Heterogeneous Agent Collaborative Reinforcement Learning

Zhixia Zhang[1]* **Zixuan Huang**[1,2]* **Gongxun Li**[1] **Huaiyang Wang**[1] **Chengyi Yuan**[5]
Xin Xia[2] **Deqing Wang**[1] **Fuzhen Zhuang**[1]† **Shuai Ma**[1] **Ning Ding**[3]
Yaodong Yang[4] **Yikun Ban**[1]‡

[1]***Beihang University*** [2]***Bytedance China***
[3]***Tsinghua University*** [4]***Peking University*** [5]***Apple***

code: `https://github.com/zzx-peter/HACRL-code`

Abstract

We introduce **H**eterogeneous **A**gent **C**ollaborative **R**einforcement **L**earning (**HACRL**), a new Reinforcement Learning from Verifiable Reward (RLVR) problem that addresses the inefficiencies of isolated multi-agent on-policy optimization. HACRL targets model portfolios that serve a shared task distribution under different latency, hardware, and cost constraints: heterogeneous agents share verified rollouts during training to mutually improve, while operating independently at inference time. Building on this problem, we propose **HACPO**, a collaborative RL algorithm that enables principled rollout sharing to maximize sample utilization and cross-agent knowledge transfer. To mitigate capability discrepancies and policy distribution shifts, HACPO introduces four tailored mechanisms with theoretical guarantees for the oracle baseline and finite-batch capability ratio estimation. Extensive experiments across diverse heterogeneous model combinations and reasoning benchmarks show that HACPO consistently improves all participating agents, outperforming GSPO with double rollouts by an average of 3.6% while using only half the rollout cost. A three-agent experiment spanning three model families further demonstrates the effectiveness of heterogeneous collaboration.

1 Introduction

Reinforcement Learning with Verifiable Rewards (RLVR) trains reasoning models using automatically checkable reward signals, such as unit tests and formal verifiers Yang et al. [2026], Ban et al. [2026]. Group-based algorithms such as GRPO Shao et al. [2024] replace the critic in PPO Schulman et al. [2017] with group-relative advantages, motivating variants including DAPO Yu et al. [2025] and GSPO Zheng et al. [2025]. Despite these advances, RLVR remains bottlenecked by expensive on-policy sampling and verification, which frequently dominate the overall training overhead and limit scalability.

Our intended setting is a *heterogeneous model portfolio*: multiple models serve a shared task distribution under different latency, hardware, cost, and answer-quality requirements. For example, a mathematical problem-solving platform may maintain a small model for latency-sensitive requests

*Equal contribution. Feel free to contact `zhixiaz@buaa.edu.cn` for details of the paper.
†School of Artificial Intelligence, Beihang University.
‡Corresponding author: `yikunb@buaa.edu.cn`. State Key Laboratory of Complex & Critical Software Environment, Beihang University.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

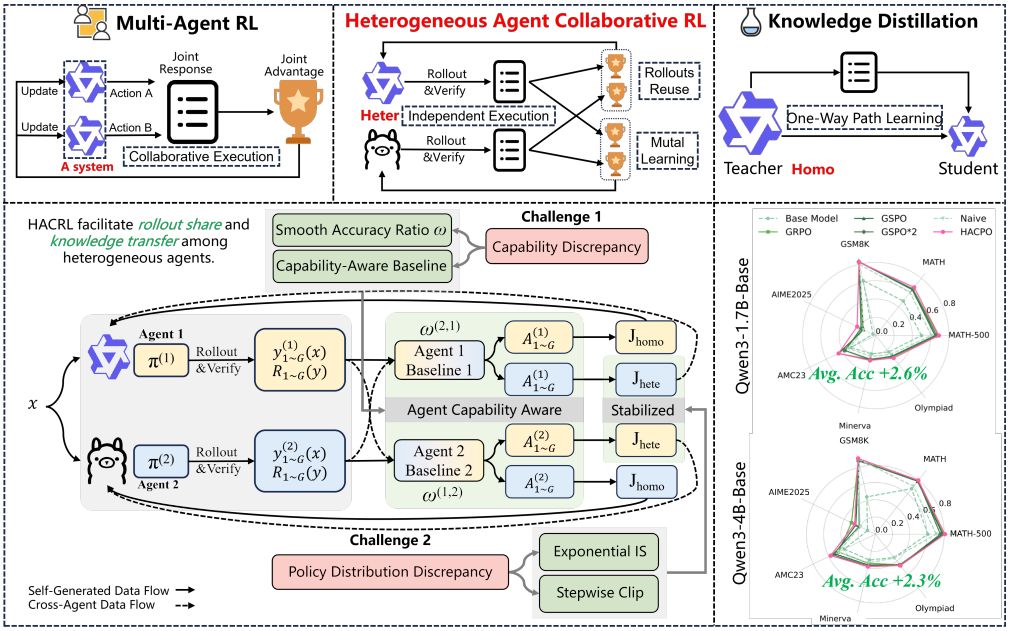


Figure 1: In HACPO, shared rollouts from multiple heterogeneous agents are leveraged for collaborative training. Built upon vanilla RL Optimization, HACPO introduces four algorithmic innovations to mitigate capability and policy distribution discrepancy.

and a larger model for requests requiring higher accuracy. Separate client deployments may also use different model families Yang et al. [2025a], Grattafiori et al. [2024] to fit their existing hardware and software interfaces. These operating constraints motivate improving every model in the portfolio. *Independent deployment* means that each inference instance invokes a single model, without communication with or access to its training partners.

Under isolated RLVR training, these models repeatedly generate and verify rollouts for the same task, but use them only for self-training. This motivates the question: *can heterogeneous models improve together by reusing one another's verified rollouts while retaining independent deployment?* The aim is mutual benefit across the portfolio, while improving the utilization efficiency of rollout.

We first formalize this setting as **H**eterogeneous **A**gent **C**ollaborative **R**einforcement **L**earning (**HACRL**), which captures collaborative policy optimization among heterogeneous agents that execute independently at inference time. HACRL differs fundamentally from existing paradigms as illustrated in Figure 1: (1) **LLM-based Multi-Agent Reinforcement Learning (MARL).** Liao et al. [2025] MARL trains agents to coordinate and jointly solve tasks through interaction within a coupled multi-agent system. In contrast, HACRL does not require coordinated execution. In many practical scenarios, only a single agent is deployed at inference time; however, we still desire that this agent benefits from knowledge acquired from other agents during training. (2) **On-/Off-Policy Distillation.** Distillation typically follows a one-directional "teacher-to-student" paradigm, often among homogeneous agents. HACRL instead enables bidirectional **mutual learning** among **heterogeneous agents**, where each agent simultaneously acts as both a knowledge provider and a learner.

We then propose **H**eterogeneous **A**gent **C**ollaborative **P**olicy **O**ptimization (**HACPO**) to solve HACRL (Figure 1). Compared to vanilla RL optimization, HACPO improves training in two critical aspects: **(1) Maximized Sample Utilization.** In an n-agent system, each rollout can be reused up to n times, substantially improving sample efficiency. **(2) Bidirectional Knowledge Transfer.** By learning from one another, agents acquire complementary knowledge unavailable through self-learning alone, enabling all agents to break performance bottlenecks.

In this work, our contributions can be summarized as:

[Problem Definition]. We formulate HACRL as a collaborative policy optimization problem for heterogeneous agents under RLVR, aiming to achieve mutual benefit through cross-agent rollout reuse while controlling distribution shifts caused by heterogeneity.

[Algorithm]. We propose HACPO to address this problem, with four modifications: **(1)** Agent-Capability-Aware Advantage Estimation, **(2)** Model Capabilities Discrepancy Coefficient, **(3)** Exponential Importance Sampling, and **(4)** Stepwise Clipping. These tailored techniques enable the agents to engage in effective and stable mutual learning.

[Performance]. We evaluate HACPO across three types of heterogeneity and seven challenging mathematical reasoning benchmarks, demonstrating consistent performance improvements, averaging 3.6%, while utilizing only half the rollout cost, compared to GSPO with double rollouts.

2 Heterogeneous Agent Collaborative Reinforcement Learning

2.1 Heterogeneous LLM Agent Taxonomy

Let π_θ denote a large language model (LLM) agent parameterized by $\theta \in \Theta$, where Θ specifies the complete parameter space, including architecture, dimensionality, and trainable weights. Let V_π denote the output vocabulary of agent π_θ. We consider a collaborative policy optimization setting in which multiple LLM agents are jointly optimized toward a shared objective.

We categorize heterogeneity among distinct LLM agents into three types: (1) heterogeneous state; (2) heterogeneous size; (3) heterogeneous model.

Definition 2.1 (**Heterogeneous State**)**.** Two LLM agents $\pi_\theta^{(1)}$ and $\pi_\theta^{(2)}$ are said to exhibit *heterogeneous state* if $\Theta_1 = \Theta_2$ and $\dim(\theta_1) = \dim(\theta_2)$, but $\theta_1 \neq \theta_2$ at the start of collaborative policy optimization.

Definition 2.2 (**Heterogeneous Size**)**.** Two LLM agents $\pi_\theta^{(1)}$ and $\pi_\theta^{(2)}$ are said to exhibit *heterogeneous size* if they belong to the same model family and share the same architectural design principles, but have different parameter dimensionalities, i.e., $\dim(\theta_1) \neq \dim(\theta_2)$, with $\theta_1 \neq \theta_2$ at the start of collaborative policy optimization.

Definition 2.3 (**Heterogeneous Model**)**.** Given two LLM agents $\pi_\theta^{(1)}$ and $\pi_\theta^{(2)}$, we define them to exhibit *heterogeneous model* heterogeneity if their model architectures differ (e.g., tokenizer, attention mechanism, or training objective), their parameter spaces and sizes are distinct (i.e., $\Theta_1 \neq \Theta_2$), and their initial parameter instantiations are unique (i.e., $\theta_1 \neq \theta_2$).

Remark 2.4. This taxonomy represents increasing degrees of heterogeneity: heterogeneous state differs only in optimization state, heterogeneous size introduces capacity mismatch, and heterogeneous model captures architectural and representational divergence. This hierarchy enables a systematic study of collaborative policy optimization among heterogeneous LLM agents.

2.2 Problem Formalization

We consider the Heterogeneous Agent Collaborative Reinforcement Learning (HACRL) framework with n LLM agents. Each agent $k \in \{1, \ldots, n\}$ is associated with a policy $\pi_\theta^{(k)}$. All agents operate on a shared task distribution $\mathcal{D}$ and exhibit heterogeneity as defined in Section 2.1.

During training step t, for a prompt $x \sim \mathcal{D}$, each agent k independently samples G candidate responses from its policy. The joint response set is:

$$\mathcal{Y}_t^{(k)}(x) = \{y_1^{(k)}, \ldots, y_G^{(k)}\} \sim \pi_\theta^{(k)}(\cdot \mid x), \quad \mathcal{Y}_t(x) = \bigcup_{k=1}^{n} \mathcal{Y}_t^{(k)}(x). \tag{1}$$

Since all agents solve the same task, a shared reward function $R(\cdot)$ is applied to every response. The joint reward set is:

$$\mathcal{R}_t^{(k)}(x) = \{R(y_i^{(k)}) \mid i = 1, \ldots, G\}, \quad \mathcal{R}_t(x) = \bigcup_{k=1}^{n} \mathcal{R}_t^{(k)}(x). \tag{2}$$

Definition 2.5 (**HACRL Problem**)**.** Consider a system of n heterogeneous agents. For a prompt $x \sim \mathcal{D}$, let $\mathcal{Y}(x)$ and $\mathcal{R}(x)$ denote the joint response and reward sets, respectively. The objective of *Heterogeneous Agent Collaborative Reinforcement Learning* is to optimize each agent $k \in \{1, \ldots, n\}$ by maximizing

$$J^{(k)} = J_{\text{homo}}^{(k)}\Big(Y_t^{(k)}(x), \mathcal{R}_t^{(k)}(x)\Big) + J_{\text{hete}}^{(k)}\Big(\{Y_t^{(j)}(x), \mathcal{R}_t^{(j)}(x)\}_{j \neq k}\Big), \tag{3}$$

where $J^{(k)}_{\text{homo}}$ is computed using rollouts generated by agent k itself, and $J^{(k)}_{\text{hete}}$ leverages rollouts generated by the other agents.

This formulation enables each agent to benefit from both self-generated experiences and cross-agent information under collaborative reinforcement learning.

3 Heterogeneous Agent Collaborative Policy Optimization

In this section, we propose HACPO, a novel multi-agent collaborative optimization algorithm (procedure is shown in Appendix E): for *one* given task, *multiple* heterogeneous LLM agents execute independently and learn from each other.

