P^{2}O: Joint Policy and Prompt Optimization

Xinyu Lu [*1,2] **Kaiqi Zhang** [*1,2] **Jinglin Yang** [3,4,5] **Boxi Cao** [1] **Yaojie Lu** [1] **Hongyu Lin** [1] **Min He** [5] **Xianpei Han** [1,2] **Le Sun** [1,2]

Abstract

Reinforcement Learning with Verifiable Rewards (RLVR) has emerged as a powerful paradigm for enhancing the reasoning capabilities of Large Language Models (LLMs). However, vanilla RLVR suffers from inefficient exploration, particularly when confronting "hard samples" that yield near-zero success rates. In such scenarios, the reliance on sparse outcome rewards typically results in zero-advantage estimates, effectively starving the model of supervision signals despite the high informational value of these instances. To address this, we propose P^{2}O, a novel framework that synergizes Prompt Optimization with Policy Optimization. P^{2}O identifies hard samples during training iterations and leverages the Genetic-Pareto (GEPA) prompt optimization algorithm to evolve prompt templates that guide the model toward discovering successful trajectories. Crucially, unlike traditional prompt engineering methods that rely on input augmentation, P^{2}O distills the reasoning gains induced by these optimized prompts directly into the model parameters. This mechanism provides denser positive supervision signals for hard samples and accelerates convergence. Extensive experiments demonstrate that P^{2}O not only achieves superior performance on in-distribution datasets but also exhibits strong generalization, yielding substantial improvements on out-of-distribution benchmarks (+4.7% avg.).

[1]Chinese Information Processing Laboratory, Institute of Software, Chinese Academy of Sciences, Beijing, China [2]University of Chinese Academy of Sciences, Beijing, China [3]Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100085, China [4]School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100085, China [5]National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing 100029, China. Correspondence to: Kaiqi Zhang <zhangkaiqi.zlk@gmail.com>, Xinyu Lu <luxinyu2021@iscas.ac.cn>.

Preprint. March 24, 2026.

1. Introduction

Large Language Models (LLMs) have achieved remarkable proficiency across a spectrum of complex reasoning tasks (Jaech et al., 2024; Guo et al., 2025). A defining characteristic of these tasks is the existence of objective verification mechanisms capable of providing deterministic feedback. Capitalizing on this, Reinforcement Learning with Verifiable Rewards (RLVR) (Lambert et al., 2024) has emerged as a dominant paradigm for reasoning alignment. By leveraging outcome-based supervision, RLVR enables models to surpass the limitations of imitation learning, facilitating autonomous exploration of the solution space to optimize reasoning trajectories aligned with rigorous verification signals (Liu et al., 2025a).

Despite its promise, the performance of RLVR is severely constrained when the training data intermixes "simple samples" with "hard samples"—specifically, instances where the model achieves near-zero success rates across K rollouts. For hard samples, the model fails to capture reliable gradient feedback due to exploration difficulties and sparse rewards (Song et al., 2025). Consequently, the optimization process heavily relies on the accessible signals from simple samples. This causes the model to overfit to simple instances, trapping it in a suboptimal policy where it excels at trivial tasks but fails to resolve complex reasoning problems.

Common approaches to mitigate exploration challenges include curriculum learning strategies, which progressively introduce harder samples as training advances (Blakeman et al., 2025), and various reward shaping techniques that provide intermediate feedback. However, curriculum learning requires heuristic-based and computationally expensive schedule generation (Bercovich et al., 2025), while reward shaping demands domain-specific expert heuristics.

The recent success of prompt optimization methods (Fernando et al., 2023; Opsahl-Ong et al., 2024; Yuksekgonul et al., 2025) offers a compelling way to break this stalemate. These methods demonstrate that even when a model fails to solve a hard problem under a standard policy, a carefully evolved prompt can often elicit the correct reasoning path. This implies that the solution lies within the model's latent search space but is inaccessible via standard gradient

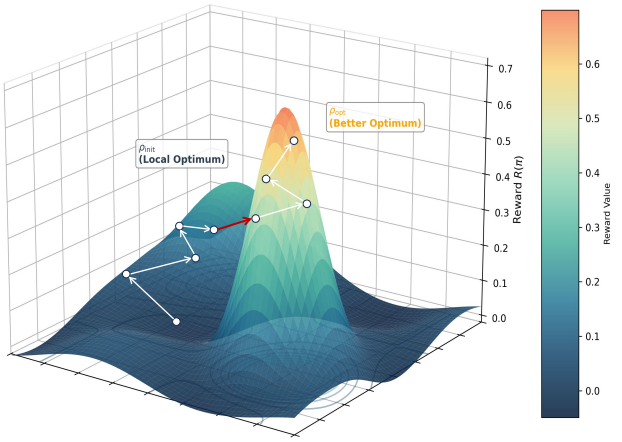

Figure 1. **Conceptual Illustration of the P^{2}O Framework.** Standard policy optimization often gets trapped in local optima (ρ_{init}) due to sparse rewards on hard samples. P^{2}O bridges this exploration gap using optimized prompts (the red arrow) to reach high-reward regions that are inaccessible via standard exploration. Subsequently, the model consolidates these gains (the white arrows) by updating its parameters to master the new region (ρ_{opt}), effectively internalizing the prompt-induced capabilities.

ascent (Zelikman et al., 2022). As illustrated in Figure 1, we visualize this mechanism as the red arrow: optimized prompts act as a bridge, enabling the model to "jump" out of the local optimum (ρ_{init}) and cross the reward-sparse valley. However, relying solely on inference-time prompts is insufficient; the ultimate goal is to internalize these capabilities. Therefore, the "jump" must be followed by consolidation—represented by the white arrows ascending the second peak—where the model parameters are updated to master the new region (ρ_{opt}).

Building on this intuition, we propose P^{2}O (Joint **P**olicy and **P**rompt **O**ptimization), a novel framework that synergizes adaptive prompt evolution with reinforcement learning to overcome the hard sample bottleneck. Specifically, P^{2}O identifies challenging instances during training and employs the Genetic-Pareto (GEPA) (Agrawal et al., 2025) prompt optimization algorithm to evolve prompts that elicit successful reasoning chains. Rather than relying on inference-time prompting, we utilize these improved trajectories as supervision signals, enabling the policy to internalize reasoning patterns directly into its parameters. This creates a virtuous cycle: as the model improves, previously hard samples become tractable, and GEPA focuses its efforts on the new frontier of difficulty. Finally, to verify the effectiveness of our method, we evaluate P^{2}O on representative reasoning benchmarks, demonstrating its superior performance compared to baselines.

The contributions of this paper are summarized as follows:

- We propose P^{2}O, a joint optimization framework that alternates between policy updates and prompt evolution. This approach addresses the exploration bottleneck in RLVR by leveraging optimized prompts to discover valid trajectories for hard samples.
- We introduce a context distillation strategy that allows the model to learn from prompt-guided trajectories without relying on them at inference. By computing gradients on the original input, the model internalizes the reasoning patterns elicited by the prompts.
- We demonstrate through experiments on six benchmarks that P^{2}O consistently outperforms GRPO baselines, showing particularly strong improvements on hard reasoning tasks such as AIME (+12.3% avg.).

