# Prototype Entropy Alignment: Reinforcing Structured Uncertainty in LLM Reasoning

**Zhengyuan Pan[1,*], Yanhao Chen[1,*], Zhongquan Jian[3,†], Wanru Zhao[2], Haonan Ma[4], Meihong Wang[2], Qingqiang Wu[1,2,5,†]**

[1]School of Film, Xiamen University, Xiamen, China
[2]School of Informatics, Xiamen University, Xiamen, China
[3]School of Computer and Data Science, Minjiang University, Fuzhou, China
[4]University of Chinese Academy of Sciences, Beijing, China
[5]Key Laboratory of Digital Protection and Intelligent Processing of Intangible Cultural Heritage of Fujian and Taiwan, Ministry of Culture and Tourism, Xiamen University, China
zypan@stu.xmu.edu.cn, jianzq@mju.edu.cn, wuqq@xmu.edu.cn

## Abstract

Recent research reveals that a minority of high-entropy tokens significantly influence the reasoning quality of large language models (LLMs). Inspired by this, we propose Prototype Entropy Alignment (PEA), a reinforcement learning framework that models effective reasoning not as a single path but as a collection of learnable "entropy signatures." PEA identifies these signatures by clustering expert trajectories' uncertainty patterns into a diverse and continuously updated set of prototypes. The model is then rewarded for aligning its own reasoning process with these evolving targets, creating a self-improvement loop. Instead of replacing traditional outcome-based rewards, PEA provides a complementary, process-oriented signal. Our experiments show that this synergy is crucial: PEA substantially boosts performance on creative and general reasoning tasks and, when combined with outcome rewards, achieves SOTA results on structured tasks such as mathematics. By rewarding alignment with diverse and evolving reasoning structures, PEA offers a robust, verifier-free pathway to enhance reasoning's adaptability.

**Code** — https://github.com/scut-pzy/PEA_verl_unsloth

## Introduction

Enhancing the complex reasoning capabilities of Large Language Models (LLMs) remains a central challenge, with Reinforcement Learning (RL) emerging as a promising solution (Ouyang et al. 2022a; Sheng et al. 2025). Current RL approaches, however, rely largely on two reward schemes that face inherent limitations. Outcome-based RL provides only sparse, final-answer rewards, leading to severe credit-assignment challenges in long-chain reasoning and offering little guidance on how to improve intermediate steps (Yu et al. 2025a).

A second direction employs *step-level reward shaping*, which assigns rewards to intermediate reasoning steps rather than solely the final output (Liu et al. 2025a). Although this yields finer-grained supervision, it creates a strong dependency on external verifiers whose construction is often as difficult as the task itself. Moreover, these verifiers are highly vulnerable to *reward hacking*, enabling models to exploit imperfections with superficially valid yet semantically incorrect steps (Cui et al. 2025c).

Meanwhile, entropy in RL is typically treated as a scalar for exploration control or mode-collapse prevention (Cui et al. 2025b; Hao et al. 2025; Zhu et al. 2025; Yang et al. 2025), thereby overlooking its structured distribution during reasoning. In practice, strong reasoning exhibits localized entropy peaks at critical deliberation points rather than uniformly high uncertainty. Motivated by this observation, we analyze the positional distribution of high-entropy tokens as a richer descriptor of a model's internal reasoning dynamics.

Building on these insights, we propose **Prototype Entropy Alignment (PEA)**, a complementary reward dimension that leverages entropy as an intrinsic, process-level signal. We first extract entropy-based signatures from high-quality reasoning traces and cluster them into canonical "prototypes." The policy model is then rewarded for aligning its reasoning process with these expert patterns, providing guidance that is orthogonal to purely outcome-based supervision. Furthermore, PEA integrates a dynamic updating mechanism that continually refreshes the prototype bank, enabling the discovery and assimilation of emerging reasoning strategies. This results in a self-improving learning loop that avoids stagnation on a fixed set of patterns and encourages the development of diverse and robust reasoning behaviors. Our key contributions are summarized as follows:

- We introduce PEA, a novel RL framework that employs intrinsic entropy signals as a complementary reward source, reducing reliance on fine-grained external verifiers and mitigating key issues such as reward hacking and reasoning stagnation.
- We demonstrate that PEA produces nuanced yet substantial improvements: it significantly enhances specialized reasoning models on structured tasks and yields univer-

[*]These authors contributed equally.
[†]Corresponding authors.

sal gains on open-ended creative tasks. When combined with standard outcome-based rewards, PEA consistently achieves the strongest overall performance, highlighting a strong synergistic effect.

- We provide compelling evidence that dynamically assimilating new prototypes is critical for sustained improvement, enabling models to continuously integrate novel reasoning strategies and preventing collapse onto a limited set of initial patterns.

## Related Work

### Reinforcement Learning for LLM Reasoning

RL has become the core paradigm in the post-training phase of LLMs (Ouyang et al. 2022a; Zhou et al. 2023), with its objectives evolving from 'human alignment' towards 'reasoning enhancement'. Early efforts focused on improving instruction following and preference alignment through Reinforcement Learning with Human Feedback (RLHF) (Ouyang et al. 2022b). These methods include online algorithms (e.g., PPO (Schulman et al. 2017), REINFORCE (Williams 1992), RLOO (Ahmadian et al. 2024)) and offline algorithms (e.g., DPO (Rafailov et al. 2023), KTO (Ethayarajh et al. 2024)). Although offline methods improve training efficiency, their performance typically lags behind their online counterparts (Tang et al. 2024). Following OpenAI's initial validation of RL's feasibility in long-chain reasoning (OpenAI 2024), RLVR (Lambert et al. 2024) has emerged as a prominent method for enhancing LLM reasoning. It has been particularly successful in domains with clear correctness standards, such as mathematical reasoning and code generation (Guo et al. 2025b; Shao et al. 2024a; Xiaomi et al. 2025; Team et al. 2025; Team 2025b), attracting significant attention (Cui et al. 2025a; Liu et al. 2025b; Hu et al. 2025). While effective, these approaches often lead models to find shortcuts to maximize rewards—a phenomenon known as reward hacking—rather than genuinely improving their reasoning abilities. Inspired by this challenge, subsequent works such as GRPO (Shao et al. 2024b) and its variants (e.g., DAPO (Yu et al. 2025b)) have demonstrated advantages in optimization.