Facing two challenges shown in Figure 1: the discrepancy of *agent capability* and *policy distribution*, Our method addresses the above challenges through the following components: (1) Agent-Capability-Aware Advantage Estimation; (2) Model Capability Discrepancy Coefficient; (3) Exponential Importance Sampling; (4) Stepwise Clipping.

3.1 Agent-Capability-Aware Advantage Estimation

At training step t, for each prompt x, each agent $k \in \{1, ..., n\}$ generates G responses $\{y^{(k)}_{t,i}\}^{G}_{i=1} \sim \pi^{(k)}_{\theta}(\cdot \mid x)$. For a single agent, the standard group-relative advantage estimator Shao et al. [2024] is:

$$A^{(k)}_{\text{single}.t,i}(y^{(k)}_{t,i}) = \frac{R\left(y^{(k)}_{t,i}\right) - \frac{1}{G}\sum_{i=1}^{G} R\left(y^{(k)}_{t,i}\right)}{std\{\mathcal{R}^{(k)}_{t}(x)\}}. \tag{4}$$

While Eq. (4) is appropriate for training a single model in isolation, it becomes suboptimal in a multi-agent settings where agents exhibit heterogeneous capabilities. Relying solely on self-generated responses fails to leverage valuable information from other agents, while *naively averaging rewards across all agents disregards inter-model capability differences and often results in miscalibrated advantage estimates*.

To address this issue, we propose an *agent-capability-aware* advantage estimator. The advantage of response $y^{(k)}_{t,i}$ for agent k is defined as

$$A^{(k)}_{t,i}\left(y^{(k)}_{t,i}\right) = \frac{R\left(y^{(k)}_{t,i}\right) - \mu^{(k)}_{t}}{std\{\mathcal{R}_{t}(x)\}}, \quad \mu^{(k)}_{t} = \frac{1}{nG}\sum_{j=1}^{n}\sum_{i=1}^{G} \omega^{(k,j)}_{t}\, R\left(y^{(j)}_{t,i}\right), \tag{5}$$

Here, $\mu^{(k)}_{t}$ is the capability-adjusted baseline. $\omega^{(k,j)}_{t}$ is a *capability ratio* that rescales responses from agent j when estimating the baseline for agent k, defined as:

$$\omega^{(k,j)}_{t} = \frac{P^{(k)}_{t}}{P^{(j)}_{t}}, \quad P^{(k)}_{t} = \frac{1}{|\mathcal{B}|G}\sum_{x\in\mathcal{B}_t}\sum_{i=1}^{G} R\left(y^{(k)}_{t,i}\right), \tag{6}$$

where the $P^{(k)}_{t}$ denotes an estimate of the performance of agent k, obtained by averaging the mean rewards of the batch $\mathcal{B}_t$ at the current step t.

Intuitively, when estimating the advantage baseline in a group for agent k, rewards from other agents are reweighted according to their relative capabilities, allowing all responses to contribute while preserving agent-specific calibration.

We further analyze the capability-aware adjustment in Appendix D. Since the normalization factor $\text{std}\{\mathcal{R}_t(x)\}$ is known to introduce a common but minor bias in practice, our analysis focuses on the centered reward $\bar{A}^{(k)}_{t,i}(y) = R(y) - \mu^{(k)}_{t}$.

As shown in Appendix D.1, the oracle capability-aware baseline is exactly unbiased: its expectation matches the agent's true expected reward, so the corresponding centered reward has zero mean. The Appendix D.2 further analyzes the practical batch-level empirical ratio and proves uniform high-probability bounds on its estimation error via Hoeffding's inequality and union bounds.

This theoretical guarantee ensures that HACPO can safely incorporate heterogeneous cross-agent rollouts to enrich the learning signal and maximize sample efficiency—while introducing no systematic bias in the oracle case, and only bounded, controllable error in the finite-batch implementation.

3.2 Model Capabilities Discrepancy Coefficient

To address capability discrepancies across heterogeneous agents, we employ the capability ratio $\omega_t^{(k,j)}$, introduced earlier, as a quantitative measure of relative model competence. When training agent k, advantages computed from samples generated by other agents are rescaled according to their relative capability. This design encourages an agent to learn more aggressively from stronger agents, while adopting a more conservative update when incorporating samples from weaker ones.

Formally, suppose that agent k is updated at training step t using a response $y_{t,i}^{(j)}$ generated by agent j. The effective advantage used for updating agent k is defined as

$$\tilde{A}_{t,i}^{(k)}\left(y_{t,i}^{(j)}\right) = \begin{cases} A_{t,i}^{(k)}\left(y_{t,i}^{(j)}\right) & j = k \\ \omega_t^{(j,k)} \cdot A_{t,i}^{(k)}\left(y_{t,i}^{(j)}\right) & j \neq k \end{cases} \tag{7}$$

Here, $\omega_t^{(k,j)}$ represents the performance ratio between agents k and j at training step t, with larger values indicating that agent k outperforms agent j.

The capability ratio $\omega_t^{(k,j)}$ serves two complementary roles to enable stable collaboration:

(1) Baseline Calibration: In Sec. 3.1, it rescales cross-agent reward statistics to properly calibrate the baseline $\mu_t^{(k)}$.

(2) Gradient Modulation: In Eq. (7), it acts as a modulation factor that amplifies learning signals from stronger agents while attenuating those from weaker ones.

3.3 Exponential Importance Sampling

Importance sampling is commonly used to correct distributional mismatches between samples generated by different policies. Following GSPO, we adopt a sequence-level importance ratio and extend it to the heterogeneous multi-agent setting. When updating agent k at step t, for a response $y_{t,i}^{(j)}$ generated by agent j, we define

$$s_{t,i}^{(k,j)} = \frac{\pi_\theta^{(k)}\left(y_{t,i}^{(j)}\right)^{\frac{1}{L_k}}}{\pi_{\theta_{\text{old}}}^{(j)}\left(y_{t,i}^{(j)}\right)^{\frac{1}{L_j}}}. \tag{8}$$

Here, L_k and L_j respectively represent the length of response $y_{t,i}^{(j)}$ under tokenizers of agent k and j. For combinations of heterogeneous agents that satisfy Definition 2.3 with incompatible tokenizers, we detokenize the response into text and retokenize it using the target agent's tokenizer. Through sequence-level normalization, the slight length discrepancies arising from re-tokenization become negligible. The experiment shows that on the MATH training set, the tokenizers of Qwen and Llama differ in the number of tokens produced for the same reasoning (700 tokens average) by only 4%.

In heterogeneous settings, inter-agent policy discrepancies can be much larger than on-policy updates, making direct use of this ratio overly aggressive. To mitigate this issue, we introduce a non-gradient exponential reweighting:

$$\tilde{s}_{t,i}^{(k,j)} = s_{t,i}^{(k,j)} \cdot \left(\text{sg}[\, s_{t,i}^{(k,j)}\,]\right)^{\alpha} \quad k \neq j, s_{t,i}^{(k,j)} < 1.0 \tag{9}$$

where $\text{sg}[\cdot]$ denotes the stop-gradient operator and $\alpha \geq 0$ controls the degree of conservativeness.

This design biases agent k toward learning from agents whose output distributions are more aligned with its own, while reducing the impact of large cross-agent distribution shifts.

3.4 Stepwise Clipping

The cross-agent importance sampling ratio $s_{t,i}^{(k,j)}$ exhibits fundamentally different behaviors from the self-agent ratio $s_{t,i}^{(k,k)}$, necessitating tailored clipping mechanisms (see Appendix C for further empirical details):

Asymmetric Clipping Bounds. Across training iterations, $s_{t,i}^{(k,k)}$ typically fluctuates closely around 1.0, requiring a narrow clipping range (e.g., $[0.9997, 1.0004]$). In contrast, $s_{t,i}^{(k,j)}$ evolves dynamically and remains significantly smaller than 1.0. To maintain a reasonable clipping proportion, we adopt an asymmetric clipping range for the cross-agent setting, which is typically bounded within $[0.8, 1.0]$.

Stepwise Clipping. Within a single training batch, $s_{t,i}^{(k,k)}$ decays as parameter updates increase, which naturally elevates the clipping proportion. However, $s_{t,i}^{(k,j)}$ fluctuates irregularly. To prevent the later updates within a batch from being dominated by cross-agent responses, we introduce a stepwise strategy that gradually tightens the cross-agent clipping bounds as updates progress within one step.

Let m denote the number of parameter updates performed so far within the current step, and δ_{step} denote the per-update tightening factor. Combining these principles, the clipping function for cross-agent importance sampling is formulated as:

$$\text{clip}\Big(s_{t,i}^{(k,j)},\, 1-\delta+m\cdot\delta_{\text{step}},\, 1.0\Big)\,, \tag{10}$$

where the lower-bound hyperparameter δ is empirically initialized to 0.2 and is dynamically within a step reduced following our stepwise strategy.

4 Experiment

Setting Details. We adopt 7.5k high quality math questions from the MATH dataset Hendrycks et al. [2021] for training. During evaluation, we select a comprehensive set of benchmarks: MATH-500, MATH, GSM8K Cobbe et al. [2021], AIME2025, AMC23, MinervaLewkowycz et al. [2022] and OlympiadHe et al. [2024].

To verify the effectiveness of our method, we conduct experiments on the three heterogeneity settings mentioned in Section 2.1. We compare our approach against the following baselines: (1) **Standard Single-Agent Baselines (GRPO, GSPO)**, which serve as benchmarks for isolated training performance (same rollout cost as HACPO but with half the policy updates); (2) **Data-Equivalent Baseline (GSPO$\times$2)**, a single-agent GSPO setting with double rollouts and updates in every step. This serves to rule out the impact of increased data and updates, verifying the complementary value of heterogeneous agents (double the rollout cost of HACPO but with the same policy updates); (3) **Naive Collaborative Baseline (Naive)**, a two-agent setting with shared rollouts but lacking the algorithmic innovations in Section 3, used to validate the necessity of our proposed discrepancy mitigation techniques (same rollout and policy update costs as HACPO).

We focus our empirical comparisons on different RLVR baselines, omitting Knowledge Distillation (KD) and Multi-Agent RL (MARL) due to fundamental differences in problem formulation. Specifically: (1) Unlike KD, which is a unidirectional process relying on a frozen, homogeneous teacher's output distribution, HACRL enables bidirectional mutual improvement across heterogeneous agents (regardless of relative capabilities). (2) Unlike MARL, which typically requires coupled agents for joint execution, our agents maintain strictly independent execution at inference time.

4.1 Result and Analysis

As detailed in Table 1, HACPO demonstrates superior final performance compared to all baselines across various heterogeneous settings. Across 7 benchmarks, HACPO achieves an average accuracy improvement of +3.6% over the strong baseline GSPO×2, while requiring only half the rollout cost. To illustrate the learning dynamics, Figure 2 presents the training curves of HACPO versus the single-agent GSPO and GSPO $\times$ 2 baseline. We attribute these performance gains to two primary mechanisms inherent in the HACPO: (1) Capability-driven guidance, where stronger models assist in enhancing the performance of weaker ones; and (2) Mutual knowledge exchange, which

Table 1: Main results across three heterogeneity settings (state, size and model). We compare our method against Standard Single-Agent Baselines (GRPO, GSPO), a Resource-Equivalent Baseline (GSPO×2) and a Naive multi-agent rollout share baseline(Naive). The content in the brackets represents a comparison with GSPO × 2.