2. Preliminaries

2.1. Policy Optimization

Problem Definition. In the context of LLMs, policy optimization is formulated as a Reinforcement Learning (RL) problem where the LLM acts as a policy π_θ parameterized by weights θ. Given a dataset of queries $\mathcal{D} = \{x\}$, the policy generates a response y consisting of a sequence of tokens. The quality of the generated response is evaluated by a reward function $r(x, y)$, which may be derived from ground-truth verification (e.g., in reasoning tasks) or a preference model.

Optimization Objective. The goal of policy optimization is to maximize the expected reward while ensuring the updated policy does not deviate excessively from a reference policy π_{ref}. The standard objective function is:

$$\mathrm{J}_{\text{RL}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(\cdot|x)}\left[r(x,y) - \beta\mathbb{D}_{\text{KL}}(\pi_\theta(\cdot|x)\|\pi_{\text{ref}}(\cdot|x))\right]$$

where β is a coefficient controlling the KL-divergence penalty to maintain training stability.

Group Relative Policy Optimization (GRPO). Traditional RL methods like PPO (Schulman et al., 2017) require a separate value function to estimate the baseline for advantage computation, which doubles the memory overhead. GRPO (Shao et al., 2024) eliminates the critic by employing a group-based baseline. For each query x, GRPO samples a group of K outputs $\{y_1, y_2, \ldots, y_K\}$ from the current policy. The advantage for the i-th output is computed by normalizing its reward against the group statistics:

$$A_i = \frac{r(x, y_i) - \mu_{\text{group}}}{\sigma_{\text{group}}}$$

where μ_{group} and σ_{group} are the mean and standard deviation of the rewards within the group. GRPO updates the policy by maximizing a surrogate objective based on these relative advantages, significantly improving training efficiency for reasoning tasks.

2.2. Prompt Optimization

Problem Definition. Prompt optimization treats the LLM as a frozen black-box function $\mathcal{M}$. The optimization variable is the discrete prompt z from the space of all possible natural language strings $\mathcal{P}$. Given an input x, the model generates an output $\hat{y} = \mathcal{M}(z, x)$. The problem is to find an optimal prompt z^* that maximizes a metric $S(\hat{y}, y)$ (e.g., accuracy or F1 score) over the validation dataset $\mathcal{D}_{\text{val}}$.

Optimization Objective. This is a discrete, derivative-free optimization problem. The objective is formally defined as:

$$z^* = \arg\max_{z \in \mathcal{P}} \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{val}}} \left[S(\mathcal{M}(z, x), y) \right] \tag{1}$$

Unlike policy optimization, which adjusts continuous weights θ, prompt optimization searches the discrete semantic space of instructions to elicit better capabilities from the fixed model.

GEPA (Genetic-Pareto). To solve this optimization problem efficiently, we consider GEPA (Agrawal et al., 2025), a state-of-the-art evolutionary framework. GEPA conceptualizes prompt optimization as a genetic process driven by language-based reflection. Instead of random mutations, it employs a "Reflection LLM" to analyze error traces from the current prompt's performance and generate targeted semantic mutations that address specific failure modes. Furthermore, GEPA maintains a population of diverse prompts using a Pareto-based selection mechanism, optimizing for multiple objectives (e.g., accuracy and conciseness) simultaneously. This allows the framework to navigate the discrete prompt space effectively, escaping local optima that trap conventional prompt optimization methods.

3. P^{2}O: Joint Policy and Prompt Optimization

3.1. Overview of P^{2}O Framework

We propose P^{2}O, a framework designed to overcome the exploration bottleneck in RLVR. Standard RL methods often fail on "hard samples"—instances where the current policy yields zero successful trajectories—leading to vanishing gradients. P^{2}O addresses this by treating the prompt template as a dynamic latent variable that is jointly optimized with the policy parameters.

As illustrated in Figure 2, P^{2}O operates as an alternating maximization procedure comprising two distinct phases: (1) **Policy Optimization with Context Distillation**, where the model internalizes the reasoning capabilities elicited by optimized prompts; and (2) **Evolutionary Prompt Optimization**, where a genetic algorithm discovers new prompts to crack the remaining hard samples.

3.2. Problem Formulation

We build upon the RLVR setting defined in Section 2.1. The standard objective is to maximize the expected reward $J_{\text{RL}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta}[r(x, y)]$. However, in complex reasoning tasks, the solution space is vast and the reward signal is sparse.

The Exploration Bottleneck. We formally define a subset of the data, $\mathcal{D}_{\text{hard}} \subset \mathcal{D}$, as "hard samples" where the current policy π_θ fails to discover almost any successful trajectory within a reasonable computational budget. Specifically, for $x \in \mathcal{D}_{\text{hard}}$, the expected reward is nearly zero: $\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}[r(x, y)] \approx 0$. Under standard policy gradient methods, the gradient for these samples vanishes:

$$\nabla_\theta J(x) \approx \mathbb{E}_{y\sim\pi_\theta}[\underbrace{(r(x, y) - b)}_{\approx 0} \nabla_\theta \log \pi_\theta(y|x)] \approx 0 \tag{2}$$

where b is a baseline function (e.g., a value function $V(x)$ or moving average) used for variance reduction. Since both the realized reward r and the baseline b approach zero for hard samples, the model ceases to learn from $\mathcal{D}_{\text{hard}}$, leading to sample inefficiency and sub-optimal convergence.

Prompt as a Latent Variable. To tackle this, we introduce a set of auxiliary prompt templates $\mathcal{Z}$ as discrete *latent variables*. We posit that for a hard sample $x \in \mathcal{D}_{\text{hard}}$, while the direct mapping $x \to y^*$ is inaccessible to the current policy, there exists a latent instruction $z \in \mathcal{Z}$ such that the augmented input $\tilde{x} = \mathcal{T}(x, z)$ (where $\mathcal{T}$ is a template insertion function) significantly increases the density of successful trajectories.

Therefore, we reformulate the optimization problem. Instead of optimizing θ solely on the raw input x, we seek to maximize a joint objective over the policy parameters θ and the latent template space $\mathcal{Z}$:

$$\max_{\theta, \mathcal{Z}} \mathcal{J}_{\text{joint}} = \sum_{x\in\mathcal{D}} \max_{z\in\mathcal{Z}\cup\{\epsilon\}} \mathbb{E}_{y\sim\pi_\theta(\cdot|\mathcal{T}(x,z))}[r(x, y)] \tag{3}$$

where ϵ denotes the empty template (original input). This formulation implies a bilevel optimization challenge: we must identify the optimal latent prompt z^* to unlock the gradient for x, and simultaneously update θ to maximize the likelihood of high-reward trajectories. Since directly optimizing the discrete $\mathcal{Z}$ and continuous θ is intractable, P^{2}O decouples this into an iterative alternating maximization process.