### RLVR in Entropy

The success of RLVR has renewed interest in understanding LLM reasoning (Gandhi et al. 2025; Li et al. 2025). Prior work showed that certain Chain-of-Thought (CoT) tokens disproportionately affect final answers, termed "critical tokens" (Vassoyan, Beau, and Plaud 2025; Lin et al. 2024). Wang et al. (Wang et al. 2025) further linked these critical tokens to high-entropy positions, finding that only about 20% of such "branching" tokens drive performance gains, and that updating policy gradients solely on them can match or exceed full-trajectory training.

## Methodology

This section details the technical framework of PEA. Our objective is to align a policy model's reasoning process with abstract prototypes distilled from high-quality expert trajectories. Unlike conventional reward models that often incentivize rigid imitation of specific token sequences (e.g., fixed CoT templates), PEA operates on a higher level of abstraction. It guides the model by aligning its behavior with *structured uncertainty patterns*—specifically, the distribution of high-entropy tokens—observed in expert reasoning. This design promotes structured exploration: the model is encouraged to follow the *macro-level structure* of expert reasoning (i.e., where to be uncertain or pivotal) while retaining the freedom to generate diverse, novel content at the *micro-level* (i.e., the specific tokens). This mechanism is designed to facilitate the acquisition of **Rare Behaviors**: effective and creative reasoning strategies that are rarely sampled by the base model, but are crucial for solving challenging problems (Cheng et al. 2025).

As illustrated in Figure 1, the PEA architecture is organized into two primary phases. The **offline phase** (Phase A: Prototype Initialization) constructs a library of expert prototypes from high-quality reasoning trajectories. Subsequently, in the **online phase** (Phase B: Reinforcement Learning), these prototypes serve as dynamic alignment targets to guide the policy model during reinforcement learning. The implementation comprises three core steps:

### Building the Expert Reasoning Trace Corpus

We begin with the offline process of constructing a corpus of high-quality reasoning traces, which involves generation and filtering. We employ **Qwen3-32B** and **Qwen3-30B-A3B** (Team 2025a) as generator models to produce initial reasoning trajectories and **ArmoRM-Llama3-8B** (Wang et al. 2024), a state-of-the-art reward model, for quality assessment, with additional rechecks by GPT5 and humans. This curated corpus forms the foundation for our subsequent entropy-based clustering and prototype extraction. Specifically, ArmoRM provides a holistic quality score by integrating multiple dimensions (e.g., correctness, coherence, safety) and employs a Mixture-of-Experts architecture to dynamically weight these objectives, thereby mitigating common reward hacking issues such as length bias.

For each prompt in our training set, we sample multiple CoT trajectories from the generator model. To foster diversity in reasoning paths, we utilize high-temperature and nucleus sampling $top\_p$. Since the output includes both the reasoning process and the final answer, we truncate each generation to isolate the token sequence corresponding to the reasoning steps.

Next, we employ a two-stage filtering process to isolate high-quality traces that correspond to problems of moderate difficulty.

**Stage 1: Difficulty Filtering**. We first filter based on problem difficulty. For each prompt, we compute the Pass@16 success rate using the generated trajectories. We retain the prompts where $0 < \text{Pass@16} \leq 0.5$, effectively discarding those that are either trivial for the model (Pass@16 approaching 1) or seemingly unsolvable (Pass@16 = 0). This step ensures that our dataset comprises challenging yet solvable problems.

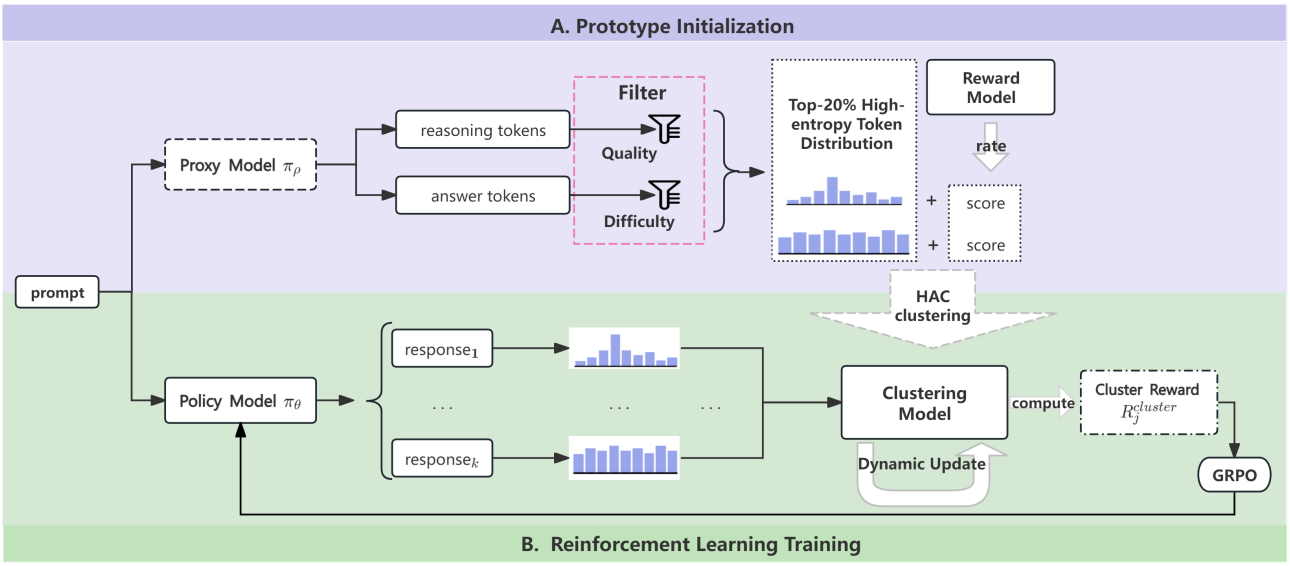


Figure 1: The process consists of two main phases: **(1) Prototype Initialization**: A strong proxy model ($\pi_\rho$) generates correct reasoning traces. We extract their Top-20% high-entropy token distributions (i.e., entropy signatures) to form an initial set of expert prototypes, scored by a reward model. **(2) Reinforcement Learning**: The policy model ($\pi_\theta$) generates responses. A reward is calculated based on the alignment of its response's entropy signature with the expert prototypes via a clustering model. This reward guides the policy update via GRPO. Critically, the **dynamic update** loop allows the framework to absorb novel, high-quality signatures discovered by the policy, enabling continuous self-improvement.