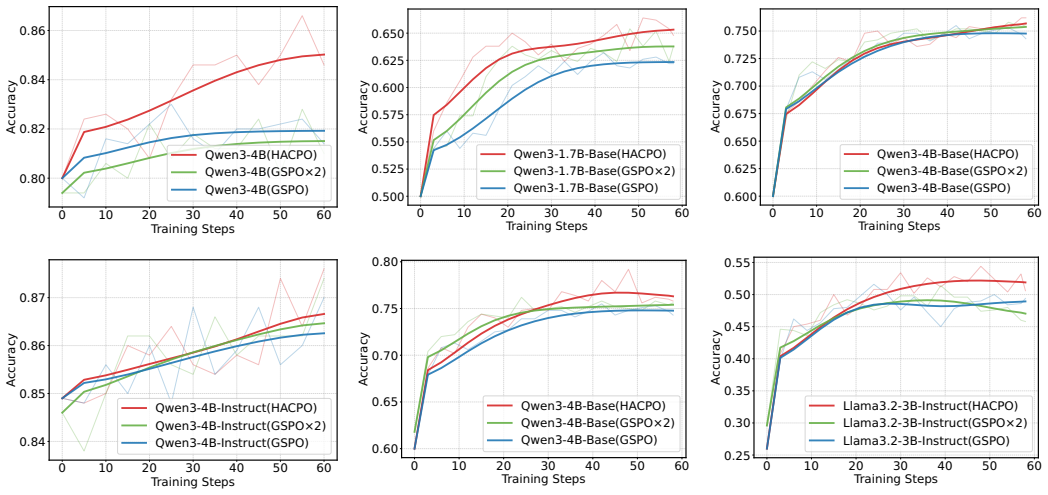
Model	MATH-500	MATH	GSM8K	AIME2025	AMC23	Minerva	Olympiad	AVG
Qwen3-4B and Qwen3-4B-Instruct (**Heterogeneous State**)								
4B	0.802	0.836	0.907	0.335	0.65	0.39	0.524	0.635
4B (GRPO)	0.88	0.889	0.918	0.582	0.775	0.386	0.592	0.717
4B (GSPO)	0.854	0.87	0.925	0.485	0.675	0.412	0.564	0.684
4B (GSPO×2)	0.876	0.875	0.923	0.522	0.675	0.39	0.579	0.691
4B (Naive)	0.728	0.737	0.891	0.378	0.6	0.353	0.394	0.583
4B(HACPO)	**0.91**	**0.905**	**0.933**	**0.622**	**0.85**	**0.423**	**0.643**	**0.755 (+0.064)**
4B-Instruct	0.938	0.937	0.936	0.696	0.85	0.441	0.722	0.789
4B-Instruct (GRPO)	0.93	0.933	0.933	0.676	0.875	0.43	0.72	0.785
4B-Instruct (GSPO)	0.938	0.94	0.939	0.72	0.9	0.43	0.726	0.799
4B-Instruct (GSPO×2)	0.932	0.939	0.942	0.74	0.9	0.43	0.711	0.799
4B-Instruct(Naive)	0.844	0.845	0.936	0.547	0.725	0.39	0.552	0.691
4B-Instruct(HACPO)	**0.948**	**0.943**	**0.946**	**0.757**	**0.95**	**0.452**	**0.732**	**0.818 (+0.019)**
Qwen3-1.7B-Base and Qwen3-4B-Base (**Heterogeneous Size**)								
1.7B-Base	0.5	0.483	0.616	0.033	0.3	0.206	0.229	0.338
1.7B-Base (GRPO)	0.682	0.652	0.824	0.16	0.375	0.272	0.298	0.466
1.7B-Base (GSPO)	0.648	0.641	0.826	0.148	0.45	0.272	0.287	0.467
1.7B-Base (GSPO×2)	0.664	0.65	**0.829**	0.177	0.375	0.265	0.293	0.465
1.7B-Base (Naive)	0.608	0.601	0.798	0.147	0.325	0.235	0.263	0.425
1.7B-Base(HACPO)	**0.69**	**0.674**	0.822	**0.225**	**0.45**	**0.279**	**0.314**	**0.493 (+0.028)**
4B-Base	0.61	0.676	0.445	0.1	0.4	0.308	0.347	0.412
4B-Base (GRPO)	0.796	0.788	0.885	**0.307**	0.475	0.349	0.454	0.579
4B-Base (GSPO)	0.782	0.787	0.877	0.25	0.525	0.368	0.46	0.578
4B-Base (GSPO×2)	0.756	0.794	0.873	0.208	0.55	0.382	0.463	0.575
4B-Base (Naive)	0.708	0.712	0.895	0.196	0.475	0.342	0.354	0.526
4B-Base (HACPO)	**0.808**	**0.801**	**0.903**	0.267	**0.575**	**0.386**	**0.467**	**0.601 (+0.026)**
Qwen3-4B-Base and Llama3.2-3B-Instruct (**Heterogeneous Model**)								
Qwen3	0.61	0.676	0.445	0.1	0.4	0.308	0.347	0.412
Qwen3 (GRPO)	**0.796**	0.788	0.885	**0.307**	0.475	0.349	0.454	0.579
Qwen3 (GSPO)	0.782	0.787	0.877	0.25	0.525	0.368	**0.46**	0.578
Qwen3 (GSPO×2)	0.756	**0.794**	0.873	0.208	0.55	**0.382**	0.463	0.575
Qwen3 (Naive)	0.734	0.712	0.895	0.143	0.55	0.342	0.354	0.526
Qwen3 (HACPO)	0.786	0.783	**0.921**	0.268	**0.6**	0.379	0.442	**0.597 (+0.022)**
Llama3.2	0.267	0.441	0.788	0.0	0.2	0.169	0.158	0.289
Llama3.2 (GRPO)	0.502	0.507	0.814	0.0	0.25	**0.199**	0.174	0.349
Llama3.2 (GSPO)	0.512	0.501	0.812	0.054	0.225	0.184	0.17	0.351
Llama3.2 (GSPO×2)	0.488	0.498	**0.829**	0.0	0.175	0.188	0.159	0.334
Llama3.2 (Naive)	0.406	0.407	0.734	0.0	0.225	0.177	0.107	0.294
Llama3.2 (HACPO)	**0.566**	**0.548**	0.826	**0.054**	**0.35**	0.176	**0.208**	**0.39 (+0.056)**

involves the sharing of complementary rollouts—encompassing both correct solutions and informative errors—between agents.

The improvements are consistent across model combinations, benchmarks, and random seeds, confirming that HACPO's gains are robust. Specifically, we conduct reproducibility experiments with five random seeds on the Qwen3-1.7B-Base and Qwen3-4B-Base combination. Across all runs, HACPO consistently outperforms GSPO by a stable margin: +3.58% on average for Qwen3-1.7B-Base and +2.86% for Qwen3-4B-Base on MATH500. Full results are provided in Appendix F.2. In total, we evaluate HACPO on 6 heterogeneous agent combinations. The main text presents 3 representative settings; results for the additional 3 combinations are deferred to Appendix F.1.

We also report peak GPU memory utilization and wall-clock runtime to demonstrate HACPO's more efficient use of rollout compute. Detailed profiling results are included in Appendix F.3.

Heterogeneous State. In the Qwen3-4B and Qwen3-4B-Instruct setting, we observe asymmetric but non-trivial gains: while the 4B model improves more substantially, the Instruct model also exhibits consistent performance improvements. Although this setting corresponds to heterogeneous state, where agents differ only due to post-training stages, HACPO still enables the stronger agent to benefit from the weaker one. Specifically, the weaker agent contributes complementary exploration signals—such as alternative reasoning paths and informative errors—that are underrepresented in the stronger agent's own rollouts. As a result, learning is not purely unidirectional. Even when



(a) Qwen3-4B and Qwen3-4B-Instruct (b) Qwen3-1.7B-Base and Qwen3-4B-Base (c) Qwen3-4B-Base and Llama3.2-3B-Instruct

Figure 2: Training Curves of HACPO, GSPO and GSPO $\times$ 2

capability-driven guidance dominates, the stronger agent can still extract useful supervisory signals from the weaker agent, leading to measurable performance gains.

Heterogeneous Size. In the Qwen3-1.7B-Base and Qwen3-4B-Base setting, both models improve significantly, validating the mechanism of mutual knowledge exchange. Even with lower capability, the 1.7B model serves as a distinct explorer, generating valuable erroneous responses and a few unique correct solutions that the 4B model fails to produce, thereby facilitating bidirectional knowledge transfer. Rollout analysis further shows that the weaker model attains a higher sampled success rate on 8.4% (627/7424) of the analyzed problems (Appendix F.6).

Heterogeneous Model. Finally, we consider the heterogeneous model setting involving Qwen3-4B-Base and Llama3.2-3B-Instruct, which differ substantially in architecture, tokenizer, and training objectives. Despite this high degree of heterogeneity, we observe consistent performance improvements in both models. These results demonstrate that HACPO is able to extract transferable knowledge from cross-model rollouts and effectively share it across heterogeneous agents. By leveraging verified responses—including correct solutions and informative failure cases—each model can learn from complementary reasoning patterns that are absent from its own policy distribution.

The experimental results show that HACPO significantly improves performance across all three types of heterogeneity, validating its generality and robustness. Additionally, the differences observed among the three settings shed light on the two underlying mechanisms of HACPO.

Beyond One-Way Transfer. For the 1.7B/4B-Base pair, the small model's average score increases from 0.406/0.408 with OPD/OPD+GRPO to 0.452 with HACPO (Frozen Strong), and to 0.493 with full HACPO (Appendix F.4). Joint optimization therefore provides benefits beyond a frozen partner in this setting. Compared with first training the large model and then distilling the small model, HACPO improves their average scores by 2.6 and 3.3 percentage points, respectively, while reducing the time to obtain both models by 14.2% (Appendix F.5). Together with the rollout analysis, these results support bidirectional learning within an independently deployable model portfolio.

4.2 Three-Agent Collaboration

We jointly train Qwen3-1.7B-Base, Llama3.2-1B-Instruct, and Gemma3-1B-Instruct Gemma Team [2025], extending collaboration to three distinct model families. We compare each resulting model with independently trained GSPO and GSPO$\times$3. The latter triples the single-agent rollout budget to match the number of training rollouts available to each HACPO agent from the shared pool.

As shown in Table 2, HACPO improves the average scores of all three models over GSPO$\times$3 by 1.6, 1.8, and 1.9 percentage points, respectively. These results demonstrate effective collaboration beyond two agents, including partners from three different families. Gains vary across individual benchmarks, while all three models benefit on average.

Table 2: Three-agent collaboration across model families. AVG is the mean accuracy across seven benchmarks; parentheses report absolute gains over GSPO×3.

Method	MATH500	MATH	GSM8K	AIME2025	AMC23	Minerva	Olympiad	AVG
Qwen3-1.7B-Base								
GSPO	0.648	0.641	0.826	0.148	**0.450**	0.272	0.287	0.467
GSPO×3	0.664	0.668	**0.833**	0.133	0.400	**0.302**	0.296	0.471
HACPO	**0.690**	**0.688**	0.824	**0.177**	0.425	0.287	**0.317**	**0.487 (+0.016)**
Llama3.2-1B-Instruct								
GSPO	**0.356**	0.346	0.523	0.021	0.125	0.066	**0.088**	0.218
GSPO×3	0.340	0.354	0.520	0.000	0.175	0.077	0.077	0.220
HACPO	0.340	**0.363**	**0.563**	**0.033**	**0.200**	**0.081**	**0.088**	**0.238 (+0.018)**
Gemma3-1B-Instruct								
GSPO	0.452	0.451	0.616	0.033	0.225	0.052	0.150	0.283
GSPO×3	0.472	0.450	0.623	0.033	0.225	0.040	**0.167**	0.287
HACPO	**0.496**	**0.463**	**0.626**	**0.067**	**0.275**	**0.059**	0.158	**0.306 (+0.019)**

4.3 Ablation Study

Agent-Capability-Aware Advantage Estimation. Ablation on the Qwen3-1.7B/4B-Base combination (Table 3) confirms that removing this module significantly degrades performance. This decline stems from the systematic bias in standard group-relative advantages in multi-agent setting due to the capability discrepancy cross heterogenous agents. Our method addresses this by constructing *agent-capability-aware* advantage baselines—raising the standard for the stronger models and lowering it for the weaker ones—thereby preserving baseline unbiasedness in the oracle setting of Theorem D.5.

Table 3: Ablation of Advantage Estimator

Model	MATH-500	MATH	GSM8K	AIME2025	AMC23	Minerva	Olympiad	AVG
1.7B(HACPO - Adv)	**0.696**	0.659	**0.825**	0.126	0.375	0.261	0.313	0.465
1.7B(HACPO)	0.69	**0.674**	0.822	**0.225**	**0.45**	**0.279**	**0.314**	**0.493**
4B(HACPO - Adv)	0.774	0.771	**0.912**	**0.308**	0.55	0.348	0.442	0.586
4B(HACPO)	**0.808**	**0.801**	0.903	0.267	**0.575**	**0.386**	**0.467**	**0.601**

Model Capabilities Discrepancy Coefficient. We isolate this coefficient in gradient modulation by disabling it in Eq.7, while retaining it for advantage estimation. Table 4 confirms that removing this modulation degrades performance. This validates the coefficient's critical function as a capability-aware scaler: it amplifies gradients from stronger agents to accelerate learning, while attenuating updates from weaker ones to mitigate potential noise.