3.3. Phase 1: Policy Optimization with Context Distillation

In this phase, we fix the template set $\mathcal{Z}$ and update the policy parameters θ. A naïve approach would be to train the

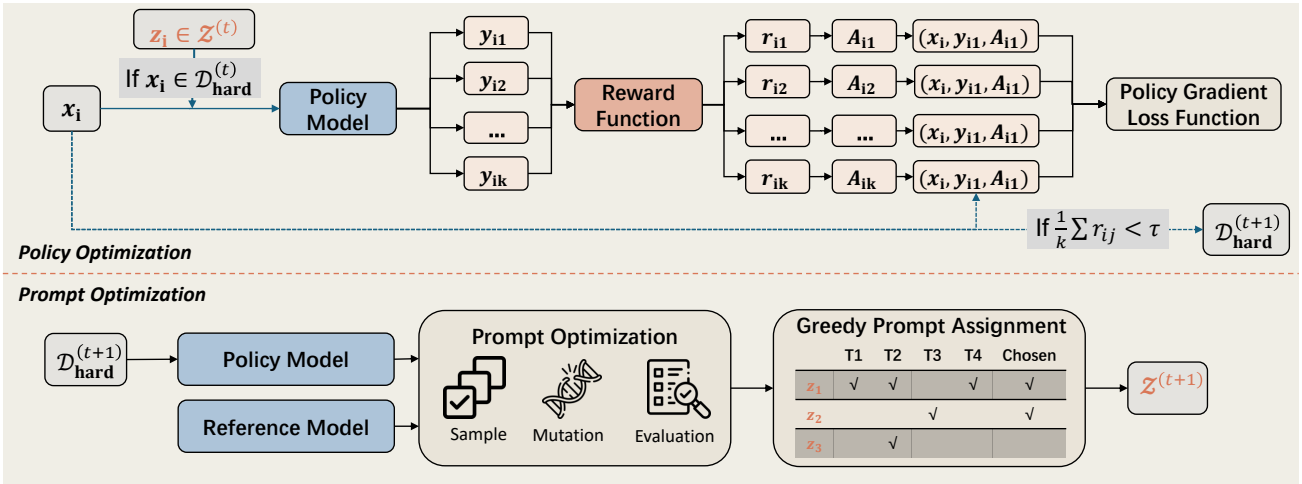

Figure 2. **Overview of the P^{2}O Framework.** The training process is formulated as an alternating maximization procedure between two phases: (1) Policy Optimization with Context Distillation, where the policy π_θ is updated to internalize reasoning patterns elicited by augmented inputs $\tilde{x}$; and (2) Evolutionary Prompt Optimization, where the prompt template set $\mathcal{Z}$ is evolved using GEPA to discover successful trajectories for the remaining hard samples ($\mathcal{D}_{\text{hard}}$).

policy to map the augmented input $\tilde{x}$ to the correct output y. However, this creates a dependency on inference-time prompting. Instead, we employ Context Distillation (Snell et al., 2022) to transfer the reasoning capabilities triggered by z directly into the parameters of π_θ for the original input x.

Hard Sample Mining. At epoch t, we dynamically identify the set of hard samples $\mathcal{D}_{\text{hard}}^{(t+1)}$. For each query x_i, we perform K rollouts. We define a hardness metric based on the empirical success rate:

$$x_i \in \mathcal{D}_{\text{hard}}^{(t+1)} \iff \frac{1}{K}\sum_{k=1}^{K} r(x_i, y_{ik}) < \tau \tag{4}$$

where τ is a threshold (typically nearly zero). These hard samples are collected to serve as the seed data for the subsequent Prompt Optimization phase.

Trajectory Augmentation & Distillation. For samples identified as hard in the immediately preceding epoch, standard exploration is insufficient. We leverage the template set $\mathcal{Z}^{(t)}$ evolved in the immediately preceding epoch. For a hard sample $x \in \mathcal{D}_{\text{hard}}^{(t)}$, we sample a template $z \sim \mathcal{Z}^{(t)}$ and generate trajectories using the augmented context $\tilde{x} = \mathcal{T}(x, z)$.

Crucially, while the generation is conditioned on $\tilde{x}$, the policy update is calculated with respect to the *original* input x. Let $\tilde{y}$ be an improved trajectory generated via prompt augmentation. The gradient update is computed as:

$$\nabla_\theta J \approx \frac{1}{N}\sum_{i=1}^{N}\sum_{\tilde{y}\in\tilde{\mathcal{Y}}} A(x,\tilde{y})\nabla_\theta \log \pi_\theta(\tilde{y}|x) \tag{5}$$

where $\tilde{\mathcal{Y}}$ denotes the set of improved trajectories with input $\tilde{x}$. By decoupling the rollout context ($\tilde{x}$) from the gradient context (x), P^{2}O forces the model to learn the reasoning path y as an intrinsic capability, independent of the auxiliary prompt z. This effectively distills the "teacher" distribution $\pi(\cdot|\tilde{x})$ into the "student" distribution $\pi(\cdot|x)$.

3.4. Phase 2: Evolutionary Prompt Optimization

In the second phase, we fix the policy π_θ and optimize the template set $\mathcal{Z}^{(t+1)}$ to address the newly identified hard set $\mathcal{D}_{\text{hard}}^{(t+1)}$. We employ GEPA (Algorithm 3) for this discrete optimization task.

Reflective Evolution. Unlike traditional genetic algorithms that rely on random mutation, GEPA utilizes the reference model π_{init} as a mutation operator to "perform gradient descent in semantic space" (Yang et al., 2024). For each template z in the current Pareto front $\mathcal{Z}_{\text{front}}$, we sample a mini-batch $\mathcal{D}_{\text{mini}}$ from $\mathcal{D}_{\text{hard}}^{\text{train}}$ and generate error feedback $\mathcal{F}$ containing failed predictions and ground truth. The mutation step then produces an improved candidate:

$$z' \leftarrow \pi_{\text{init}}(\text{"Propose Improvement"} \mid z, \mathcal{F}) \tag{6}$$

where "Propose Improvement" denotes the meta-prompt for proposing new candidate templates. We adopt the same meta-prompt as in Agrawal et al. (2025). Candidates that demonstrate improvement on $\mathcal{D}_{\text{mini}}$ are evaluated on the development set $\mathcal{D}_{\text{hard}}^{\text{dev}}$ and added to the template pool $\mathcal{Z}$.