**Stage 2: Reasoning Quality Filtering**. For the set of prompts selected in Stage 1, we now filter their associated trajectories based on reasoning quality. ArmoRM scores each trace. This step complements the outcome-based Pass@16 metric by assessing the intrinsic quality of the reasoning process itself (e.g., coherence, logical flow). We retain only the highest-scoring traces for each prompt, as determined by ArmoRM and rechecks, to form our final corpus of expert trajectories. For each trace in the final corpus, we compute its token-level entropy sequence using Equation (1):

$$H_t = -\sum_{v \in \mathcal{V}} p_t(v) \log p_t(v), \tag{1}$$

where $p_t(v)$ is the predicted next-token distribution at step $t$. From the full entropy sequence, we select the top 20% highest-entropy positions. We then partition their relative sequence positions into $B = 10$ equal-width bins. We empirically set $B = 10$, as this granularity balances the need to capture distributional shifts without being overly sensitive to minor positional variations. By counting the occurrences of high-entropy tokens in each bin, we form a normalized, length-10 histogram vector $h_i$.

At this stage, we have constructed a set of high-quality, moderately challenging reasoning trajectories, along with their corresponding entropy representations $\{h_i\}$ and their scores. These will be hierarchically clustered in the following section to extract representative entropy prototypes.

## Entropy Prototype Clustering

Given the set of high-quality reasoning traces constructed in Section A, we proceed to extract representative expert prototypes. Our inputs are the set of the top-20% entropy distribution vectors, $\mathcal{H} = \{\mathbf{h}_1, \ldots, \mathbf{h}_N\}$, and their corresponding ArmoRM scores, $\mathcal{S} = \{s_1, \ldots, s_N\}$. Each vector $\mathbf{h}_i \in \mathbb{R}^{10}$ is a normalized histogram in ten relative position bins, encoding the proportion of high-entropy tokens in each segment of a reasoning trace. This representation thereby captures the "shape" of uncertainty throughout the reasoning process.

We apply **hierarchical agglomerative clustering** to $\mathcal{H}$, using cosine distance and average linkage (also used in Section ). The number of clusters, $K$, is determined by selecting a cut-point on the dendrogram in a principled manner that balances intra-cluster similarity and inter-cluster dissimilarity, often guided by metrics such as the silhouette score. Table 1 shows key hyperparameters. This process yields $K$ clusters $\{\mathcal{C}_j\}_{j=1}^K$, each grouping trajectories with similar entropy patterns. For each cluster, we define a prototype and its associated quality bias according to Equation (2):

$$\mathbf{c}_j = \frac{1}{|\mathcal{C}_j|} \sum_{\mathbf{h} \in \mathcal{C}_j} \mathbf{h}, \quad s_j \quad = \frac{1}{|\mathcal{C}_j|} \sum_{\mathbf{h}_i \in \mathcal{C}_j} s_i, \tag{2}$$

where we define the initial prototype bank as $\mathcal{P}_{\text{init}} = \{(\mathbf{c}_j, s_j)\}_{j=1}^K$. Each prototype $\mathbf{c}_j$ encapsulates a canonical entropy distribution pattern, while $s_j$ represents the mean ArmoRM quality score for trajectories exhibiting that pat-

Algorithm 1: Extracting Expert Reasoning Prototypes. The algorithm distills high-quality reasoning trajectories into a set of abstract prototypes by clustering their token-level uncertainty patterns (entropy histograms).

**Input:** Prompt set $\mathcal{D}$, Generator $\pi_g$, Reward Model RM
**Output:** Prototype Bank $\mathcal{P} = \{(\mathbf{c}_j, s_j)\}_{j=1}^{K}$

$\mathcal{T}_{\text{raw}} \leftarrow$ SampleDiverseTraces($\mathcal{D}, \pi_g$) {Generate varied CoT reasoning paths}
$\mathcal{T}_{\text{expert}} \leftarrow$ CurateExpertDemonstrations($\mathcal{T}_{\text{raw}}$, RM) {Filter for high process quality on solvable problems}
$\mathcal{H}, \mathcal{S} \leftarrow$ ConvertToEntropySignatures($\mathcal{T}_{\text{expert}}$) {Represent each trace by its uncertainty dynamics}
$\{\mathcal{C}_j\}_K \leftarrow$ IdentifyReasoningStyles($\mathcal{H}$) {Group signatures via hierarchical clustering}
$\mathcal{P} \leftarrow$ DefinePrototypes($\{\mathcal{C}_j\}, \mathcal{S}$) {Compute centroid and mean score for each style}
**return** $\mathcal{P}$

tern. Importantly, alignment is primarily driven by distributional proximity: the policy model is encouraged to produce an entropy vector close to an expert prototype, and different tasks may naturally favor different prototypes. Quality scores, denoted by $s_j$, introduce a subtle bias into the downstream reward, favoring higher-quality prototypes without enforcing exclusivity. The algorithm 1 summarizes this static prototype extraction process for reproducibility. The resulting prototype bank provides multiple distinguishable, quality-aware alignment targets for the reward design and policy optimization detailed in the following section.

## Policy Optimization via Dynamic Prototype Alignment

We now detail the online policy optimization phase. The policy model, $\pi_\theta$, is optimized using a reinforcement learning framework designed to balance the imitation of known expert patterns with the exploration and dynamic assimilation of novel, effective ones.