Table 4: Ablation of Model Capabilities Discrepancy Coefficient

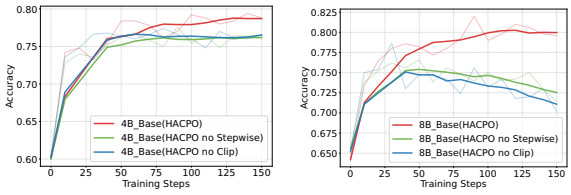
Model	MATH-500	math	gsm8k	aime2025	ACM23	minerva	olympiad	AVG
1.7B(HACPO - ω)	0.666	0.657	0.806	0.105	0.425	0.25	**0.324**	0.462
1.7B(HACPO)	**0.69**	**0.674**	**0.822**	**0.225**	**0.45**	**0.279**	0.314	**0.493**
4B(HACPO - ω)	0.803	0.797	0.902	0.261	0.55	**0.401**	**0.475**	0.6
4B(HACPO)	**0.808**	**0.801**	**0.903**	**0.267**	**0.575**	0.386	0.467	**0.601**

Exponential Importance Sampling. We examined the impact of α on Qwen3-1.7B/4B-Base and Qwen3-4B/8B-Base combinations (Table 5). Results highlight a critical trade-off: increasing α enforces a more conservative policy towards cross-agent responses, which aids stability by suppressing large distribution shifts but hinders efficiency by reducing the effective learning signal. Thus, the optimal α is model combination dependent, necessitating a balance between stable convergence and maximal information extraction.

Stepwise Clipping. We assess the necessity of this mechanism on the Qwen3-4B/8B-Base combination. As visualized in Figure 3, removing the clipping constraint (**no Clip**) causes severe instability, while omitting the stepwise schedule (**no Stepwise**) leads to suboptimal convergence compared to the full HACPO. This confirms that the stepwise clipping is indispensable for stabilizing collaborative learning, as neither unconstrained nor statically bounded updates suffice to handle high-variance cross-agent responses.

Table 5: The Impact of α (MATH500)

α	0.0	**1.0**	2.0	3.0
Qwen3-1.7B-Base and Qwen3-4B-Base				
1.7B-Base	0.63	0.664	0.654	**0.668**
4B-Base	0.756	**0.792**	0.768	0.77
Qwen3-4B-Base and Qwen3-8B-Base				
4B-Base	0.772	0.776	0.77	**0.776**
8B-Base	0.764	0.772	0.766	**0.778**



(a) Qwen3-4B-Base (b) Qwen3-8B-Base

Figure 3: Ablation of Stepwise Clipping (MATH500)

5 Related Work

Our work is most closely related to Reinforcement Learning with Verifiable Rewards (RLVR), with Group Sequence Policy Optimization (GSPO) being the most relevant prior study. GSPO demonstrates the efficacy of sequence-level importance sampling in Mixture-of-Experts (MoE) models, where tokens may originate from different networks. This insight inspires our approach to facilitate rollout sharing among heterogeneous agents. Additionally, our work shares conceptual parallels with Multi-Agent Reinforcement Learning (MARL). A more detailed discussion of related work is provided in Appendix B.

6 Conclusion

We propose HACRL, a collaborative multi-agent reinforcement learning problem tailored for heterogeneous agent ecosystems. HACRL enables principled rollout sharing among heterogeneous agents, improving sample utilization efficiency while promoting cross-agent knowledge transfer. To instantiate this problem, we introduce HACPO, which incorporates four tailored mechanisms to mitigate capability discrepancies and policy distribution shifts arising during collaborative policy optimization. Our theoretical analysis establishes the unbiasedness of the oracle capability-aware baseline and bounds the finite-batch capability ratio estimation error under the stated assumptions. Extensive experiments demonstrate that HACPO consistently and significantly improves performance across all heterogeneity types.

7 Limitation and Future Work

While HACPO inherently supports $n \geq 3$ agents, current empirical evaluations are restricted to two- and three-agent settings due to prohibitive computational resource and non-trivial infrastructure modifications required within the verl framework. Nevertheless, our extensive validation across six diverse two-agent model combinations and three distinct types of heterogeneity serves as a strong empirical proxy. The consistent, robust gains observed across these highly varied two-agent pairs and the three-agent setting indicate that HACPO's collaborative mechanisms will generalize effectively to larger agent ecosystems. Future work will address these engineering bottlenecks to empirically scale HACPO.

Acknowledgments and Disclosure of Funding

This research was supported by China NSFC (No. 62506024). The authors declare no competing interests.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*, 2024.

Jeffrey Amico, Gabriel Passamani Andrade, John Donaghy, Ben Fielding, Tristin Forbus, Harry Grieve, Semih Kara, Jari Kolehmainen, Yihua Lou, Christopher Nies, Edward Phillip Flores Nuño, Diogo Ortega, Shikhar Rastogi, Austin Virts, and Matthew J. Wright. Sharing is caring: Efficient LM post-training with collective RL experience sharing, 2025. URL https://arxiv.org/abs/2509.08721.

Rohan Anil, Gabriel Pereyra, Alexandre Passos, Robert Ormandi, George E Dahl, and Geoffrey E Hinton. Large scale distributed neural network training through online distillation. *arXiv preprint arXiv:1804.03235*, 2018.

Yikun Ban, Fengkai Yang, Fangzheng Chen, Yibo Wang, Zhijun Chen, Zhongyi Li, Zixuan Huang, Xiaoyuan Zhang, Gongxun Li, Zehao Chen, et al. Epistemic exploration toward artificial general intelligence. 2026.

Zehao Chen, Gongxun Li, Tianxiang Ai, Zixuan Huang, Xiaodong Liu, Yifei Li, Wang Zhou, Fuzhen Zhuang, Xianglong Liu, Jianxin Li, et al. Weak-driven learning: How weak agents make strong agents stronger. *arXiv preprint arXiv:2602.08222*, 2026.

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

Gemma Team. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025. URL https://arxiv.org/abs/2503.19786.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International journal of computer vision*, 129(6):1789–1819, 2021.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3828–3850, 2024.

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

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Zixuan Huang, Yikun Ban, Lean Fu, Xiaojie Li, Zhongxiang Dai, Jianxin Li, and Deqing Wang. Adaptive batch-wise sample scheduling for direct preference optimization. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=8FN25PlktS.

Zixuan Huang, Xin Xia, Yuxi Ren, Jianbin Zheng, Xuanda Wang, Zhixia Zhang, Hongyan Xie, Songshi Liang, Zehao Chen, Xuefeng Xiao, et al. Does your reasoning model implicitly know when to stop thinking? *arXiv preprint arXiv:2602.08354*, 2026a.

Zixuan Huang, Xin Xia, Yuxi Ren, Jianbin Zheng, Xuefeng Xiao, Hongyan Xie, Li Huaqiu, Songshi Liang, Zhongxiang Dai, Fuzhen Zhuang, et al. Real-time aligned reward model beyond semantics. *arXiv preprint arXiv:2601.22664*, 2026b.

Zixuan Huang, Yang Zhou, Kaixuan Wang, Guli Zhang, Hongyan Xie, Yakun Zhu, Hao Geng, Xiaozhi Chen, Yikun Ban, and Deqing Wang. Deferred exposure of future trajectories for verifiable reasoning in autonomous driving VLMs. *arXiv preprint arXiv:2608.01755*, 2026c. URL https://arxiv.org/abs/2608.01755.

Mingze Kong, Zikun Qu, Zhongquan Zhou, Pengyu Liang, Xiang Li, Zhiwei Shang, Zhi Hong, Kaiyu Huang, Zhiyong Wang, and Zhongxiang Dai. Workflow-R1: Group sub-sequence policy optimization for multi-turn workflow construction. *arXiv preprint arXiv:2602.01202*, 2026.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022.

Zhongyi Li, Wan Tian, Jingyu Chen, Kangyao Huang, Huiming Zhang, Hui Yang, Tao Ren, Jinyang Jiang, Yijie Peng, Yikun Ban, et al. Adaptive robust estimator for multi-agent reinforcement learning. *arXiv preprint arXiv:2603.21574*, 2026.

Junwei Liao, Muning Wen, Jun Wang, and Weinan Zhang. Marft: Multi-agent reinforcement fine-tuning. *arXiv preprint arXiv:2504.16129*, 2025.

Hehai Lin, Shilei Cao, Sudong Wang, Haotian Wu, Minzhi Li, Linyi Yang, Juepeng Zheng, and Chengwei Qin. Interactive learning for LLM reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 6085–6108. Association for Computational Linguistics, 2026a. doi: 10.18653/v1/2026.findings-acl.303. URL https://aclanthology.org/2026.findings-acl.303/.

Yexiong Lin, Yu Yao, Yang Zhou, and Tongliang Liu. Beyond optimal transport: Model-aligned coupling for flow matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Findings*, pages 3955–3964, June 2026b. URL https://openaccess.thecvf.com/content/CVPR2026F/html/Lin_Beyond_Optimal_Transport_Model-Aligned_Coupling_for_Flow_Matching_CVPRF_2026_paper.html.

Lihui Liu. Graph-o1: Monte carlo tree search with reinforcement learning for text-attributed graph reasoning. *arXiv preprint arXiv:2512.17912*, 2025.

Weiting Liu, Jieyi Bi, Wanqi Zhou, Jianfeng Feng, Yining Ma, Ai Han, and Wenlian Lu. ToolAnchor: Anchoring counterfactual context to boost agentic tool-use capability. *arXiv preprint arXiv:2607.14145*, 2026a. URL https://arxiv.org/abs/2607.14145.

Weiting Liu, Han Wu, Yufei Kuang, Xiongwei Han, Tao Zhong, Jianfeng Feng, and Wenlian Lu. Automated optimization modeling via a localizable error-driven perspective. *arXiv preprint arXiv:2602.11164*, 2026b.

Xiaodong Lu, Xiaohan Wang, Jiajun Chai, Guojun Yin, Wei Lin, Zhijun Chen, Yu Luo, Fuzhen Zhuang, Yikun Ban, and Deqing Wang. Contextual rollout bandits for reinforcement learning with verifiable rewards. *arXiv preprint arXiv:2602.08499*, 2026.

Hao Ma, Tianyi Hu, Zhiqiang Pu, Boyin Liu, Xiaolin Ai, Yanyan Liang, and Min Chen. Coevolving with the other you: Fine-tuning llm with sequential cooperative multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 37:15497–15525, 2024.

Sumeet Ramesh Motwani, Chandler Smith, Rocktim Jyoti Das, Rafael Rafailov, Ivan Laptev, Philip H. S. Torr, Fabio Pizzati, Ronald Clark, and Christian Schroeder de Witt. MALT: Improving reasoning with multi-agent LLM training. In *Conference on Language Modeling*, 2025. URL https://arxiv.org/abs/2412.01928.

Chanwoo Park, Seungju Han, Xingzhi Guo, Asuman Ozdaglar, Kaiqing Zhang, and Joo-Kyung Kim. Maporl: Multi-agent post-co-training for collaborative large language models with reinforcement learning. *arXiv preprint arXiv:2502.18439*, 2025.

Zikun Qu, Min Zhang, Mingze Kong, Zhiwei Shang, Yikun Ban, Shuang Qiu, and Zhongxiang Dai. SPOT: Sparse probing and outcome calibration for on-policy distillation. *arXiv preprint arXiv:2608.04419*, 2026.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *CoRR*, abs/2409.19256, 2024. doi: 10.48550/ARXIV.2409.19256. URL https://arxiv.org/abs/2409.19256.

Huaiyang Wang, Xiaojie Li, Xiaohan Wang, Zhixia Zhang, Xiaodong Lu, Zixuan Huang, Jiajun Chai, Guojun Yin, Deqing Wang, Haoyi Zhou, et al. Policy improvement reinforcement learning. *arXiv preprint arXiv:2604.00860*, 2026.

Jiakang Wang, Runze Liu, Lei Lin, Wenping Hu, Xiu Li, Fuzheng Zhang, Guorui Zhou, and Kun Gai. Aspo: Asymmetric importance sampling policy optimization. *arXiv preprint arXiv:2510.06062*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a.

Fengkai Yang, Zherui Chen, Xiaohan Wang, Xiaodong Lu, Jiajun Chai, Guojun Yin, Wei Lin, Shuai Ma, Fuzhen Zhuang, Deqing Wang, Yaodong Yang, Jianxin Li, and Yikun Ban. Your group-relative advantage is biased, 2026. URL https://arxiv.org/abs/2601.08521.

Shihui Yang, Chengfeng Dou, Peidong Guo, Kai Lu, Qiang Ju, Fei Deng, and Rihui Xin. Dcpo: Dynamic clipping policy optimization. *arXiv preprint arXiv:2509.02333*, 2025b.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Yuzhong Zhao, Yue Liu, Junpeng Liu, Jingye Chen, Xun Wu, Yaru Hao, Tengchao Lv, Shaohan Huang, Lei Cui, Qixiang Ye, et al. Geometric-mean policy optimization. *arXiv preprint arXiv:2507.20673*, 2025.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

Jiaru Zou, Yikun Ban, Zihao Li, Yunzhe Qi, Ruizhong Qiu, Ling Yang, and Jingrui He. Transformer copilot: Learning from the mistake log in LLM fine-tuning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=MRvxlTlkNQ.

