ($a+b+c = 8$) and bottom row ($d + e + f = 19$). Finally, it correctly applied the Principle of Inclusion-Exclusion to count the number of ways to form $d + e + f = 19$ with digits, leading to the correct answer of 45.

This case study suggests that REDI training, by incorporating signals from both positive and negative reasoning traces, can enhance a model's ability to navigate complex problem-solving paths, including recovering from intermediate errors and identifying correct solution strategies, which might be less developed in models trained solely on positive examples.

F Additional Analysis on Training Dynamics

To further investigate the training dynamics of REDI, particularly the observed phenomenon where both chosen (y_w) and rejected (y_l) log-probabilities (LogPS) can decline even in stable runs, we tracked additional statistics. The specific run analyzed here is our REDI configuration with $\alpha = 0.8$, initialized from the `Qwen-SFT-1.5B-3ep`. We randomly sampled 4 prompts from the preference dataset ($\mathcal{D}_{\text{Pref}}$), supplemented by 1 test prompt from AIME24, 1 from AIME25, and 2 from MATH-500. For the training data samples, we analyzed the logits directly. For the test data samples, we first performed auto-regressive generation and then analyzed the logits of the generated sequences.

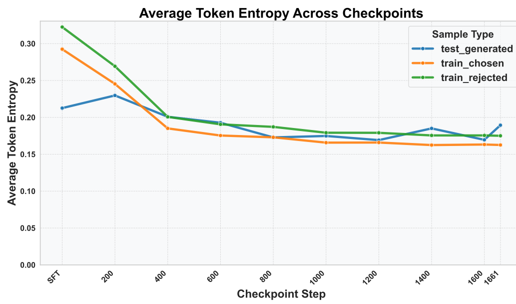

Figure 9: Average token entropy dynamics across training checkpoints for different sample types, for the REDI $\alpha = 0.8$ run. "SFT" denotes the initial model state (`Qwen-SFT-1.5B-3ep`). "train_chosen" and "train_rejected" refer to sequences from the preference dataset, while "test_generated" refers to sequences generated by the model on test prompts.

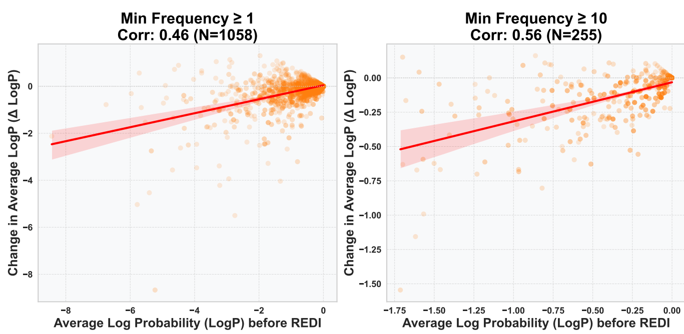

Figure 10: Correlation between the change in average token log-probability (LogP) after REDI training (Δ LogP) and the average token LogP before REDI training (i.e., in the `Qwen-SFT-1.5B-3ep` model). Analysis performed on tokens from chosen responses in the training data for the REDI $\alpha = 0.8$ run. Left: Tokens with a minimum frequency ≥ 1. Right: Tokens with a minimum frequency ≥ 10. A positive correlation indicates that tokens with initially higher LogP tend to see their LogP increase (or decrease less), while tokens with initially lower LogP tend to experience a larger decrease.

Figure 9 illustrates the average token entropy across training checkpoints for the REDI $\alpha = 0.8$ run. We observe a rapid decrease in entropy from the initial SFT model state to approximately step 400-600 of REDI training. This decrease is more pronounced for sequences from the training data ("train_chosen" and "train_rejected") compared to sequences generated by the model on test prompts ("test_generated"). This suggests that the model becomes more confident (i.e., assigns sharper probability distributions) over tokens when conditioned on training sequences. After the initial drop, the entropy tends to stabilize or fluctuate slightly.

To understand where the model's probability mass is shifting, we analyzed the change in token log-probabilities (LogPs) relative to their initial values in the `Qwen-SFT-1.5B-3ep` model. Figure 10 displays this relationship for tokens in the chosen responses from the training data, specifically for the REDI $\alpha = 0.8$ run. A moderate positive correlation is observed (Pearson correlation coefficient of 0.46 for tokens with frequency ≥ 1, and 0.56 for tokens with frequency ≥ 10). This positive correlation suggests that, on average:

- Tokens for which the SFT model already had a high probability (higher initial LogP) tend to see their probabilities further increase or decrease less after REDI training.
- Conversely, tokens for which the SFT model had a low probability (lower initial LogP) tend to experience a more significant decrease in their probabilities.

In essence, REDI appears to amplify the model's existing tendencies to some extent, making it more confident about tokens it was already likely to predict and even less confident about tokens it was unlikely to predict. This behavior, where negative gradients might inadvertently suppress probabilities of tokens beyond the specific negative example, has been discussed in recent literature on off-policy preference optimization [Yan et al., 2025, Ren and Sutherland, 2025, Razin et al., 2025].

Despite these complex dynamics and the observed shifts in token probabilities, our main results in Section 4 indicate that as long as the training process avoids catastrophic collapse (which REDI's asymmetric weighting helps to prevent), the model achieves strong performance improvements on downstream reasoning tasks.

The analyses presented in this section are preliminary and offer initial insights into REDI's learning mechanisms for the specific $\alpha = 0.8$ configuration. A more comprehensive understanding of how REDI precisely refines the model's internal representations and generation strategies warrants further investigation.