**GRPO Policy Optimization** We employ the GRPO (Shao et al. 2024b) algorithm. Equation (3) defines an objective that stabilizes and guides policy updates using an advantage estimate, $\hat{A}_{i,t}$, and a group-level reward weight, $r_{i,t}$.

$$J_{\text{GRPO}}(\theta) = \frac{1}{G}\sum_{i=1}^{G}\frac{1}{T_i}\sum_{t=1}^{T_i}\min\Big(r_{i,t}\hat{A}_{i,t}, \text{clip}(r_{i,t}, 1-\epsilon, 1+\epsilon)\hat{A}_{i,t}\Big) + KL \quad (3)$$

where $\tau_i$ denotes a trajectory and $\mathcal{B}$ is a batch. The core of GRPO lies in the reward weight $r_i$, which is computed from the total reward quantile of trajectory $\tau_i$ within the batch, thus prioritizing updates from higher-return sample groups. $\hat{A}_{i,t}$ is the standard advantage estimate. The full objective function, not shown for brevity, typically includes a KL-divergence penalty against a reference policy $\pi_{\text{ref}}$ to regularize policy updates and prevent catastrophic forgetting.

**PEA Reward and Dynamic Prototype Update** The core of our method is the PEA reward function, $R_{\text{PEA}}$, which provides adaptive feedback. For each newly generated trajectory with entropy histogram $\mathbf{h}$, we first identify its closest prototype $\mathbf{c}_{j^*}$ from the library $\mathcal{P}$ by computing the minimum distance $d_{j^*} = \min_j d(\mathbf{h}, \mathbf{c}_j)$. The PEA reward is then calculated using the three-branch conditional structure defined in Equation (4):

$$R_{\text{PEA}} = \begin{cases} S'_{j^*} & \text{if } d_{j^*} \le \delta_{\text{match}}, \\ R_{\text{novelty}} & \text{if } d_{j^*} > \delta_{\text{match}} \wedge \text{RM}(\tau) \ge \theta_{\text{high}}, \\ -\kappa \cdot d_{j^*} & \text{otherwise.} \end{cases} \quad (4)$$

where $S'_{j^*}$ is the normalized quality score of the prototype $j^*$, $\text{RM}(\tau)$ is the quality score of the full trajectory $\tau$ by ArmoRM, and $\theta_{\text{high}}$ is a high-quality threshold. These three branches correspond to:

- **Imitation Reward:** When a trajectory's entropy pattern aligns with a known expert prototype ($d_{j^*} \le \delta_{\text{match}}$), the model receives a reward equal to the prototype's quality score, $S'_{j^*}$ (a normalized version of $s_{j^*}$). This encourages the replication of proven, high-quality reasoning structures.
- **Discovery Reward & Dynamic Update:** When a trajectory exhibits a novel entropy pattern ($d_{j^*} > \delta_{\text{match}}$) and is independently verified as high-quality by the reward model ($\text{RM}(\tau) \ge \theta_{\text{high}}$), it signifies a successful exploration. The model receives a constant discovery reward, $R_{\text{novelty}}$. Critically, this new high-quality pair $(\mathbf{h}, \text{RM}(\tau))$ is added to a buffer $\mathcal{B}_{\text{new}}$. Periodically, the prototype library $\mathcal{P}$ is updated by re-clustering with the contents of this buffer, allowing the system to assimilate new expert patterns dynamically. Additionally, the efficient reward computation, coupled with an asynchronous update process, ensures the RL training loop remains unimpeded.
- **Exploration Penalty:** In all other cases (i.e., novel but low-quality trajectories), the model receives a penalty proportional to its deviation from the nearest prototype, scaled by $\kappa$. This discourages aimless exploration far from known 'good' regions of the policy space.

All reward components are clipped to the range $[-1, 1]$ to ensure training stability. This entire mechanism forms an adaptive, closed-loop system where the policy model balances between exploiting known expert patterns and exploring novel ones. By continuously internalizing newly discovered strategies as new alignment targets, PEA encourages the model to expand its repertoire of effective reasoning skills progressively.

# Experiment Settings

## Backbone Models

To assess the differences between our proposed *PEA reward* and conventional RLVR methods that rely on answer-based verifiers, we conduct a series of experiments. Our approach is applied not only to the original Qwen-2.5-1.5B/7B models (Team 2024) but also to two publicly released Qwen derivatives: DeepSeek-R1-Distill-Qwen-7B (DeepSeek-AI 2025) and Nemotron-Research-Reasoning-Qwen-1.5B (Liu et al. 2025a). We treat these community models as strong, pre-tuned baselines and further fine-tune them with GRPO

| Parameter | Value |
|---|---|
| Clustering Algorithm | Hierarchical Agglomerative |
| Number of Clusters ($K$) | 15 |
| Distance Metric | Cosine |
| Linkage Criterion | Average |
| Prototype Quality Score | ArmoRM Score |

Table 1: Key hyperparameters for entropy clustering.

| Hyperparameter | Value |
|---|---|
| Optimizer | AdamW |
| Policy learning rate | $1 \times 10^{-6}$ |
| Training batch size | 256 |
| Rollout number per prompt | 16 |
| Mini-batch size | 32 |
| $\mathcal{B}_{\text{new}}$ | 500 |
| Max response length | 32000 tokens |
| Rollout temperature | 1.2 |
| Clip range $\epsilon_{\text{low}}$, $\epsilon_{\text{high}}$ | 0.2, 0.28 |

Table 2: Hyperparameters for RL training.

under our PEA reward scheme. To demonstrate the generality and self-sufficiency of our reward paradigm, we adopt a core experimental design: In the course of the experiments, the computation of rewards is based exclusively on the **alignment with high-entropy token distributions**, without reliance on external verifiers or on the correctness of answers.

## RL Training Configuration

**Training Dataset** The training corpus comprises two main sources. First, we use the **MATH**, drawing on the version curated by (Yu et al. 2025a) in their DAPO work. Second, we assemble a **General Reasoning** dataset, carefully selected to minimize overlap with pretraining corpora and to emphasize complex, multi-step factual reasoning and argumentative inference. This component comprises the training splits of three publicly available benchmarks: (i) Natural Reasoning (Yuan et al. 2025), (ii) General Thought (Reasoning 2025), and (iii) WebInstruct (Ma et al. 2025).