Jiaru Zou, Rui Pan, Ruizhong Qiu, Pan Lu, Shizhe Diao, Jindong Jiang, Hanghang Tong, Tong Zhang, Markus J. Buehler, Jingrui He, and James Zou. Recursive multi-agent systems, 2026a. URL https://arxiv.org/abs/2604.25917.

Jiaru Zou, Ruizhong Qiu, Gaotang Li, Xiyuan Yang, Katherine Tieu, Pan Lu, Ke Shen, Hanghang Tong, Yejin Choi, Jingrui He, James Zou, Mengdi Wang, and Ling Yang. Latent collaboration in multi-agent systems. In *Forty-third International Conference on Machine Learning*, 2026b. URL https://openreview.net/forum?id=syG9I9ofd8.

A Training and Evaluation Details

All experiments in this paper are conducted using verl Sheng et al. [2024]. In the experiments, we set the maximum prompt length to 1024 and the maximum response length to 4096. We use the MATH dataset for training. The learning rate is set to 1×10^{-6}. For the responses generated by the trained agents in HACPO or single GSPO, we set $\epsilon_{\text{low}} = 0.0003$ and $\epsilon_{\text{high}} = 0.0004$, which is consistent with the setting mentioned in GSPOZheng et al. [2025]. As for the single GRPO, we set $\epsilon_{\text{low}} = 0.2$ and $\epsilon_{\text{high}} = 0.28$, which follows the trick mentioned in DAPO Yu et al. [2025] and is widely used. The batch size is set to 128, with a mini-batch size of 64 and $n = 8$ rollouts per prompt. In the Resource-Equivalent Baseline (GSPO×2), we use a mini-batch size of 32 and $n = 16$ rollouts per prompt to ensure double updates per step, while maintaining a consistent number of rollouts per update with other settings. We train for one epoch, except when examining the impact of stepwise clipping on stabilizing the training process. During evaluation, due to the high complexity of benchmarks such as AIME2025, we adopt a maximum response length of 8192 tokens in the main experiments and the ablation of Agent-Capability-Aware Advantage Estimator and Model Capability Discrepancy Coefficient (Table 1, Table 11, Table 4 and Table 3). The three-agent and distillation comparisons, as well as the reevaluated seed experiments (Table 8), also use the 8192-token evaluation limit. For all other experiments, the maximum response length is kept consistent with the training configuration and is set to 4096 tokens. Experiments are conducted on eight NVIDIA A800 GPUs, each with 80 GB of memory.

Regarding the models used in our experiments, We employed the Qwen3 Yang et al. [2025a] and Llama3.2 Grattafiori et al. [2024] series of models, as well as Gemma3-1B-Instruct Gemma Team [2025] in the three-agent experiment. In detail, Qwen3-(1.7B/4B/8B)-Base denotes the base models, while Qwen3-(1.7B/4B/8B) refer to the distilled variants obtained through strong model distillation from their corresponding base models. In addition, Qwen3-4B-Instruct is a further fine-tuned version of Qwen3-4B, designed to better follow user instructions and generate more accurate responses.

For the clipping boundary δ in the exponential importance sampling of α, as well as the gradient clipping step size δ_{step}, each experiment has slight variations. We provide the specific settings used for each experiment in the Table 6. A commonly used set of parameters is $\alpha = 1$, 1-$\delta = 0.8$, and $\delta_{\text{step}} = 0.025$.

Table 6: The Details of Hyperparameter

Model Combination	α	1-δ	δ_{step}
Qwen3-4B and Qwen3-4B-Instruct	3.0	0.8	0.01
Qwen3-1.7B-Base and Qwen3-4B-Base	1.0	0.8	0.025
Qwen3-4B-Base and Qwen3-8B-Base	3.0	0.8	0.025
Llama3.2-1B-Instruct and Llama3.2-3B-Instruct	1.0	0.9	0.01
Qwen3-1.7B-Base and Llama3.2-1B-Instruct	1.0	0.8	0.025
Qwen3-4B-Base and Llama3.2-3B-Instruct	1.0	0.8	0.025

B Additional Related Work

B.1 Reinforcement Learning with Verifiable Rewards

Group Relative Policy Optimization (GRPO) Shao et al. [2024] estimates advantages from groups of responses to the same prompt, avoiding a separate critic network. Subsequent work studies group-based advantage estimation Yang et al. [2026] and develops variants that improve optimization Yu et al. [2025], Yang et al. [2025b], Zhao et al. [2025], Wang et al. [2025], Liu [2025], Wang et al. [2026], Kong et al. [2026], Lu et al. [2026], Li et al. [2026]. Group Sequence Policy Optimization (GSPO) Zheng et al. [2025] uses sequence-level importance sampling to improve training stability, including for Mixture-of-Experts models. HACPO builds on this sequence-level objective and addresses the additional capability and policy-distribution differences that arise when heterogeneous agents directly reuse one another's rollouts.

Related post-training work studies adaptive sample scheduling for DPO Huang et al. [2025], guided rollout sampling for efficient reasoning Huang et al. [2026a], and reward-model adaptation to

changing policy distributions Huang et al. [2026b]. In adjacent work on model-dependent supervision, Model-Aligned Coupling (MAC) selects source–target couplings for flow matching according to the current model's prediction error Lin et al. [2026b]. Beyond mathematical reasoning, DEFT-RLVR explores verifiable reasoning in autonomous-driving VLMs by deferring exposure to candidate future trajectories Huang et al. [2026c].

B.2 Collaborative Learning and Scope of Novelty

Training-time collaboration followed by independent deployment is an established idea, including in LLM co-training methods such as CORY Ma et al. [2024], WMSS Chen et al. [2026] and ILR Lin et al. [2026a]. HACRL studies a specific problem within this broader setting: bidirectional sharing of verified rollouts among heterogeneous LLM agents in RLVR. Our contribution is to formulate this problem across state, size, and model heterogeneity and develop HACPO to address the resulting capability differences and cross-policy distribution shifts.

CORY Ma et al. [2024] initializes a pioneer and an observer as two copies of the same LLM. The pioneer responds to the query, and the observer conditions on both the query and the pioneer's response. Periodic role exchange allows either model to be used independently after training. ILR Lin et al. [2026a] also targets independent reasoning after collaborative training, but transfers information through explicit inter-agent discussions and calibrates rewards using the other model's reward distribution. HACRL instead lets each agent generate responses independently and directly reuses complete verified rollouts for policy updates, without conditioning generation on another agent's response.

SAPO implements decentralized, asynchronous rollout sharing across distributed training nodes and validates it in controlled experiments with eight homogeneous Qwen2.5-0.5B agents Amico et al. [2025].

Other methods optimize agents for a collaborative inference systemZou et al. [2026a, 2025, 2026b]. MAPoRL Park et al. [2025] co-trains LLMs for multi-turn discussion to improve their joint answer, while MALT Motwani et al. [2025] trains specialized generator, verifier, and refiner agents for a sequential reasoning pipeline. In HACRL, each trained model solves a request without communicating with its training partners or relying on specialized inference-time roles.

B.3 Knowledge Distillation

Knowledge distillation transfers information between models Hinton et al. [2015], Gou et al. [2021]; its variants include online collaboration Anil et al. [2018] and on-policy training using student-generated responses Agarwal et al. [2024], Qu et al. [2026]. Our experimental comparison concerns specific one-way baselines: OPD and OPD+GRPO update the weaker model using a frozen stronger teacher, whereas HACPO updates every participating model using shared verified rollouts. We also evaluate HACPO with a frozen stronger partner to isolate the benefit of joint optimization (Appendix F.4). A separate sequential train-then-distill comparison first trains the stronger model and then uses its frozen checkpoint to train the weaker model (Appendix F.5). Together, these comparisons assess whether bidirectional collaboration improves the target model portfolio beyond the evaluated one-way alternatives.

Teacher-assisted post-training also improves the available learning signals. MIND combines reinforcement learning with supervised updates on locally corrected student responses Liu et al. [2026b], while ToolAnchor verifies interventions proposed by diverse teachers through student rollouts before incorporating them into agentic post-training Liu et al. [2026a].

C Heterogeneous Agent Importance Sampling Analysis

In the reinforcement learning paradigm, importance sampling is commonly used to stabilize updates, often through a clipping mechanism. The clipping range typically centers around 1.0. For instance, in GSPO, the upper and lower bounds for clipping are set to 1.0004 and 0.9997, respectively. However, in a multi-agent setting, the importance sampling values for samples from other agents do not exhibit the same pattern and fluctuate as training progresses.

In the experiment involving Qwen3-1.7B-Base and Qwen3-4B-Base, we distinguish between self-generated responses and cross-agent responses, denoted as $s_{t,i}^{(1,1)}$ and $s_{t,i}^{(1,2)}$, respectively. These values represent the average importance sampling across each training step. It is important to note that while $s_{t,i}^{(1,1)}$ remains stable and tends to stay around 1 throughout training, $s_{t,i}^{(1,2)}$ does not follow a fixed range and fluctuates as training progresses. The results are shown in Table 7

Table 7: $s_{t,i}^{(1,1)}$ and $s_{t,i}^{(1,2)}$ of Qwen3-1.7B-Base in all steps

Model	mean	max	min	range
s^{homo}	1.00002	1.00020	0.99960	0.00060
s^{hete}	0.89550	0.93615	0.86198	0.07417

For self-generated responses, as the number of updates(mini batches) within a batch increases, the discrepancy between the sampling policy $\pi_{\theta_{\text{old}}}$ and the current policy π_θ grows, leading to an increased $s_{t,i}^{(1,1)}$ and a higher ratio of clipped tokens. However, for cross-agent responses, the discrepancy between the current policy $\pi_\theta^{(k)}$ and the sampling model's policy $\pi_{\theta_{\text{old}}}^{(j)}$ fluctuates unpredictably, leading to a variable $s_{t,i}^{(1,2)}$ and the ratio of clipped tokens.

In a batch with multiple mini-batches, as the number of updates increases, self-generated responses become more heavily clipped in later mini-batches due to the growing discrepancy between the current and old policies. Therefore, the influence of cross-agent responses is likely to increase in later mini-batches, as their importance sampling values are less predictable, leading to an instability if they dominate the update.

D Theoretical Analysis

To ensure mathematical rigor, the appendix proofs use the expanded notation $R(x, y)$ for rewards and π_t for policies at step t. All other symbols are consistent with the main text.

D.1 Oracle Capability Baseline

This subsection defines the oracle capability ratio using expectations over the prompt distribution and the response distributions of the agents. It proves that the capability-aware baseline in Eq. (5) has the same expectation as the reward of the learner agent. All expectations below are taken jointly over prompt sampling and response sampling.

We first define the expected reward of each agent and the batch-level reward mean used by the oracle baseline.

Assumption D.1 (Finite expected reward)**.** Fix a training step t and n agents. Let $\mathcal{D}_t$ be the prompt distribution at step t. For each agent $a \in \{1, \dots, n\}$, let $\pi_t^{(a)}(\cdot \mid x)$ denote the response distribution of agent a given prompt x. Assume the reward has finite expectation for each agent:

$$\mathbb{E}_{x\sim\mathcal{D}_t,\, y\sim\pi_t^{(a)}(\cdot|x)}\left[|R(x,y)|\right] < \infty. \tag{11}$$

Define the conditional prompt value and the prompt-averaged expected reward as

$$q_t^{(a)}(x) := \mathbb{E}_{y\sim\pi_t^{(a)}(\cdot|x)}\left[R(x,y)\right], \tag{12}$$

$$p_t^{(a)} := \mathbb{E}_{x\sim\mathcal{D}_t}\left[q_t^{(a)}(x)\right] = \mathbb{E}_{x\sim\mathcal{D}_t,\, y\sim\pi_t^{(a)}(\cdot|x)}\left[R(x,y)\right]. \tag{13}$$

Assumption D.2 (Positive oracle denominators)**.** Throughout the oracle baseline construction, for every denominator agent $j \in \{1, \dots, n\}$,

$$p_t^{(j)} > 0. \tag{14}$$

Definition D.3 (Oracle capability ratio)**.** For agents k and j with $p_t^{(j)} > 0$, the oracle capability ratio is

$$\bar{\omega}_t^{(k,j)} := \frac{p_t^{(k)}}{p_t^{(j)}}. \tag{15}$$

In particular, $\bar{\omega}_t^{(k,k)} = 1$.