Pareto Selection. Using only the best template is insufficient to cover the diversity of failure modes in reasoning tasks. GEPA maintains a population of templates and employs Pareto optimization on $\mathcal{D}_{\text{hard}}^{\text{dev}}$ to identify nondominated

Algorithm 1 P^{2}O

Require: Training Data $\mathcal{D}$, Initial Template Set $\mathcal{Z}^{(0)} = \emptyset$
Require: Policy Model π_θ, Reference Model π_{init}
Require: Epochs N_{epoch}, Rollout Num K, Threshold τ
Ensure: Optimized Policy π_{θ^*}
for $t = 0$ **to** $N_{\text{epoch}} - 1$ **do**
  Initialize next hard set $\mathcal{D}_{\text{hard}}^{(t+1)} \leftarrow \emptyset$
  // Phase 1: Policy Optimization
  for all batch $(x_i, \hat{y}_i) \in \mathcal{D}$ **do**
    for $k = 1$ **to** K **do**
      $x_{ik} \leftarrow x_i$
      if $x_i \in \mathcal{D}_{\text{hard}}^{(t)}$ **and** $\mathcal{Z}^{(t)} \neq \emptyset$ **then**
        Sample template $z \sim \mathcal{Z}^{(t)}$
        $x_{ik} \leftarrow \mathcal{T}(x_{ik}, z)$
        // $\mathcal{T}$ is a template insertion function
      end if
      Sample $y_{ik} \sim \pi_\theta(\cdot|x_{ik})$
      $r_{ik} \leftarrow \textsc{RewardFunc}(y_{ik}, \hat{y}_i)$
    end for
    $\bar{r}_i \leftarrow \frac{1}{K} \sum_{k=1}^{K} r_{ik}$
    if $\bar{r}_i < \tau$ **then**
      $\mathcal{D}_{\text{hard}}^{(t+1)} \leftarrow \mathcal{D}_{\text{hard}}^{(t+1)} \cup \{x_i\}$
    end if
    Compute advantages A_{ik}
    Update π_θ using $\{x_i, y_{ik}, A_{ik}\}$
  end for
  // Phase 2: Prompt Optimization
  if $\mathcal{D}_{\text{hard}}^{(t+1)} \neq \emptyset$ **then**
    $\mathcal{Z}^{(t+1)} \leftarrow \textsc{Gepa}(\mathcal{D}_{\text{hard}}^{(t+1)}, \pi_\theta, \pi_{\text{init}})$ *// Alg. 3*
  else
    $\mathcal{Z}^{(t+1)} \leftarrow \emptyset$
  end if
end for

candidates (Algorithm 4). This approach ensures that $\mathcal{Z}$ remains diverse and generalizable across different error patterns. After evolution, we apply a greedy set cover strategy (Algorithm 2) to select a minimal Pareto set $\mathcal{Z}_{\text{covered}}$ and assign sample-specific templates to each instance in $\mathcal{D}_{\text{hard}}$ for the next training epoch. This mechanism enables targeted intervention while maintaining template diversity.

The complete training procedure is summarized in Algorithm 1. By iteratively refining the policy to absorb prompt-induced capabilities and evolving prompts to target the policy's remaining weaknesses, P^{2}O establishes a virtuous cycle of self-improvement.

Algorithm 2 GREEDYPROMPTASSIGNMENT

Require: Template set with scores $\mathcal{Z} = \{(z_1, (r_{1,1}, \ldots, r_{1,N})), \ldots, (z_M, (r_{M,1}, \ldots, r_{M,N}))\}$, Hard samples $\mathcal{D}_{\text{hard}}$, Rollout Num K *// Each $(r_{i,1}, \ldots, r_{i,N})$ contains scores of template z_i on N dev samples*
Ensure: Sample-specific template assignments $\{(x_j, \mathbf{z}_j)\}$
// Step 1: Greedy set cover to find minimal Pareto set
Initialize Template Pareto Set $\mathcal{Z}_{\text{covered}} \leftarrow \emptyset$ and Covered Sample Set $\mathcal{S}_{\text{covered}} \leftarrow \emptyset$
while TRUE **do**
  Find $z^* \leftarrow \arg\max_{z_i \notin \mathcal{Z}_{\text{covered}}} |\mathcal{C}(z_i) \setminus \mathcal{S}_{\text{covered}}|$
  // $\mathcal{C}(z_i) = \{x_j \mid r_{i,j} = 1\}$ is the set of dev samples solved by z_i
  if $|\mathcal{C}(z^*) \setminus \mathcal{S}_{\text{covered}}| = 0$ **then**
    break
  end if
  $\mathcal{Z}_{\text{covered}} \leftarrow \mathcal{Z}_{\text{covered}} \cup \{z^*\}$
  $\mathcal{S}_{\text{covered}} \leftarrow \mathcal{S}_{\text{covered}} \cup \mathcal{C}(z^*)$
end while
// Step 2: Assign K templates to each hard sample
Compute weights: $\bar{r}_i \leftarrow \frac{1}{N} \sum_{n=1}^{N} r_{i,n}$ for each $z_i \in \mathcal{Z}_{\text{covered}}$
Initialize assignment: $\mathcal{A} \leftarrow \{\}$
for x_j in $\mathcal{D}_{\text{hard}}$ **do**
  if $x_j \in \mathcal{S}_{\text{covered}}$ **then**
    $\mathcal{Z}_j \leftarrow \{z_i \in \mathcal{Z}_{\text{covered}} \mid x_j \in \mathcal{C}(z_i)\}$
    // Sample from templates that solve this sample
  else
    $\mathcal{Z}_j \leftarrow \mathcal{Z}_{\text{covered}}$
  end if
  Sample $\mathbf{z}_j = (z_{j,1}, \ldots, z_{j,K})$ from $\mathcal{Z}_j$ weighted for K times by $\{\bar{r}_i\}$
  $\mathcal{A} \leftarrow \mathcal{A} \cup \{(x_j, \mathbf{z}_j)\}$
end for
RETURN $\mathcal{A}$

4. Experiments

4.1. Settings

Datasets. We conduct experiments on two distinct datasets, each comprising $N = 5,000$ samples. The first dataset is a subset randomly sampled from DeepScaler (Luo et al., 2025), while the second is derived from DeepMath (He et al., 2025) by selecting samples with a difficulty level of ≥ 7.

Method Configuration. We combine GRPO with a one-step off-policy strategy, excluding the KL divergence penalty. The training hyperparameters include a maximum learning rate of 1×10^{-6}, a global batch size of 128, and a maximum generation length of 12k tokens. During the rollout phase, we utilize a temperature of $T = 0.6$ and sam-

Table 1. **Comparative Evaluation on Mathematical Reasoning Benchmarks.** We report the accuracy (%) of P^{2}O against baselines. P^{2}O consistently outperforms the baselines, achieving the most significant gains on challenging benchmarks such as AIME24 and AIME25. Best results are highlighted in **bold**.