**Training Settings** Our implementation is built upon the VeRL framework (Sheng et al. 2025) with its DAPO extensions, and adopts three key improvements from prior work (Yu et al. 2025a; Yue et al. 2025): Clip Higher, Token-level Loss, and Group Sampling. This design choice reflects the expectation that the PEA reward drives the policy away from its original output distribution toward a higher-quality regime, a transition that may temporarily cause validation accuracy to plateau or decline. The hyperparameters for RL training are provided in Table 2.

## Evaluation Benchmarks and Metrics

We evaluate our models on two categories of benchmarks to assess both mathematical and general reasoning capabilities. For datasets included in our training corpus, we strictly evaluate them based on their held-out test splits.

| Hyperparameter | Value |
|---|---|
| Max response length | 32000 tokens |
| Temperature | 0.8 |
| Top-p | 0.95 |

Table 3: Hyperparameters for Evaluation.

**Mathematical Reasoning** We use AIME 2024 and AIME 2025(MAA 2024, 2025), AMC 2023(MAA 2023), and the MATH500(Hendrycks et al. 2021; Shao et al. 2024a) subset. Following standard practice, a prediction is marked correct (1) only if the final numerical answer exactly matches the ground truth; otherwise, it is marked incorrect (0).

**General Reasoning** We use the test splits of NaturalReasoning (Yuan et al. 2025), GeneralThought-430K (Reasoning 2025), and WebInstruct (Ma et al. 2025). Following (Guo et al. 2025a), we use a generative RM with their Reward Prompt as an automated grader. A response scores 1 if its RM score exceeds the reference answer, and 0 otherwise. The evaluation benchmarks include:

- **AIME 2024/2025**: Two 15-problem invitational mathematics competitions.
- **AMC 2023**: A modified 83-question subset covering algebra, geometry, and pre-calculus.
- **MATH500**: 500 problems uniformly sampled from the official MATH test set.
- **NaturalReasoning**: A large-scale dataset focusing on STEM and humanities questions.
- **GeneralThought-430K**: A collection of open-domain problems with reference reasoning traces.
- **WebInstruct**: Instruction-response pairs mined from web data that emphasize multi-step reasoning.

Evaluation is performed using the hyperparameters detailed in Table 3. We conduct training on 8*NVIDIA-H20.

# Results and Analysis

## Main Performance

We evaluate our proposed PEA framework by fine-tuning four foundation models: Qwen2.5-7B/1.5B, DeepSeek-R1-Distill-Qwen-7B, and Nemotron-Research-Reasoning-Qwen-1.5B. We compare their performance against their original, non-finetuned versions. For a comprehensive comparison, we benchmark PEA against a standard baseline fine-tuned with a binary correct/incorrect (C/I) reward, as well as a configuration combining both PEA and C/I rewards. Table 4 summarizes the results on mathematical reasoning tasks, and Table 5 reports the scores on general reasoning tasks.

Our results show that PEA consistently yields performance gains across all tested models and benchmarks. Specifically, on challenging mathematical tasks such as the AIME, PEA substantially surpasses the outcome-based C/I

| Model | MATH500 | AMC23 | AIME |
|---|---|---|---|
| *Qwen2.5-7B-Instruct* | | | |
| Baseline | 76.2 | 42.3 | 11.0 |
| + C/I Reward | 79.1 (+2.9↑) | 45.8 (+3.5↑) | 13.9 (+2.9↑) |
| + PEA (Ours) | 76.7 (+0.5↑) | 40.3 (-2.0↓) | 11.9 (+0.9↑) |
| **+ PEA + C/I** | **79.3** (+3.1↑) | **48.7** (+6.4↑) | **14.6** (+3.6↑) |
| *Qwen2.5-1.5B-Instruct* | | | |
| Baseline | 54.8 | 25.1 | 2.6 |
| + C/I Reward | 61.8 (+7.0↑) | 32.0 (+6.9↑) | 16.8 (+14.2↑) |
| + PEA (Ours) | 53.2 (-1.6↓) | 24.0 (-1.1↓) | 4.7 (+2.1↑) |
| **+ PEA + C/I** | **64.7** (+9.9↑) | **33.1** (+8.0↑) | **18.0** (+15.4↑) |
| *DeepSeek-R1-Distill-Qwen-7B* | | | |
| Baseline | 93.6 | 82.8 | 47.1 |
| + C/I Reward | 90.6 (-3.0↓) | 81.8 (-1.0↓) | 46.9 (-0.2↓) |
| **+ PEA (Ours)** | 93.9 (+0.3↑) | **85.7** (+2.9↑) | 55.3 (+8.2↑) |
| + PEA + C/I | **94.1** (+0.5↑) | 84.3 (+1.5↑) | **56.7** (+9.6↑) |
| *Nemotron-Research-Reasoning-Qwen-1.5B* | | | |
| Baseline | 91.8 | 79.2 | 33.3 |
| + C/I Reward | 90.3 (-1.5↓) | 79.5 (+0.3↑) | 33.3 (-) |
| + PEA (Ours) | 92.4 (+0.6↑) | 81.1 (+1.9↑) | 35.0 (+1.7↑) |
| **+ PEA + C/I** | **92.6** (+0.8↑) | **81.3** (+2.1↑) | **35.7** (+2.4↑) |

Table 4: Performance on mathematical reasoning benchmarks. We report the mean Pass@1 accuracy over 10 runs. "+ C/I Reward" denotes fine-tuning with a binary correct/incorrect reward, while "+ PEA" refers to our proposed PEA reward. Improvements/declines relative to the baseline are shown with arrows (↑/↓). Best performance in each model block is in **bold**.

| Model | NR | GT | WI |
|---|---|---|---|
| *Qwen2.5-7B-Instruct* | | | |
| Baseline | 23.8 | 22.5 | 10.7 |
| **+ PEA (Ours)** | **31.8** (+8.0↑) | **28.3** (+5.8↑) | **22.9** (+12.2↑) |
| *Qwen2.5-1.5B-Instruct* | | | |
| Baseline | 18.9 | 27.2 | 11.5 |
| **+ PEA (Ours)** | **29.1** (+10.2↑) | **36.0** (+8.8↑) | **21.4** (+9.9↑) |
| *DeepSeek-R1-Distill-Qwen-7B* | | | |
| Baseline | 31.2 | 36.2 | 22.1 |
| **+ PEA (Ours)** | **37.9** (+6.7↑) | **43.2** (+7.0↑) | **26.1** (+4.0↑) |
| *Nemotron-Research-Reasoning-Qwen-1.5B* | | | |
| Baseline | 55.7 | 47.3 | 43.8 |
| **+ PEA (Ours)** | **61.3** (+5.6↑) | **54.8** (+7.5↑) | **47.1** (+3.3↑) |