Definition D.4 (Oracle capability-aware baseline)**.** Let the batch at step t contain B prompt instances $X_{t,1}, \ldots, X_{t,B}$ sampled independently from $\mathcal{D}_t$. Conditional on $X_{t,b}$, let $Y_{t,b,1}^{(j)}, \ldots, Y_{t,b,G}^{(j)}$ be sampled independently from $\pi_t^{(j)}(\cdot \mid X_{t,b})$. Define

$$\widetilde{P}_t^{(j)} := \frac{1}{BG} \sum_{b=1}^{B} \sum_{i=1}^{G} R\Big(X_{t,b}, Y_{t,b,i}^{(j)}\Big) \tag{16}$$

as the batch-level empirical mean reward of agent j. The oracle capability-aware baseline for learner agent k is

$$\mu_{t,\star}^{(k)} := \frac{1}{n} \sum_{j=1}^{n} \bar{\omega}_t^{(k,j)} \widetilde{P}_t^{(j)}. \tag{17}$$

Theorem D.5 (Unbiased oracle capability-aware baseline)**.** *Under Assumptions D.1 and D.2, the oracle capability-aware baseline $\mu_{t,\star}^{(k)}$ in Definition D.4 satisfies*

$$\mathbb{E}\Big[\mu_{t,\star}^{(k)}\Big] = p_t^{(k)} = \mathbb{E}_{x \sim \mathcal{D}_t,\, y \sim \pi_t^{(k)}(\cdot|x)} \left[R(x,y)\right]. \tag{18}$$

Proof. By the definition of $\widetilde{P}_t^{(j)}$ and Assumption D.1,

$$\mathbb{E}\Big[\widetilde{P}_t^{(j)}\Big] = p_t^{(j)}. \tag{19}$$

Therefore,

$$\mathbb{E}\Big[\mu_{t,\star}^{(k)}\Big] = \frac{1}{n} \sum_{j=1}^{n} \bar{\omega}_t^{(k,j)} \mathbb{E}\Big[\widetilde{P}_t^{(j)}\Big] \tag{20}$$

$$= \frac{1}{n} \sum_{j=1}^{n} \frac{p_t^{(k)}}{p_t^{(j)}} p_t^{(j)} = p_t^{(k)}. \tag{21}$$

□

Corollary D.6 (Zero-mean oracle centered reward)**.** *Let $y \sim \pi_t^{(k)}(\cdot \mid x)$ with $x \sim \mathcal{D}_t$, and define the oracle centered reward*

$$\bar{A}_{t,\star}^{(k)}(x,y) := R(x,y) - \mu_{t,\star}^{(k)}. \tag{22}$$

Under the conditions of Theorem D.5,

$$\mathbb{E}\Big[\bar{A}_{t,\star}^{(k)}(x,y)\Big] = 0. \tag{23}$$

Proof. By linearity of expectation,

$$\mathbb{E}\Big[\bar{A}_{t,\star}^{(k)}(x,y)\Big] = \mathbb{E}_{x \sim \mathcal{D}_t,\, y \sim \pi_t^{(k)}(\cdot|x)} \left[R(x,y)\right] - \mathbb{E}\Big[\mu_{t,\star}^{(k)}\Big]. \tag{24}$$

Theorem D.5 shows that the two terms are equal, so the difference is zero. □

Remark D.7 (Single-agent degeneration). When $n = 1$, the oracle ratio and baseline become

$$\bar{\omega}_t^{(k,k)} = 1, \qquad \mu_{t,\star}^{(k)} = \widetilde{P}_t^{(k)}. \tag{25}$$

Thus the multi-agent statement reduces to the standard single-agent sample-mean baseline.

D.2 Empirical Ratio Concentration

The next result bounds the deviation between the batch-level empirical ratio and the oracle ratio.

Assumption D.8 (Bounded batch-level sampling)**.** In addition to Assumption D.1, assume the reward is bounded:

$$0 \le R(x, y) \le 1. \tag{26}$$

Let B be the number of prompts in the batch at step t, and let G be the number of responses sampled for each prompt and agent. Define

$$N_{\text{batch}} := BG. \tag{27}$$

The batch at step t is sampled under the prompt distribution and agent policies in Assumption D.1. For each prompt instance $b \in \{1, \dots, B\}$, let $X_b \sim \mathcal{D}_t$ be sampled independently. Conditional on X_b, responses $Y_{b,1}^{(a)}, \dots, Y_{b,G}^{(a)}$ are sampled independently from $\pi_t^{(a)}(\cdot \mid X_b)$ for each agent a, and different prompt groups are independent. The observed reward samples are

$$Z_{b,g}^{(a)} := R\Big(X_b, Y_{b,g}^{(a)}\Big), \qquad b = 1, \dots, B, \quad g = 1, \dots, G. \tag{28}$$

Definition D.9 (Batch-level empirical capability ratio)**.** Using the batch-level samples in Eq. (28), define

$$\hat{P}_t^{(a)} := \frac{1}{N_{\text{batch}}} \sum_{b=1}^{B} \sum_{g=1}^{G} Z_{b,g}^{(a)}, \qquad \hat{\omega}_t^{(k,j)} := \frac{\hat{P}_t^{(k)}}{\hat{P}_t^{(j)}}. \tag{29}$$

Lemma D.10 (Batch-level concentration of empirical capabilities)**.** *Under Assumption D.8, for any $\delta \in (0, 1)$, with probability at least $1 - \delta$, all agents simultaneously satisfy*

$$\left|\hat{P}_t^{(a)} - p_t^{(a)}\right| \le \epsilon_{\text{batch}}(\delta), \qquad \textit{for all } a = 1, \dots, n, \tag{30}$$

where

$$\epsilon_{\text{batch}}(\delta) := \epsilon_{\text{prompt}}(\delta) + \epsilon_{\text{resp}}(\delta), \tag{31}$$

with

$$\epsilon_{\text{prompt}}(\delta) := \sqrt{\frac{\log(4n/\delta)}{2B}}, \tag{32}$$

$$\epsilon_{\text{resp}}(\delta) := \sqrt{\frac{\log(4n/\delta)}{2BG}}. \tag{33}$$

Proof. For a fixed agent a, define the empirical prompt-value average

$$\bar{q}_{t,\text{batch}}^{(a)} := \frac{1}{B} \sum_{b=1}^{B} q_t^{(a)}(X_b). \tag{34}$$

Then

$$\hat{P}_t^{(a)} - p_t^{(a)} = \left(\hat{P}_t^{(a)} - \bar{q}_{t,\text{batch}}^{(a)}\right) + \left(\bar{q}_{t,\text{batch}}^{(a)} - p_t^{(a)}\right). \tag{35}$$

Since $q_t^{(a)}(X_b) \in [0, 1]$ and the prompt instances are independent, Hoeffding's inequality gives

$$\Pr\left(\left|\bar{q}_{t,\text{batch}}^{(a)} - p_t^{(a)}\right| \ge \epsilon\right) \le 2\exp(-2B\epsilon^2). \tag{36}$$

Conditional on the prompt instances, the rewards $Z_{b,g}^{(a)}$ are independent and bounded in $[0, 1]$, with conditional means $q_t^{(a)}(X_b)$. Therefore,

$$\Pr\left(\left|\hat{P}_t^{(a)} - \bar{q}_{t,\text{batch}}^{(a)}\right| \ge \epsilon \;\middle|\; X_1, \dots, X_B\right) \le 2\exp(-2BG\epsilon^2). \tag{37}$$

Taking expectation over the prompt instances gives the unconditional bound

$$\Pr\left(\left|\hat{P}_t^{(a)} - \bar{q}_{t,\text{batch}}^{(a)}\right| \ge \epsilon\right) \le 2\exp(-2BG\epsilon^2). \tag{38}$$

Choose $\epsilon_{\text{prompt}}(\delta)$ and $\epsilon_{\text{resp}}(\delta)$ as in Eqs. (32)–(33). For each agent, the prompt and response failure probabilities are each at most $\delta/(2n)$. Applying the union bound over both terms and over the n agents gives Eq. (30). □

Theorem D.11 (High-probability bound for the empirical capability ratio)**.** *Suppose Assumption D.8 holds. Let* $\epsilon_{\mathrm{batch}} = \epsilon_{\mathrm{batch}}(\delta)$ *be defined as in Eq.* (31)*, and assume* $\epsilon_{\mathrm{batch}} < p_t^{(j)}$*. Then with probability at least* $1-\delta$*,*

$$\frac{p_t^{(k)} - \epsilon_{\mathrm{batch}}}{p_t^{(j)} + \epsilon_{\mathrm{batch}}} \le \hat{\omega}_t^{(k,j)} \le \frac{p_t^{(k)} + \epsilon_{\mathrm{batch}}}{p_t^{(j)} - \epsilon_{\mathrm{batch}}}. \tag{39}$$

Moreover,

$$\left|\hat{\omega}_t^{(k,j)} - \bar{\omega}_t^{(k,j)}\right| \le \frac{\epsilon_{\mathrm{batch}}\left(p_t^{(j)} + p_t^{(k)}\right)}{p_t^{(j)}\left(p_t^{(j)} - \epsilon_{\mathrm{batch}}\right)}. \tag{40}$$

If, in addition, there exists a constant $\gamma > 0$ *such that* $p_t^{(j)} \ge \gamma$ *and* $\epsilon_{\mathrm{batch}} \le \gamma/2$*, then*

$$\left|\hat{\omega}_t^{(k,j)} - \bar{\omega}_t^{(k,j)}\right| \le \frac{4}{\gamma^2}\left(\sqrt{\frac{\log(4n/\delta)}{2B}} + \sqrt{\frac{\log(4n/\delta)}{2BG}}\right). \tag{41}$$

Proof. On the event from Lemma D.10, we have

$$p_t^{(a)} - \epsilon_{\mathrm{batch}} \le \hat{P}_t^{(a)} \le p_t^{(a)} + \epsilon_{\mathrm{batch}} \tag{42}$$

for every agent a. Since $\epsilon_{\mathrm{batch}} < p_t^{(j)}$, the denominator $\hat{P}_t^{(j)}$ is positive. Therefore,

$$\frac{p_t^{(k)} - \epsilon_{\mathrm{batch}}}{p_t^{(j)} + \epsilon_{\mathrm{batch}}} \le \frac{\hat{P}_t^{(k)}}{\hat{P}_t^{(j)}} \le \frac{p_t^{(k)} + \epsilon_{\mathrm{batch}}}{p_t^{(j)} - \epsilon_{\mathrm{batch}}}. \tag{43}$$

This proves Eq. (39).

For the deviation bound, write

$$\begin{aligned}
\left|\frac{\hat{P}_t^{(k)}}{\hat{P}_t^{(j)}} - \frac{p_t^{(k)}}{p_t^{(j)}}\right| &= \left|\frac{\hat{P}_t^{(k)} p_t^{(j)} - p_t^{(k)} \hat{P}_t^{(j)}}{\hat{P}_t^{(j)} p_t^{(j)}}\right| \\
&\le \frac{p_t^{(j)}\left|\hat{P}_t^{(k)} - p_t^{(k)}\right| + p_t^{(k)}\left|\hat{P}_t^{(j)} - p_t^{(j)}\right|}{\hat{P}_t^{(j)} p_t^{(j)}} \\
&\le \frac{\epsilon_{\mathrm{batch}}\left(p_t^{(j)} + p_t^{(k)}\right)}{p_t^{(j)}\left(p_t^{(j)} - \epsilon_{\mathrm{batch}}\right)}.
\end{aligned} \tag{44}$$

If $p_t^{(j)} \ge \gamma$ and $\epsilon_{\mathrm{batch}} \le \gamma/2$, then

$$p_t^{(j)} - \epsilon_{\mathrm{batch}} \ge \gamma/2. \tag{45}$$

Since rewards lie in $[0,1]$, we also have

$$p_t^{(j)} + p_t^{(k)} \le 2. \tag{46}$$

Combining Eqs. (44)–(46) with Eq. (31) proves Eq. (41). □

Definition D.12 (Empirical capability-aware baseline)**.** Replacing the oracle ratio in Definition D.4 with the batch-level empirical ratio gives

$$\hat{\mu}_{t,\mathrm{emp}}^{(k)} := \frac{1}{n}\sum_{j=1}^{n} \hat{\omega}_t^{(k,j)} \widetilde{P}_t^{(j)}. \tag{47}$$