MODEL	AIME24	AIME25	AMC	MATH500	MINERVA	OLYMPIAD	AVG
BASE MODELS							
QWEN3-0.6B	2.1	1.9	30.2	53.8	12.5	19.4	19.6
QWEN3-1.7B	13.1	10.2	47.7	72.8	22.1	38.4	34.1
QWEN3-4B-BASE	9.2	5.8	37.7	67.4	32.4	35.6	31.9
QWEN3-4B	23.8	19.4	68.0	82.8	31.6	49.9	45.9
DEEPMATH-5K							
GRPO	33.8	29.0	79.5	87.6	41.5	57.5	54.8
QWEN3-4B-P^{2}O$_{\text{SELF-REF}}$	**52.1**	**39.8**	**85.3**	**90.2**	**41.9**	**60.9**	**61.7**
QWEN3-4B-P^{2}O$_{\text{TEACHER-REF}}$	44.6	34.4	81.9	**90.2**	36.0	60.0	57.9
DEEPSCALER-5K							
GRPO	46.9	37.7	88.1	90.4	**41.5**	58.2	60.5
QWEN3-4B-P^{2}O$_{\text{SELF-REF}}$	51.5	42.1	88.1	91.0	40.1	61.8	62.4
QWEN3-4B-P^{2}O$_{\text{TEACHER-REF}}$	**59.8**	**49.4**	**92.2**	**91.4**	36.4	**62.2**	**65.2**

ple $K = 6$ trajectories per prompt. All models are trained for 5 epochs. We use Qwen3-4B (Yang et al., 2025) as the training backbone. For GEPA, we use a validation set of 300 examples. Unlike the original GEPA setting (Agrawal et al., 2025), we use beam size W to parallelize the Pareto search for improved computational efficiency. The candidate selection beam size is set to $W = 16$ (yielding ~ 40 templates per iteration). Regarding the reflection model in GEPA, we evaluate two variants: P^{2}O$_{\text{Self-Ref}}$, which utilizes the reference model (Qwen3-4B) as the mutation operator, and P^{2}O$_{\text{Teacher-Ref}}$, which employs Kimi-K2 (Team et al., 2025) as a stronger external teacher to provide high-quality feedback and prompt mutations.

Reward. We employ a strict binary reward ($r \in \{0, 1\}$). A reward of 1 is granted solely when the response follows the `\boxed{}` format and matches the ground truth.

Evaluation Protocol. We employ the open-source evaluation suite[1] provided by Qwen for all mathematical benchmarks. To ensure statistical robustness and encourage exploration on smaller datasets (fewer than 100 samples), we report the average performance across 16 rollouts per question, generated with a temperature of 0.6. For larger datasets, we adopt greedy sampling.

4.2. Main Results

Table 1 summarizes the performance of models trained on DeepMath-5K and DeepScaler-5K across six challenging mathematical benchmarks, comparing P^{2}O against Qwen3 base models and the GRPO baselines.

Comparison with Baselines. As shown in Table 1, P^{2}O consistently outperforms both the backbone model and the GRPO baseline across most mathematical benchmarks. On the DeepScaler-5K dataset, our best configuration achieves an average accuracy of 65.2%, surpassing the GRPO baseline by 4.7%. Similar performance gains are observed on the DeepMath-5K dataset, where P^{2}O yields a 6.9% improvement over GRPO, demonstrating robustness across different training distributions.

The benefits of P^{2}O are most pronounced in high-difficulty reasoning tasks that demand extensive exploration. Notably, on the AIME24 and AIME25 benchmarks under the DeepScaler-5K setting, P^{2}O improves upon GRPO by 12.9% and 11.7% respectively. These results empirically validate our hypothesis: prompt evolution bridges the exploration gap for hard samples, recovering vital training signals from instances that would otherwise remain intractable.

Impact of Reflection Models on Prompt Evolution. Our comparison reveals that the optimal reflection source is task-dependent: on DeepScaler-5K, the Teacher-Reflection variant dominates (65.2% vs. 62.4% avg.), whereas on DeepMath-5K, P^{2}O$_{\text{Self-Ref}}$ (61.7% avg.) surprisingly outperforms the Teacher-Reflection variant (57.9% avg.). This divergence suggests that for certain specialized domains, a policy model may benefit more from self-generated refinements that strictly align with its own reasoning capacity, whereas an external teacher might introduce complex priors that are harder for the policy to internalize. Crucially, all P^{2}O variants consistently surpass the GRPO baseline across both datasets, confirming that the structural explo-

[1] https://github.com/QwenLM/Qwen2.5-Math/tree/main/evaluation

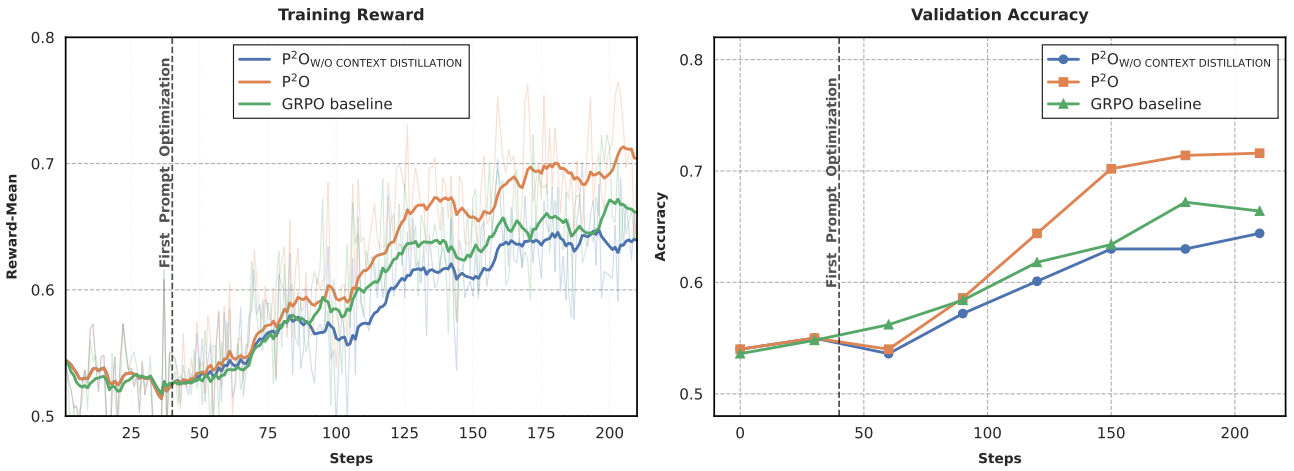

Figure 3. **Training Dynamics of P^{2}O.** The plots illustrate the dynamics of Training Reward (Left) and Validation Accuracy (Right) throughout the optimization process using the Teacher-Ref variant on the DeepScaler-5K dataset. P^{2}O maintains a higher reward compared to the baseline by dynamically cracking hard samples. Crucially, this reward advantage translates into robust gains in validation accuracy, confirming that the context distillation mechanism effectively transfers prompt-dependent success into intrinsic model capability.

ration enabled by GEPA is a robust driver of performance gains regardless of the reflection source.

4.3. Training Dynamics

Analysis of Learning Curves. As shown in Figure 3, the continuous integration of optimized prompts maintains a training reward consistently superior to the baseline. Crucially, this advantage translates into significant gains in validation accuracy, confirming that the model effectively internalizes the elicited reasoning patterns to achieve robust in-distribution generalization.

Analysis of Prompt Optimization Process. Figure 4 elucidates the mechanism driving P^{2}O's performance. As the model's intrinsic capability grows during training, GEPA effectively assists in continuously conquering "hard samples", evidenced by the steady decline in the number of intractable instances (Right). Furthermore, the performance breakdown (Left) reveals that prompt optimization yields consistent gains across both Pass@1 and Pass@6 metrics. This demonstrates that the evolved templates do not merely facilitate a single lucky guess; rather, they robustly enhance the model's solution space, boosting both the deterministic success rate and the broader exploration coverage required for effective distillation.