Table 5: Performance on general reasoning benchmarks (NR: Novel-Reasoning, GT: General-Thinking, WI: Writing-Instruction). Scores represent the win rate against the baseline, where a 'win' is defined as the model's output achieving a higher score from the reward model (RM) on a given test prompt. Thus, a score of 31.8 means our model outperforms the baseline on 31.8% of the test set. All models are fine-tuned with PEA only.

reward baseline. Furthermore, its effectiveness on general reasoning tasks demonstrates that its applicability extends beyond formal domains.

**Analysis of Mathematical Reasoning Results** Table 4 reveals two key findings regarding the interplay between reward type and model architecture. First, the efficacy of process-based (PEA) versus outcome-based (C/I) rewards is highly dependent on the model's design. For standard instruction-tuned models like **Qwen2.5-7B/1.5B**, which lack a dedicated long-chain reasoning structure, the direct, outcome-based C/I reward yields substantial gains (e.g., +7.0 on AMC23 for Qwen2.5-1.5B). In contrast, applying PEA alone to these models provides marginal or even negative results (e.g., -2.0 on AMC23 for Qwen2.5-7B). We hypothesize this is because PEA attempts to align a reasoning process structure that is not inherently present or prioritized in these models, making the signal ineffective.

Conversely, for models explicitly built for complex reasoning like **DeepSeek-R1** and **Nemotron-Research-Reasoning**, the trend is reversed. Here, our process-oriented PEA reward delivers the most significant improvements, especially on difficult tasks like AIME (+8.2 for DeepSeek-R1). The simple C/I reward, however, struggles to provide a useful signal for these models, often resulting in negligible or even negative changes. This suggests that for models with strong a priori reasoning capabilities, optimizing the reasoning *process* is a far more effective strategy than relying on a sparse, final-answer reward signal.

Second, despite their varying individual effectiveness, PEA and C/I rewards demonstrate a consistent and powerful synergy. The combined '+ PEA + C/I' approach almost always yields the best performance across all models and benchmarks. For the Qwen2.5 models, PEA acts as a valuable regularizer or complementary signal to the dominant C/I reward. For the reasoning-focused models, the C/I reward adds a final layer of outcome-oriented fine-tuning on top of the process improvements driven by PEA. This robust synergy highlights the complementary nature of aligning both the reasoning process and its final outcome, regardless of the underlying model architecture.

**Analysis of General Reasoning Results** In stark contrast to the conditional improvements observed in mathematical reasoning, PEA demonstrates universal and substantial efficacy on general reasoning tasks, as shown in Table 5. On these benchmarks, where success is often defined by the quality and creativity of the reasoning process itself rather than a single correct answer, PEA consistently elevates performance across all tested models.

Notably, even the standard **Qwen2.5** models, for which PEA alone was less effective on mathematical tasks, achieve dramatic gains here. For instance, Qwen2.5-7B shows an impressive +12.2 point improvement on the WI benchmark. The reasoning-specialized models like **Nemotron** also see consistent, strong improvements (e.g., +7.5 on GT). We attribute this universal success to the nature of these tasks. Unlike mathematics, their open-ended nature makes the final outcome difficult to evaluate with a simple binary signal, rendering process-based rewards like PEA not just beneficial, but essential. By aligning the model's generation process with diverse expert reasoning patterns, PEA directly

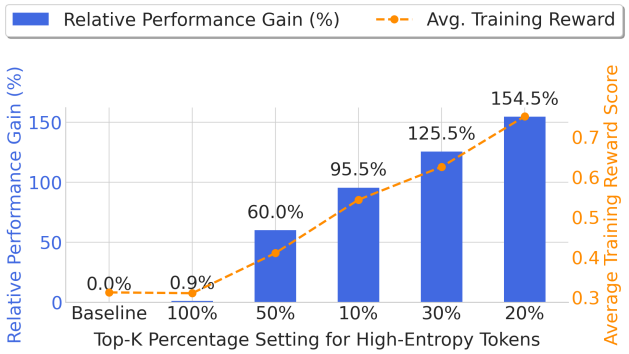


Figure 2: Ablation on the Top-K high-entropy token percentage for Qwen2.5-7B. Performance gain (bars, left axis) and average reward (line, right axis) are plotted against different thresholds. The 20% threshold provides the optimal balance, maximizing both metrics.

fosters the creativity and structural coherence required to excel in these domains.

## Ablation Study

We conduct a series of ablation studies to dissect the key components of our PEA framework.

**Ablation: High-Entropy Token Proportion** We evaluate the effect of varying the proportion of high-entropy tokens used to construct the entropy signature. The thresholds of 10%, 20%, 30%, 50%, and 100% (i.e., all tokens) are tested on Qwen2.5-7B. This examines the trade-off between signal relevance and noise: too small a proportion may omit important uncertainty cues, while too large a proportion may dilute the signal with irrelevant tokens.

Figure 2 shows that using all tokens (100%) yields no improvement and leads to a lower average reward than the baseline, indicating that including irrelevant tokens adds noise that weakens the guidance. Intermediate thresholds (10%, 30%) produce positive effects, but 20% achieves the best performance and highest average training reward. This result, consistent with prior work (Wang et al. 2025), indicates that selecting an appropriate fraction of high-entropy tokens is critical for isolating a stable and informative reward signal.

## Ablation: Static-Prototype vs. Dynamic-prototype

We investigate how the structure of prototype guidance affects learning stability and reasoning quality. We compare four configurations conceptually grouped into two regimes: **Static-Prototype strategy** uses fixed prototypes (15 clusters) that cannot absorb new reasoning patterns, whereas **Dynamic-Prototype strategy** maintains an evolving, diverse prototype set through periodic re-clustering.