Corollary D.13 (Baseline error induced by empirical ratios)**.** *Under the event in Lemma D.10, suppose* $\epsilon_{\mathrm{batch}} < p_t^{(j)}$ *for every denominator agent* j*. Then*

$$\left|\hat{\mu}_{t,\mathrm{emp}}^{(k)} - \mu_{t,\star}^{(k)}\right| \le \frac{1}{n}\sum_{j=1}^{n} \widetilde{P}_t^{(j)} \frac{\epsilon_{\mathrm{batch}}\left(p_t^{(j)} + p_t^{(k)}\right)}{p_t^{(j)}\left(p_t^{(j)} - \epsilon_{\mathrm{batch}}\right)}. \tag{48}$$

Since $\widetilde{P}_t^{(j)} \in [0,1]$*, the same bound also holds after dropping the multiplicative factor* $\widetilde{P}_t^{(j)}$ *from each summand.*

Proof. By Definitions D.4 and D.12,

$$\left|\hat{\mu}_{t,\mathrm{emp}}^{(k)} - \mu_{t,\star}^{(k)}\right| = \left|\frac{1}{n}\sum_{j=1}^{n}\left(\hat{\omega}_t^{(k,j)} - \bar{\omega}_t^{(k,j)}\right)\widetilde{P}_t^{(j)}\right|$$
$$\leq \frac{1}{n}\sum_{j=1}^{n}\widetilde{P}_t^{(j)}\left|\hat{\omega}_t^{(k,j)} - \bar{\omega}_t^{(k,j)}\right|. \tag{49}$$

Applying Theorem D.11 to each ratio gives Eq. (48). □

Corollary D.14 (Combined empirical baseline error)**.** *For any $\delta \in (0,1)$, define*

$$\epsilon_{\mathrm{batch}}^{\delta} := \epsilon_{\mathrm{batch}}(\delta/2). \tag{50}$$

Under Assumption D.8, suppose there exists $\gamma > 0$ such that $p_t^{(j)} \geq \gamma$ for every denominator agent j. If $\epsilon_{\mathrm{batch}}^{\delta} \leq \gamma/2$, then with probability at least $1-\delta$,

$$\left|\hat{\mu}_{t,\mathrm{emp}}^{(k)} - p_t^{(k)}\right| \leq \left(\frac{4}{\gamma^2} + \frac{1}{\gamma}\right)\epsilon_{\mathrm{batch}}^{\delta}. \tag{51}$$

Proof. Apply Lemma D.10 with failure probability $\delta/2$ to the empirical ratio. By Theorem D.11, the ratio error then satisfies

$$\left|\hat{\omega}_t^{(k,j)} - \bar{\omega}_t^{(k,j)}\right| \leq \frac{4}{\gamma^2}\epsilon_{\mathrm{batch}}^{\delta} \tag{52}$$

simultaneously for all denominator agents j. Applying the same batch-level Hoeffding argument to the empirical means in the baseline gives, with failure probability $\delta/2$,

$$\left|\widetilde{P}_t^{(j)} - p_t^{(j)}\right| \leq \epsilon_{\mathrm{batch}}^{\delta} \tag{53}$$

simultaneously for all $j = 1, \ldots, n$. By the union bound, the two events hold together with probability at least $1-\delta$.

On this joint event,

$$\left|\hat{\mu}_{t,\mathrm{emp}}^{(k)} - p_t^{(k)}\right| \leq \left|\hat{\mu}_{t,\mathrm{emp}}^{(k)} - \mu_{t,\star}^{(k)}\right| + \left|\mu_{t,\star}^{(k)} - p_t^{(k)}\right|$$
$$\leq \frac{4}{\gamma^2}\epsilon_{\mathrm{batch}}^{\delta} + \left|\frac{1}{n}\sum_{j=1}^{n}\bar{\omega}_t^{(k,j)}\left(\widetilde{P}_t^{(j)} - p_t^{(j)}\right)\right|$$
$$\leq \frac{4}{\gamma^2}\epsilon_{\mathrm{batch}}^{\delta} + \frac{1}{n}\sum_{j=1}^{n}\frac{p_t^{(k)}}{p_t^{(j)}}\epsilon_{\mathrm{batch}}^{\delta}. \tag{54}$$

Since rewards are bounded in $[0,1]$, $p_t^{(k)} \leq 1$, and $p_t^{(j)} \geq \gamma$ by the denominator lower-bound condition. Hence

$$\frac{1}{n}\sum_{j=1}^{n}\frac{p_t^{(k)}}{p_t^{(j)}}\epsilon_{\mathrm{batch}}^{\delta} \leq \frac{1}{\gamma}\epsilon_{\mathrm{batch}}^{\delta}. \tag{55}$$

Combining Eqs. (54) and (55) proves Eq. (51). □

Remark D.15 (Role of the concentration bound). Lemma D.10 gives finite-batch control of the empirical capability means used to form $\hat{\omega}_t^{(k,j)}$ at a fixed training step. The two terms in Eq. (31) correspond to the two sampling sources in a batch: the B prompt instances determine the prompt-distribution term $\epsilon_{\mathrm{prompt}}(\delta)$, while the BG response samples determine the response-sampling term $\epsilon_{\mathrm{resp}}(\delta)$. Theorem D.11 propagates this mean-reward error through the capability ratio under a positive denominator condition. Corollaries D.13 and D.14 then transfer the resulting ratio control to the empirical capability-aware baseline.

E Formulation and Pseudocode of HACPO

To facilitate a precise understanding of HACPO, we present the complete algorithmic formulation and training procedure.

Taking two agents (1 and 2) as an example. The optimization objective for agent 1 consists of two terms: the loss computed from its own samples, $J_{\text{homo}}(\theta)$, and the loss computed from samples of other agents, $J_{\text{hete}}(\theta)$. The final loss is the sum of these two terms. Similarly, agent 2 is updated using a loss function of the same form, but with different values.

$$\mathcal{J}_{homo}^{(1)} = \frac{1}{G}\sum_{i=1}^{G} \min\left(s_{t,i}^{(1,1)} \cdot A_{t,i}^{(1)}(y_{t,i}^{(1)}),\ clip(s_{t,i}^{(1,1)},\ 1-\epsilon_l,\ 1+\epsilon_h) \cdot A_{t,i}^{(1)}\right) \tag{56}$$

$\mathcal{J}_{homo}^{(1)}$ is the homo objective for Agent 1 using its own rollouts.

$$\mathcal{J}_{hete}^{(1)} = \frac{1}{G}\sum_{i=1}^{G} \left[clip(s_{t,i}^{(1,2)},\ 1.0-\delta+m\cdot\delta_{step},\ 1.0) \cdot sg(s_{t,i}^{(1,2)})^{\alpha} \cdot \omega_t^{(2,1)} \cdot A_{t,i}^{(1)}(y_{t,i}^{(2)})\right] \tag{57}$$

$\mathcal{J}_{hete}^{(1)}$ is the hete objective for Agent 1 using the rollouts from Agent 2.

$$A_{t,i}^{(1)}(y_{t,i}^{(1)}) = \frac{R(y_{t,i}^{(1)}) - \mu_t^{(1)}}{std\{\mathcal{R}_t(x)\}}, \quad A_{t,i}^{(1)}(y_{t,i}^{(2)}) = \frac{R(y_{t,i}^{(2)}) - \mu_t^{(1)}}{std\{\mathcal{R}_t(x)\}} \tag{58}$$

$$s_{t,i}^{(1,1)} = \left(\frac{\pi_\theta^{(1)}(y_{t,i}^{(1)})}{\pi_{\theta_{old}}^{(1)}(y_{t,i}^{(1)})}\right)^{\frac{1}{|y_{t,i}^{(1)}|}}, \quad s_{t,i}^{(1,2)} = \frac{\pi_\theta^{(1)}(y_{t,i}^{(2)})^{\frac{1}{L_1}}}{\pi_{\theta_{old}}^{(2)}(y_{t,i}^{(2)})^{\frac{1}{L_2}}} \tag{59}$$

Here, L_1 and L_2 respectively represent the length of response $y_{t,i}^{(2)}$ under tokenizers of agent 1 and 2.

$$\mathcal{J}^{(1)} = \mathcal{J}_{homo}^{(1)} + \mathcal{J}_{hete}^{(1)} \tag{60}$$

The final optimization objective is the sum of homogeneous and heterogeneous objective.

F Additional Experimental Results

Table 8: MATH500 accuracy across five random seeds, evaluated with a maximum response length of 8192 tokens. The final column reports the mean and sample standard deviation ($n-1$ denominator).

Method	0	1	42	1337	3407	Mean ± SD
Qwen3-1.7B-Base						
HACPO	**0.680**	**0.682**	**0.690**	**0.681**	**0.670**	0.6806 ± 0.0071
GSPO	0.647	0.640	0.648	0.646	0.643	0.6448 ± 0.0033
GSPO×2	0.648	0.651	0.664	0.652	0.646	0.6522 ± 0.0070
Qwen3-4B-Base						
HACPO	**0.788**	**0.803**	**0.808**	**0.797**	**0.799**	0.7990 ± 0.0074
GSPO	0.765	0.769	0.782	0.768	0.768	0.7704 ± 0.0067
GSPO×2	0.769	0.774	0.756	0.777	0.775	0.7702 ± 0.0085

F.1 Three More Model Combinations

Here, we present additional experiments in Table 11, including comparisons between Qwen3-4B-Base + Qwen3-8B-Base, Llama3.2-1B-Instruct + Llama3.2-3B-Instruct, and Qwen3-1.7B-Base + Llama3.2-1B-Instruct.

Algorithm 1 Heterogeneous Agent Collaborative Policy Optimization

Require: n initial policy models $\pi_{\theta_1}, \pi_{\theta_2}, ...\pi_{\theta_n}$, reward models R, task prompts $\mathcal{D}$, each prompt has G outputs. The training step is t. Each step has M policy updates.

for k = 1 to n **do**
  rollout policy model $\pi^{(k)}_{\theta_{old}} \leftarrow \pi_{\theta_k}$
end for
for t = 1 to N **do**
  Sample a batch $\mathcal{D}_t$ from $\mathcal{D}$
  for k = 1 to n **do**
    Update the rollout policy model $\pi^{(k)}_{\theta_{\text{old}}} \leftarrow \pi^{(k)}_{\theta}$
  end for
  for k = 1 to n **do**
    Sample G output $y \sim \pi^{(k)}_{\theta_{\text{old}}}(\cdot \mid x)$ for each question $x \in \mathcal{D}_t$
    Compute rewards $R(y_i)$ for each output y_i in the batch
    Compute accuracy for the sampling model
  end for
  for k = 1 to n **do**
    Compute $A_{t,i}(y)$ for the response y in batch (agent k)
    for mini batch = 1 to M **do**
      Update the policy model $\pi^{(k)}_{\theta}$ by maximizing the HACPO objective
    end for
  end for
end for

Ensure: $\pi^{(k)}_{\theta} | k = 1, 2, ..., n$

Table 9: Runtime Comparison

Method	Qwen3-1.7B-Base	Qwen3-4B-Base	Total Time
GSPO	1h 31m	2h 38m	4h 9m
GSPO updates×2	2h 6m	2h 59m	5h 5m
GSPO×2	2h 43m	4h 1m	6h 44m
HACPO		Overall 5h 31m	

Table 10: GPU Memory Usage

Method	Qwen3-1.7B-Base	Qwen3-4B-Base
GSPO	76.8%	82.7%
GSPO updates×2	76.6%	80.3%
GSPO×2	82.2%	84.2%
HACPO	Overall 86.0%	

F.2 The Performance over Different Seeds

We evaluate Qwen3-1.7B-Base and Qwen3-4B-Base on MATH500 across five random seeds (0, 1, 42, 1337, and 3407). All checkpoints are evaluated with a maximum response length of 8192 tokens, matching the main results, while the training response-length limit remains 4096. The seed-42 entries match the corresponding MATH500 results in Table 1. Table 8 reports individual runs and their mean $\pm$ sample standard deviation.

Across all five seeds, HACPO consistently outperforms both GSPO and the resource-equivalent GSPO×2 baseline. On Qwen3-1.7B-Base, HACPO achieves $68.06\% \pm 0.71\%$, compared with $64.48\% \pm 0.33\%$ for GSPO and $65.22\% \pm 0.70\%$ for GSPO×2. On Qwen3-4B-Base, HACPO achieves $79.90\% \pm 0.74\%$, compared with $77.04\% \pm 0.67\%$ and $77.02\% \pm 0.85\%$, respectively. The average gains over GSPO×2 are 2.84 and 2.88 percentage points. Two-sided paired t-tests on the five seed-matched differences yield $p < 0.001$ for 1.7B and $p = 0.009$ for 4B. These tests assess the

null hypothesis of zero mean improvement, assuming independent runs and approximately normally distributed paired differences; the significance results apply to this model pair on MATH500.