4.4. Ablations

Importance of Context Distillation. We investigate the necessity of context distillation by ablating the supervision signal. Instead of calculating the policy gradient on the original query x using trajectories generated from the augmented input $\tilde{x} = \mathcal{T}(x, z)$, the ablated model (P^{2}O$_{\text{w/o context distillation}}$) is trained directly on $\tilde{x}$. As shown in Table 2, excluding this component results in a severe performance degradation. Specifically, the average accuracy drops significantly from 65.2% (P^{2}O$_{\text{Teacher-Ref}}$) to 55.6%. Crucially, the ablated model not only fails to match the proposed method's performance but also falls behind the GRPO baseline (60.5%) by 4.9%. This result suggests that training directly on prompt-augmented data leads to a "dependency" effect, where the model learns to rely on the auxiliary templates to solve the task rather than internalizing the reasoning logic. Consequently, when these templates are absent during evaluation, the model fails to generalize. Thus, context distillation is not merely an enhancement but a prerequisite for effectively transferring the capabilities from the teacher reference into the model's intrinsic parameters.

Impact of Group Prompt Diversity. We further investigate the impact of prompt diversity within each rollout group during training. In the standard P^{2}O framework, we utilize a Pareto-based selection strategy and assign a diverse set of templates to each hard sample (i.e., different templates may be used across the K rollouts for the same query). To evaluate the necessity of this diversity, we compare it against a baseline variant, P^{2}O$_{\text{same template in group}}$, where the model samples a single template $z \sim \mathcal{Z}^{(t)}$ per query x_i and applies that fixed template to all K rollouts in the group.

As shown in Table 2, the diversity-driven approach (P^{2}O$_{\text{Teacher-Ref}}$) achieves an average accuracy of 65.2%, outperforming the single-template baseline (64.2%). The gain is particularly pronounced in high-difficulty benchmarks such as AIME24 (+2.5%) and AIME25 (+4.8%). This empirical evidence confirms that employing a diverse set of Pareto-optimal prompts within the same rollout group pro-

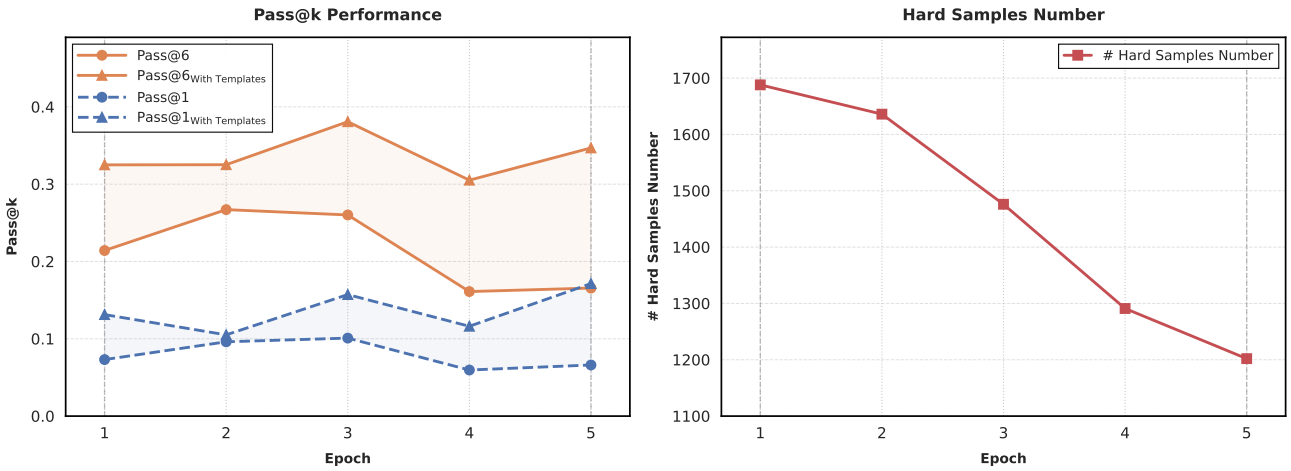

Figure 4. **Prompt Optimization Effectiveness.** *Left:* Optimized prompts (triangles) consistently yield gains over the standard policy (circles) on both Pass@1 and Pass@6 metrics, demonstrating that GEPA effectively assists the model in bridging the performance gap. *Right:* With this assistance, the model continuously conquers hard samples, resulting in a steady decline in the number of intractable instances throughout the training epochs. Data points are derived from the Teacher-Ref variant on the DeepScaler-5K dataset.

Table 2. **Ablation Study of P^{2}O Components.** Comparison of performance on mathematical benchmarks when excluding context distillation or group prompt diversity. Data points are derived from the Teacher-Ref variant on the DeepScaler-5K dataset.

MODEL	AIME24	AIME25	AMC	MATH500	MINERVA	OLYMPIAD	AVG
QWEN3-4B	23.8	19.4	68.0	82.8	31.6	49.9	45.9
GRPO	46.9	37.7	88.1	90.4	**41.5**	58.2	60.5
QWEN3-4B-P^{2}O$_{\text{TEACHER-REF}}$	**59.8**	**49.4**	**92.2**	**91.4**	36.4	**62.2**	**65.2**
QWEN3-4B-P^{2}O$_{\text{W/O CONTEXT DISTILLATION}}$	36.5	31.5	81.1	89.6	40.1	54.8	55.6
QWEN3-4B-P^{2}O$_{\text{SAME TEMPLATE IN GROUP}}$	57.3	44.6	88.9	90.8	40.4	63.0	64.2

vides broader coverage of the reasoning space. By eliciting a wider variety of successful trajectories for a single hard sample, the model receives a denser and more robust supervision signal, which facilitates more effective distillation of complex reasoning capabilities into the model's parameters.

5. Related Works

Reinforcement Learning with Verifiable Rewards (RLVR) and Group Relative Policy Optimization (GRPO) are widely used for LLM reasoning alignment, stabilizing training via group baselines and verification, but still struggle with exploration in sparse or misleading landscapes. To address these scalability and stability issues, recent works have focused on algorithmic advances. For instance, DAPO (Yu et al., 2025) prunes prompts with accuracy equal to 1 or 0 to improve training stability. Despite these improvements, these methods primarily focus on optimizing the policy on a fixed manifold, leaving the initial task prompts untouched.

To further aid the model in traversing complex reasoning paths, various "hint-based" or guidance-augmented strategies have been proposed. Approaches such as strong model guidance (Nath et al., 2025; Liu et al., 2025b), experience replay data (Zhan et al., 2025), and expert solutions (Zhang et al., 2025a; Yan et al., 2025; Li et al., 2025) attempt to scaffold the reasoning process. Similarly, Critique-GRPO (Zhang et al., 2025b) utilizes feedback signals to guide the policy. Crucially, however, these methods typically rely on heuristically obtained hints or expert trajectory fragments derived from the dataset. There is a notable absence of "optimization" regarding the guidance itself; the prompts or hints are treated as static constants rather than dynamic variables. This reliance on predefined or oracle-dependent guidance limits the model's ability to generalize beyond the scope of the provided hints. In contrast, P^{2}O treats the prompt as an optimizable parameter, jointly evolving the task topology with the policy to achieve bi-level self-improvement.