Table 6 shows that dynamic prototype evolution yields substantially higher accuracy on both AIME and NaturalReasoning. It also produces a broader reward distribution, reflecting healthier exploration rather than stagnation. Figure 3 further reveals that static baselines suffer from classic

| Method | AIME 2024 | | NaturalReasoning | |
|---|---|---|---|---|
| | Acc. | Reward Var. | Acc. | Reward Var. |
| Static | 9.1 | 0.312 | 21.3 | 0.538 |
| Dynamic | **11.9** | **0.452** | **65.8** | **0.756** |

Table 6: Ablation on the dynamic prototype update mechanism. We compare our full Dynamic PEA against a static baseline where prototypes are fixed after initial clustering. The dynamic approach consistently improves task accuracy and reward variance, indicating it discovers more diverse and effective high-reward strategies.

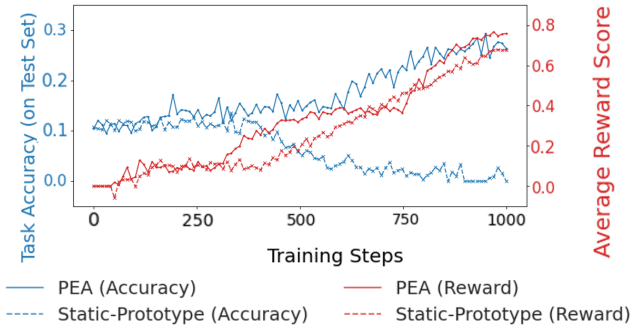


Figure 3: Training dynamics of **PEA (dynamic-prototype)** versus a static-prototype baseline. While PEA maintains aligned growth between accuracy and reward, the static-prototyp model exhibits classic reward hacking: rewards inflate while accuracy collapses. This demonstrates that dynamic prototypes are essential for preventing collapse into a static exploitable reward mode.

reward hacking: reward increases while accuracy collapses. In contrast, the dynamic multi-prototype design keeps accuracy and reward aligned throughout training, preventing collapse into a single exploitable mode.

Collectively, these results demonstrate that both *prototype dynamics* and *prototype diversity* are essential. Fixing prototypes leads the model into narrow exploitation and leads to instability, whereas evolving multi-prototype guidance enables richer strategy discovery and more robust reasoning behavior.

## Conclusion

This work introduces PEA, a framework that improves LLM reasoning by rewarding alignment with dynamically updated prototypes derived from entropy signatures from expert reasoning traces. Experiments demonstrate that this verifier-free approach improves performance on complex reasoning tasks. The multi-prototype design is essential for mitigating reward hacking. These results highlight that leveraging intrinsic patterns of the reasoning process, rather than solely final outcomes, is a promising direction for robust language model alignment.

## Acknowledgments

This work is supported by the Pre-research Project for Introduced Talents of Minjiang University (No. MJY25025), the Public Technology Service Platform Project of Xiamen City (No. 3502Z20231043), the Solfeggio Ear-Training Intelligent Robot and Cloud Platform R&D Project for Music Education (No. 2024CXY0102), the 3D Visualization Digital Twin Integrated Control System Project (No. 2023CXY0111), and the Fujian Provincial Science and Technology Major Project (No. 2024HZ022003).

## References

Ahmadian, A.; Cremer, C.; Gallé, M.; Fadaee, M.; Kreutzer, J.; Pietquin, O.; Üstün, A.; and Hooker, S. 2024. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms. *arXiv preprint arXiv:2402.14740*.

Cheng, D.; Huang, S.; Zhu, X.; Dai, B.; Zhao, W. X.; Zhang, Z.; and Wei, F. 2025. Reasoning with exploration: An entropy perspective. *arXiv preprint arXiv:2506.14758*.

Cui, G.; Yuan, L.; Wang, Z.; Wang, H.; Li, W.; He, B.; Fan, Y.; Yu, T.; Xu, Q.; Chen, W.; et al. 2025a. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*.

Cui, G.; Zhang, Y.; Chen, J.; Yuan, L.; Wang, Z.; Zuo, Y.; Li, H.; Fan, Y.; Chen, H.; Chen, W.; et al. 2025b. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*.

Cui, G.; Zhang, Y.; Chen, J.; et al. 2025c. The Entropy Mechanism of Reinforcement Learning for Reasoning Language Models. *arXiv preprint arXiv:2505.22617*.

DeepSeek-AI. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. arXiv:2501.12948.

Ethayarajh, K.; Xu, W.; Muennighoff, N.; Jurafsky, D.; and Kiela, D. 2024. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*.

Gandhi, K.; Chakravarthy, A.; Singh, A.; Lile, N.; and Goodman, N. D. 2025. Cognitive behaviors that enable self-improving reasoners, or, four habits of highly effective stars. *arXiv preprint arXiv:2503.01307*.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Wang, P.; Zhu, Q.; Xu, R.; Zhang, R.; Ma, S.; Bi, X.; et al. 2025a. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081): 633–638.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025b. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Hao, Y.; Dong, L.; Wu, X.; Huang, S.; Chi, Z.; and Wei, F. 2025. On-Policy RL with Optimal Reward Baseline. *arXiv preprint arXiv:2505.23585*.

Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; and Steinhardt, J. 2021. Measuring Mathematical Problem Solving With the MATH Dataset. *Cornell University - arXiv,Cornell University - arXiv*.

Hu, J.; Zhang, Y.; Han, Q.; Jiang, D.; Zhang, X.; and Shum, H.-Y. 2025. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*.

Lambert, N.; Morrison, J.; Pyatkin, V.; Huang, S.; Ivison, H.; Brahman, F.; Miranda, L. J. V.; Liu, A.; Dziri, N.; Lyu, S.; et al. 2024. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*.

Li, D.; Cao, S.; Griggs, T.; Liu, S.; Mo, X.; Tang, E.; Hegde, S.; Hakhamaneshi, K.; Patil, S. G.; Zaharia, M.; et al. 2025. LLMs Can Easily Learn to Reason from Demonstrations Structure, not content, is what matters! *arXiv preprint arXiv:2502.07374*.