F.3 The GPU Peak Memory and Overall Runtime

We report the runtime (Table 9) and peak GPU memory utilization ratio (Table 10) for the Qwen3-1.7B-Base and Qwen3-4B-Base combination. We added a baseline GSPO updates $\times$ 2, which fixed the sampling quantity and doubled the number of updates (i.e., both the sampling cost and the update cost were the same as those of HACPO).

Table 11: Additional Experimental Results

Model	MATH-500	math	gsm8k	aime2025	AMC23	minerva	olympiad	AVG
Qwen3-4B-Base and Qwen3-8B-Base								
4B-Base	0.61	0.676	0.445	0.1	0.4	0.308	0.347	0.412
4B-Base(GRPO)	0.796	0.788	0.885	**0.307**	0.475	0.349	0.454	0.579
4B-Base(GSPO)	0.782	0.787	0.877	0.25	0.525	**0.368**	0.46	0.578
4B-Base(GSPO$\times$2)	0.756	0.794	0.873	0.208	0.55	0.382	0.463	0.575
4B-Base(Naive)	0.734	0.712	0.895	0.143	0.55	0.342	0.354	0.526
4B-Base(HACPO)	**0.81**	**0.803**	**0.904**	0.275	**0.6**	0.364	**0.463**	**0.603**
8B-Base	0.647	0.713	0.684	0.033	0.4	0.232	0.375	0.441
8B-Base(GRPO)	0.814	0.812	0.921	0.265	0.575	0.415	**0.479**	0.612
8B-Base(GSPO)	0.794	0.804	0.923	0.225	0.6	**0.426**	0.468	0.606
8B-Base(GSPO$\times$2)	0.8	0.803	0.92	0.2	0.575	0.404	0.46	0.595
8B-Base(Naive)	0.79	0.783	0.921	0.252	0.5	0.408	0.429	0.583
8B-Base(HACPO)	**0.828**	**0.813**	**0.933**	**0.323**	**0.625**	0.423	0.467	**0.63**
Llama3.2-1B-Instruct and Llama3.2-3B-Instruct								
Llama3.2-1B	0.176	0.297	0.489	0	0.15	0.052	0.061	0.18
Llama3.2-1B(GRPO)	0.35	0.349	0.569	0	0.125	0.008	0.097	0.214
Llama3.2-1B(GSPO)	**0.356**	0.346	0.523	0.021	0.125	0.066	0.088	0.218
Llama3.2-1B(GSPO$\times$2)	0.352	0.349	**0.573**	0.07	0.125	0.079	**0.103**	0.227
Llama3.2-1B(Naive)	0.284	0.302	0.45	0.0	0.025	0.066	0.073	0.171
Llama3.2-1B(HACPO)	0.35	**0.352**	0.541	**0.022**	**0.2**	**0.081**	0.085	**0.233**
Llama3.2-3B	0.267	0.441	0.788	0.0	0.2	0.169	0.158	0.289
Llama3.2-3B(GRPO)	0.502	0.507	0.814	0.0	0.25	0.199	0.174	0.349
Llama3.2-3B(GSPO)	0.512	0.501	0.812	0.054	0.225	0.184	0.17	0.351
Llama3.2-3B (GSPO$\times$2)	0.488	0.498	**0.829**	0.0	0.175	0.188	0.159	0.334
Llama3.2-3B(Naive)	0.406	0.407	0.734	0.0	0.225	0.177	0.107	0.294
Llama3.2-3B(HACPO)	**0.522**	**0.51**	0.828	**0.067**	**0.275**	**0.199**	**0.188**	**0.37**
Qwen3-1.7B-Base and Llama3.2-1B-Instruct								
Qwen3	0.5	0.483	0.616	0.033	0.3	0.206	0.229	0.338
Qwen3(GRPO)	**0.682**	0.652	0.824	0.16	0.375	0.272	0.298	0.466
Qwen3(GSPO)	0.648	0.641	0.826	0.148	0.45	0.272	0.287	0.467
Qwen3(GSPO$\times$2)	0.664	0.65	0.829	0.177	0.375	0.265	0.293	0.475
Qwen3(Naive)	0.59	0.596	0.798	0.105	0.3	0.221	0.241	0.407
Qwen3(HACPO)	0.676	**0.661**	**0.838**	**0.22**	**0.45**	**0.305**	**0.32**	**0.496**
Llama3.2	0.176	0.297	0.489	0.033	0.15	0.052	0.061	0.18
Llama3.2(GRPO)	0.35	0.349	**0.569**	0	0.125	0.008	0.097	0.214
Llama3.2(GSPO)	0.356	0.346	0.523	0.021	0.125	0.066	0.088	0.218
Llama3.2(GSPO$\times$2)	0.352	0.349	0.573	0.07	0.125	0.079	**0.103**	0.227
Llama3.2(Naive)	0.336	0.337	0.512	0.0	0.125	0.066	0.071	0.214
Llama3.2(HACPO)	**0.356**	**0.368**	0.533	**0.033**	**0.15**	**0.066**	0.091	**0.228**

F.4 Distillation and Frozen-Model Baselines

We compare HACPO with one-way learning baselines using Qwen3-1.7B-Base and Qwen3-4B-Base. On-policy distillation (OPD) Agarwal et al. [2024] and OPD+GRPO update only the 1.7B model, using the frozen 4B base model as the teacher. HACPO (Frozen Strong) also freezes the 4B model, while updating the 1.7B model with HACPO using their shared rollouts. Full HACPO updates both models. Table 12 reports the resulting 1.7B model's performance on the seven benchmarks; the general training and evaluation settings are described in Appendix A.

HACPO achieves an average accuracy of 49.3%, compared with 40.6% for OPD, 40.8% for OPD+GRPO, and 45.2% for HACPO (Frozen Strong). In this setting, the improvement over the frozen-strong variant supports the benefit of jointly updating both models and allowing their rollout

Table 12: Distillation and frozen-model baselines for Qwen3-1.7B-Base, with Qwen3-4B-Base as the stronger model. AVG is the mean accuracy across seven benchmarks.

Method	MATH-500	MATH	GSM8K	AIME2025	AMC23	Minerva	Olympiad	AVG
OPD	0.620	0.582	0.700	0.033	0.400	0.254	0.252	0.406
OPD+GRPO	0.610	0.584	0.704	0.067	0.375	0.250	0.263	0.408
HACPO (Frozen Strong)	0.672	0.643	0.829	0.033	0.400	0.279	0.308	0.452
HACPO	0.690	0.674	0.822	0.225	0.450	0.279	0.314	**0.493**

distributions to evolve during training. These comparisons provide evidence that one-way transfer from a fixed base-model teacher does not fully account for HACPO's gains.

F.5 Sequential Train-then-Distill

We further compare HACPO with a sequential strategy that first improves the stronger model before using it as a teacher. We train Qwen3-4B-Base with GSPO×2, freeze the resulting checkpoint, and then distill Qwen3-1.7B-Base using OPD+GRPO. The final portfolio consists of this frozen 4B checkpoint and the distilled 1.7B model. Consequently, the 4B results for Train-then-Distill in Table 13 are the GSPO×2 results from Table 1.

Table 13: HACPO versus sequential Train-then-Distill. Parenthesized improvements are absolute accuracy differences relative to Train-then-Distill.

Method	MATH-500	MATH	GSM8K	AIME2025	AMC23	Minerva	Olympiad	AVG
Qwen3-1.7B-Base								
Train-then-Distill	0.674	0.663	0.814	0.150	0.350	0.283	0.284	0.460
HACPO	0.690	0.674	0.822	0.225	0.450	0.279	0.314	**0.493 (+0.033)**
Qwen3-4B-Base								
Train-then-Distill	0.756	0.794	0.873	0.208	0.550	0.382	0.463	0.575
HACPO	0.808	0.801	0.903	0.267	0.575	0.386	0.467	**0.601 (+0.026)**

HACPO improves average accuracy by 3.3 percentage points for the 1.7B model and 2.6 percentage points for the 4B model. Under the same hardware configuration, sequential Train-then-Distill takes 6h 26m in total: 4h 01m to train the 4B model and 2h 25m to distill the 1.7B model. HACPO obtains the complete two-model portfolio in 5h 31m, reducing end-to-end training time by 14.2% in this comparison.

Our HACPO implementation loads, runs, and offloads the models one at a time during rollout generation, and likewise loads, updates, and offloads them sequentially during optimization. This execution avoids keeping both models resident in GPU memory simultaneously. As reported in Table 10, HACPO's peak GPU memory utilization is 86.0%, compared with 84.2% for training the 4B model alone with GSPO×2. The latter measures the large-model training stage; it does not measure the peak memory of the complete Train-then-Distill pipeline.

F.6 Complementary Rollouts from the Weaker Model

We analyze the collected rollouts from the Qwen3-1.7B-Base/Qwen3-4B-Base experiment to examine how a weaker model can contribute useful training data to a stronger partner. For 627 of 7,424 problems (8.4%), the 1.7B model has a higher sampled success rate than the 4B model. Here, the sampled success rate is the fraction of a model's collected rollouts for a given problem that pass verification. Thus, lower average capability does not imply a lower sampled success rate on every problem.

We also observe problems for which the 4B model generates no correct rollout while the 1.7B model generates at least one. These weak-only successes supply verified solutions absent from the stronger model's own sampled rollouts. Together with the 4B model's performance improvement under HACPO, this analysis provides evidence of complementary information from the weaker model. It does not separately quantify the contribution of alternative reasoning paths or incorrect trajectories to the final improvement.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the claims about the HACRL problem and the HACPO algorithm, which are well-supported by the theoretical proofs in Appendix D.1 and the empirical results in Section 4. Furthermore, we demonstrate the generalizability of our claims by comprehensively evaluating HACPO across three types of agent heterogeneity (state, size, and model), six two-agent combinations and one three-agent combination, seven diverse mathematical reasoning benchmarks and five seeds for the Qwen3-1.7B-Base/Qwen3-4B-Base combination on MATH500, proving that our method is robust and extends well across various heterogeneous settings.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We explicitly discuss the limitations in Section 7, noting that our empirical evaluation is currently restricted to two- and three-agent scenarios due to engineering bottlenecks within the verl framework and computational resources, and outline scaling as future work.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The proof of oracle baseline unbiasedness (Theorem D.5) is provided in Appendix D.1; the required assumptions and finite-batch capability ratio error bounds are given in Appendix D.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We describe the HACRL problem and HACPO algorithm in the main text and specify the models, datasets, and hyperparameters in Appendix A. Our implementation, based on verl, is publicly available at `https://github.com/zzx-peter/HACRL-code`.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.

(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The HACPO implementation is publicly available at `https://github.com/zzx-peter/HACRL-code`, with installation and launch instructions in the repository README. The models and datasets used in our experiments are publicly available and identified in Appendix A.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Detailed experimental settings, including maximum prompt/response lengths, batch sizes and clipping parameters for different combinations, are specified in Appendix A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: To ensure statistical significance and robustness against random initialization, we evaluate the Qwen3-1.7B-Base/Qwen3-4B-Base combination on MATH500 across five different random seeds (0, 1, 42, 1337, and 3407), with the detailed performance distributions reported in Appendix F.2 to account for variance.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: Appendix A specifies eight NVIDIA A800 GPUs with 80 GB each. Appendix F.3 reports runtime and peak memory utilization for the Qwen3-1.7B-Base/Qwen3-4B-Base combination; compute profiles for all settings and total project compute are not reported.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

Justification: This research focuses on foundational algorithmic improvements in reinforcement learning (RLVR) and conforms entirely to the NeurIPS Code of Ethics, with no intent for malicious application.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [No]

Justification: As this is a foundational machine learning algorithmic study focused on optimizing reinforcement learning efficiency, it does not have direct, immediate negative societal applications. Therefore, a dedicated broader impacts section was not included.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: This work focuses on a training paradigm and does not release new pre-trained models, high-risk systems, or scraped datasets.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [No]

    Justification: We use publicly available datasets, open-weight models, and the verl framework and cite their sources. Asset-specific license names and terms of use are not exhaustively listed in the paper.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Our HACPO implementation modifies verl and is publicly released at `https://github.com/zzx-peter/HACRL-code`; the paper results correspond to the main branch. The README documents installation, launching, and algorithmic changes, and the repository includes an Apache-2.0 license.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The research does not involve crowdsourcing or human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The research does not involve human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: Qwen3, Llama3.2, and Gemma3 serve as the agents trained and evaluated in our method; their roles and experimental settings are described in Section 3 and Appendix A. LLM assistance in manuscript preparation was limited to writing, editing, and formatting.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.