6. Conclusion

In this paper, we proposed P^{2}O, a framework that synergizes genetic prompt evolution with reinforcement learning to address the exploration bottleneck in reasoning tasks. By leveraging optimized prompts to unlock successful trajectories

for hard samples and subsequently internalizing these capabilities through context distillation, P^2O effectively recovers valid training signals for previously intractable instances, thereby preventing the policy from collapsing into local optima. Extensive experiments on mathematical benchmarks demonstrate that P^2O significantly outperforms standard GRPO baselines, validating the effectiveness of jointly optimizing the prompt space and parameter space to achieve robust, autonomous self-improvement.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning, specifically in the area of reinforcement learning and reasoning post-training for Large Language Models. Our proposed method, P^2O, aims to automate the discovery of effective reasoning strategies to assist policy model training, thereby potentially reducing the barrier to developing capable reasoning models. There are many potential societal consequences of our work, primarily related to the general advancement of AI reasoning capabilities and their downstream applications. However, we feel that none of these consequences must be specifically highlighted here beyond the standard ethical considerations of the field.

References

Agrawal, L. A., Tan, S., Soylu, D., Ziems, N., Khare, R., Opsahl-Ong, K., Singhvi, A., Shandilya, H., Ryan, M. J., Jiang, M., et al. Gepa: Reflective prompt evolution can outperform reinforcement learning. *arXiv preprint arXiv:2507.19457*, 2025.

Bercovich, A., Levy, I., Golan, I., Dabbah, M., El-Yaniv, R., Puny, O., Galil, I., Moshe, Z., Ronen, T., Nabwani, N., et al. Llama-nemotron: Efficient reasoning models. *arXiv preprint arXiv:2505.00949*, 2025.

Blakeman, A., Grattafiori, A., Basant, A., Gupta, A., Khattar, A., Renduchintala, A., Vavre, A., Shukla, A., Bercovich, A., Ficek, A., et al. Nemotron 3 nano: Open, efficient mixture-of-experts hybrid mamba-transformer model for agentic reasoning. *arXiv preprint arXiv:2512.20848*, 2025.

Fernando, C., Banarse, D., Michalewski, H., Osindero, S., and Rocktäschel, T. Promptbreeder: Self-referential self-improvement via prompt evolution. *arXiv preprint arXiv:2309.16797*, 2023.

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

He, Z., Liang, T., Xu, J., Liu, Q., Chen, X., Wang, Y., Song, L., Yu, D., Liang, Z., Wang, W., et al. Deepmath-103k: A large-scale, challenging, decontaminated, and verifiable mathematical dataset for advancing reasoning. *arXiv preprint arXiv:2504.11456*, 2025.

Jaech, A., Kalai, A., Lerer, A., Richardson, A., El-Kishky, A., Low, A., Helyar, A., Madry, A., Beutel, A., Carney, A., et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

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

Li, J., Lin, H., Lu, H., Wen, K., Yang, Z., Gao, J., Wu, Y., and Zhang, J. Questa: Expanding reasoning capacity in llms via question augmentation. *arXiv preprint arXiv:2507.13266*, 2025.

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

Liu, Z., Gong, C., Fu, X., Liu, Y., Chen, R., Hu, S., Zhang, S., Liu, R., Zhang, Q., and Tu, D. Ghpo: Adaptive guidance for stable and efficient llm reinforcement learning. *arXiv preprint arXiv:2507.10628*, 2025b.

Luo, M., Tan, S., Wong, J., Shi, X., Tang, W. Y., Roongta, M., Cai, C., Luo, J., Li, L. E., Popa, R. A., and Stoica, I. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl, 2025. Notion Blog.

Nath, V., Lau, E., Gunjal, A., Sharma, M., Baharte, N., and Hendryx, S. Adaptive guidance accelerates reinforcement learning of reasoning models. *arXiv preprint arXiv:2506.13923*, 2025.

Opsahl-Ong, K., Ryan, M. J., Purtell, J., Broman, D., Potts, C., Zaharia, M., and Khattab, O. Optimizing instructions and demonstrations for multi-stage language model programs. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N. (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 9340–9366, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.525. URL https://aclanthology.org/2024.emnlp-main.525/.

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

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

Snell, C., Klein, D., and Zhong, R. Learning by distilling context. *arXiv preprint arXiv:2209.15189*, 2022.

Song, Y., Kempe, J., and Munos, R. Outcome-based exploration for llm reasoning. *arXiv preprint arXiv:2509.06941*, 2025.

Team, K., Bai, Y., Bao, Y., Chen, G., Chen, J., Chen, N., Chen, R., Chen, Y., Chen, Y., Chen, Y., et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

Yan, J., Li, Y., Hu, Z., Wang, Z., Cui, G., Qu, X., Cheng, Y., and Zhang, Y. Learning to reason under off-policy guidance. *arXiv preprint arXiv:2504.14945*, 2025.

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Yang, C., Wang, X., Lu, Y., Liu, H., Le, Q. V., Zhou, D., and Chen, X. Large language models as optimizers. In *ICLR*, 2024. URL https://openreview.net/forum?id=Bb4VGOWELI.

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

Yuksekgonul, M., Bianchi, F., Boen, J., Liu, S., Lu, P., Huang, Z., Guestrin, C., and Zou, J. Optimizing generative ai by backpropagating language model feedback. *Nature*, 639:609–616, 2025.

Zelikman, E., Wu, Y., Mu, J., and Goodman, N. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488, 2022.

Zhan, R., Li, Y., Wang, Z., Qu, X., Liu, D., Shao, J., Wong, D. F., and Cheng, Y. Exgrpo: Learning to reason from experience. *arXiv preprint arXiv:2510.02245*, 2025.

Zhang, X., Huang, Z., Li, Y., Ni, C., Chen, J., and Oymak, S. Bread: Branched rollouts from expert anchors bridge sft & rl for reasoning. *arXiv preprint arXiv:2506.17211*, 2025a.

Zhang, X., Sun, H., Zhang, Y., Feng, K., Lu, C., Yang, C., and Meng, H. Critique-grpo: Advancing llm reasoning with natural language and numerical feedback. *arXiv preprint arXiv:2506.03106*, 2025b.

A. Details on GEPA

We provide the comprehensive pseudocode for the Genetic-Pareto (GEPA) prompt optimization process used in Phase 2 of our framework. Please refer to Algorithm 3 for the detailed execution of the main evolutionary loop, and Algorithm 4 for the specific implementation of the Pareto-based frontier selection strategy.