Lin, Z.; Liang, T.; Xu, J.; Lin, Q.; Wang, X.; Luo, R.; Shi, C.; Li, S.; Yang, Y.; and Tu, Z. 2024. Critical Tokens Matter: Token-Level Contrastive Estimation Enhances LLM's Reasoning Capability. *arXiv preprint arXiv:2411.19943*.

Liu, M.; Diao, S.; Lu, X.; Hu, J.; Dong, X.; Choi, Y.; Kautz, J.; and Dong, Y. 2025a. Prorl: Prolonged reinforcement learning expands reasoning boundaries in large language models. *arXiv preprint arXiv:2505.24864*.

Liu, Z.; Chen, C.; Li, W.; Qi, P.; Pang, T.; Du, C.; Lee, W. S.; and Lin, M. 2025b. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*.

Ma, X.; Liu, Q.; Jiang, D.; Zhang, G.; Ma, Z.; and Chen, W. 2025. General-Reasoner: Advancing LLM Reasoning Across All Domains. *arXiv:2505.14652*.

MAA. 2023. American Mathematics Competition (AMC) 2023. American Mathematics Competition. Prepared by the Mathematical Association of America.

MAA. 2024. American Invitational Mathematics Examination (AIME) 2024. American Invitational Mathematics Examination. Prepared by the Mathematical Association of America.

MAA. 2025. American Invitational Mathematics Examination (AIME) 2025. American Invitational Mathematics Examination. Prepared by the Mathematical Association of America.

OpenAI. 2024. Learning to Reason with LLMs. https://openai.com/index/learning-to-reason-with-llms/.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022a. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744.

Ouyang, L.; Wu, J.; Jiang, X.; Almeida, D.; Wainwright, C.; Mishkin, P.; Zhang, C.; Agarwal, S.; Slama, K.; Ray, A.; et al. 2022b. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744.

Rafailov, R.; Sharma, A.; Mitchell, E.; Manning, C. D.; Ermon, S.; and Finn, C. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36: 53728–53741.

Reasoning, G. 2025. GeneralThought430K: Open Reasoning Dataset. https://huggingface.co/datasets/

GeneralReasoning/GeneralThought-430K. Accessed: 2025-07-30.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y.; Wu, Y.; et al. 2024a. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y.; Wu, Y.; et al. 2024b. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Sheng, G.; Zhang, C.; Ye, Z.; Wu, X.; Zhang, W.; Zhang, R.; Peng, Y.; Lin, H.; and Wu, C. 2025. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, 1279–1297.

Tang, Y.; Guo, D. Z.; Zheng, Z.; Calandriello, D.; Cao, Y.; Tarassov, E.; Munos, R.; Pires, B. Á.; Valko, M.; Cheng, Y.; et al. 2024. Understanding the performance gap between online and offline alignment algorithms. *arXiv preprint arXiv:2405.08448*.

Team, K.; Du, A.; Gao, B.; Xing, B.; Jiang, C.; Chen, C.; Li, C.; Xiao, C.; Du, C.; Liao, C.; et al. 2025. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*.

Team, Q. 2024. Qwen2.5: A Party of Foundation Models.

Team, Q. 2025a. Qwen3 Technical Report. arXiv:2505.09388.

Team, Q. 2025b. Qwq-32b: Embracing the power of reinforcement learning.

Vassoyan, J.; Beau, N.; and Plaud, R. 2025. Ignore the kl penalty! boosting exploration on critical tokens to enhance rl fine-tuning. *arXiv preprint arXiv:2502.06533*.

Wang, H.; Xiong, W.; Xie, T.; Zhao, H.; and Zhang, T. 2024. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. *arXiv preprint arXiv:2406.12845*.

Wang, S.; Yu, L.; Gao, C.; Zheng, C.; Liu, S.; Lu, R.; Dang, K.; Chen, X.; Yang, J.; Zhang, Z.; et al. 2025. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*.

Williams, R. J. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3): 229–256.

Xiaomi, L.; Xia, B.; Shen, B.; Zhu, D.; Zhang, D.; Wang, G.; Zhang, H.; Liu, H.; Xiao, J.; Dong, J.; et al. 2025. MiMo: Unlocking the Reasoning Potential of Language Model–From Pretraining to Posttraining. *arXiv preprint arXiv:2505.07608*.

Yang, Z.; Jiang, S.; Hu, C.; Li, L.; Deng, S.; and Jiang, D. 2025. Unearthing Gems from Stones: Policy Optimization with Negative Sample Augmentation for LLM Reasoning. *arXiv preprint arXiv:2505.14403*.

Yu, Q.; Zhang, Z.; Zhu, R.; Yuan, Y.; Zuo, X.; Yue, Y.; Dai, W.; Fan, T.; Liu, G.; Liu, L.; et al. 2025a. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

Yu, Q.; Zhang, Z.; Zhu, R.; Yuan, Y.; Zuo, X.; Yue, Y.; Dai, W.; Fan, T.; Liu, G.; Liu, L.; et al. 2025b. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

Yuan, W.; Yu, J.; Jiang, S.; Padthe, K.; Li, Y.; Kulikov, I.; Cho, K.; Wang, D.; Tian, Y.; Weston, J. E.; et al. 2025. Naturalreasoning: Reasoning in the wild with 2.8 m challenging questions. *arXiv preprint arXiv:2502.13124*.

Yue, Y.; Yuan, Y.; Yu, Q.; Zuo, X.; Zhu, R.; Xu, W.; Chen, J.; Wang, C.; Fan, T.; Du, Z.; et al. 2025. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*.

Zhou, C.; Liang, Y.; Meng, F.; Zhou, J.; Xu, J.; Wang, H.; Zhang, M.; and Su, J. 2023. A Multi-Task Multi-Stage Transitional Training Framework for Neural Chat Translation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(7): 7970–7985.

Zhu, X.; Xia, M.; Wei, Z.; Chen, W.-L.; Chen, D.; and Meng, Y. 2025. The surprising effectiveness of negative reinforcement in LLM reasoning. *arXiv preprint arXiv:2506.01347*.