Algorithm 3 GEPA

Require: Hard Data $\mathcal{D}_{\text{hard}}$, Policy Model π_θ, Reference Model π_{init}
Require: Budget C_{total}, Mini-batch B, Width W
Ensure: Output Template Set $\mathcal{Z}$
Split $\mathcal{D}_{\text{hard}}$ into $\mathcal{D}_{\text{hard}}^{\text{train}}$ and $\mathcal{D}_{\text{hard}}^{\text{dev}}$
Initialize $\mathcal{Z} \leftarrow \{(\epsilon, \text{EVAL}(\pi_\theta, \epsilon, \mathcal{D}_{\text{hard}}^{\text{dev}}))\}$
// ϵ means empty template.
// EVAL() will return reward of every sample in the given dataset by applying the given template
$C_{\text{left}} \leftarrow C_{\text{total}}$
while $C_{\text{left}} > 0$ **do**
  $\mathcal{Z}_{\text{front}} \leftarrow \text{SELECTPARETOFRONT}(\mathcal{Z}, W)$
  for all $z \in \mathcal{Z}_{\text{front}}$ **do**
    Sample mini-batch $\mathcal{D}_{\text{mini}} \subset \mathcal{D}_{\text{hard}}^{\text{train}}$ of size B
    $\bar{r}_{\text{old}} \leftarrow \text{mean}(\text{EVAL}(\pi_\theta, z, \mathcal{D}_{\text{mini}}))$
    Generate error feedback $\mathcal{F}$ from rollouts and rewards on $\mathcal{D}_{\text{mini}}$
    $z' \leftarrow \pi_{\text{init}}(\text{“Propose Improvement”}, z, \mathcal{F})$
    $\bar{r}_{\text{new}} \leftarrow \text{mean}(\text{EVAL}(\pi_\theta, z', \mathcal{D}_{\text{mini}}))$
    Update Cost: $C_{\text{left}} \leftarrow C_{\text{left}} - 2B$
    if $\bar{r}_{\text{new}} > \bar{r}_{\text{old}}$ **then**
      $\mathcal{Z} \leftarrow \mathcal{Z} \cup \{(z', \text{EVAL}(\pi_\theta, z', \mathcal{D}_{\text{hard}}^{\text{dev}}))\}$
      $C_{\text{left}} \leftarrow C_{\text{left}} - |\mathcal{D}_{\text{hard}}^{\text{dev}}|$
    end if
  end for
end while
$\mathcal{Z} \leftarrow \text{GREEDYPROMPTASSIGNMENT}(\mathcal{Z}, \mathcal{D}_{\text{hard}})$
RETURN $\mathcal{Z}$

B. Case Study

To further demonstrate how P^{2}O overcomes the exploration bottleneck, we analyze a representative “hard sample” from the training process in Figure 5. The problem asks for the radius of the smallest sphere enclosing four unit spheres.

Failure of the Base Policy. Without guidance, the model falls into a common cognitive trap: it defaults to a visually intuitive but mathematically suboptimal configuration. As shown in the generated trace, the model attempts to place “3 spheres on a plane... with the 4th sphere resting on top.” This corresponds to a localized packing that yields a parent sphere radius of $1 + \sqrt{2}$, failing to minimize the volume.

Mechanism of the Optimized Prompt. The template evolved by GEPA acts as a targeted intervention in the model’s latent search space. It does not merely encourage the model to “think harder”; rather, it injects specific domain knowledge—specifically the concept of the regular tetrahedron and the precise mathematical constant for its centroid-to-vertex distance ($\sqrt{3/8}$). This prompt effectively prunes the invalid search space (planar configurations) and steers the reasoning trajectory toward the global optimum (tetrahedral packing), allowing the model to derive the correct radius of $1 + \frac{\sqrt{6}}{2}$.

Algorithm 4 SELECTPARETOFRONT

Require: Template set with scores $\mathcal{Z} = \{(z_1, (r_{1,1}, \ldots, r_{1,N})), \ldots, (z_M, (r_{M,1}, \ldots, r_{M,N}))\}$, Width W
// Each $(r_{i,1}, \ldots, r_{i,N})$ contains scores of template z_i on N dev samples
Ensure: Selected templates $\mathcal{Z}_{\text{front}}$
// Step 1: Identify Pareto-optimal templates
$\mathcal{Z}_{\text{front}} \leftarrow \emptyset$
for $i = 1$ **to** M **do**
  dominated $\leftarrow$ FALSE
  for $j = 1$ **to** M **do**
    if $j \neq i$ **and** z_j dominates z_i **then**
      // z_j dominates z_i if $\exists n : r_{j,n} > r_{i,n}$ and $\forall n' : r_{j,n'} \geq r_{i,n'}$
      dominated $\leftarrow$ TRUE
      break
    end if
  end for
  if not dominated **then**
    $\mathcal{Z}_{\text{front}} \leftarrow \mathcal{Z}_{\text{front}} \cup \{(z_i, (r_{i,1}, \ldots, r_{i,N}))\}$
  end if
end for
// Step 2: Select W templates from Pareto front
if $|\mathcal{Z}_{\text{front}}| \leq W$ **then**
  $\mathcal{Z}_{\text{front}} \leftarrow \{z \mid (z, (r_1, \ldots, r_N)) \in \mathcal{Z}_{\text{front}}\}$
else
  // Sample based on mean dev scores
  Compute weights: $\bar{r}_i \leftarrow \frac{1}{N} \sum_{n=1}^{N} r_{i,n}$ for each $(z_i, (r_{i,1}, \ldots, r_{i,N})) \in \mathcal{Z}_{\text{front}}$
  Sample W templates from $\mathcal{Z}_{\text{front}}$ weighted by $\{\bar{r}_i\}$
  $\mathcal{Z}_{\text{front}} \leftarrow \{\text{sampled templates}\}$
end if
RETURN $\mathcal{Z}_{\text{front}}$

Question

What is the radius of the smallest sphere in which 4 spheres of radius 1 will fit?

Response w/o template

To find the radius of the smallest sphere containing 4 unit spheres, I need to determine the the optimal packing.
A natural approach is to place 3 spheres on a plane forming an equilateral triangle, with the 4th sphere resting on top.

[Omitted]

Final Answer: $\sqrt{2}+1$

Template

You are a concise math solver.
For every problem:
1. Think step-by-step, but **do not expose your reasoning** to the user.

[Omitted]

Special facts you must know:
- Consider configurations where objects are placed at vertices of regular polyhedra (tetrahedron,.etc.)
- For a regular tetrahedron with edge length a, the distance from its centroid to any vertex involves the ratio $\sqrt{3/8}$.

[Omitted]

Response with template

To find the smallest sphere containing 4 unit spheres
I should consider symmetric configurations. The template suggests placing objects at vertices of regular polyhedra.
For 4 objects, the optimal regular polyhedron is a regular tetrahedron

[Omitted]

From the template: "For a regular tetrahedron with edge length a, the distance from its centroid to any vertex involves the ratio $\sqrt{3/8}$."
Applying this: distance = $\sqrt{3/8} \times a = \sqrt{3/8} \times 2 = \sqrt{3/2} = \sqrt{6}/2$

[Omitted]

Final Answer: $\frac{\sqrt{6}+2}{2}$

Figure 5. Qualitative Analysis: Overcoming Local Optima in Geometric Reasoning.