Dr. MAS: Stable Reinforcement Learning for Multi-Agent LLM Systems

Lang Feng[1] **Longtao Zheng**[1] **Shuo He**[1] **Fuxiang Zhang**[1] **Bo An**[1*]
[1]Nanyang Technological University, Singapore
lang005@e.ntu.edu.sg, boan@ntu.edu.sg

Project Page: https://langfengq.github.io/DrMAS

Abstract

Multi-agent LLM systems improve reasoning and tool use via role specialization, yet reinforcement learning (RL) post-training for such systems remains unstable and underexplored. We theoretically pinpoint a key source of instability when extending group-based RL to cooperative multi-agent LLM systems: under GRPO-style optimization, a global normalization baseline can mismatch heterogeneous agents' reward distributions, inducing gradient-norm instability. Based on this finding, we propose Dr. MAS, a simple and stable RL recipe for cooperative multi-agent LLM systems. Algorithmically, Dr. MAS normalizes advantages per agent using each agent's own reward statistics, which calibrates gradient scales and dramatically stabilizes the training. Systemically, Dr. MAS provides an end-to-end multi-agent RL framework with scalable orchestration, flexible per-agent LLM serving and optimization, and shared resource scheduling of actor backends. Unlike single-actor frameworks such as veRL, Dr. MAS natively supports multiple heterogeneous LLMs over a unified GPU pool. Lifecycle-aware backend management and dynamic dispatch release inactive models' GPU memory and schedule active agents on demand, enabling hardware-efficient co-training. We evaluate Dr. MAS on multi-agent math reasoning and multi-turn search with Qwen2.5 and Qwen3. Dr. MAS achieves clear gains over vanilla GRPO (e.g., +5.6% avg@16 and +4.6% pass@16 on math, and +15.2% avg@16 and +13.1% pass@16 on search) while largely eliminating gradient spikes. It also remains highly effective under heterogeneous agent-model assignments while improving efficiency.

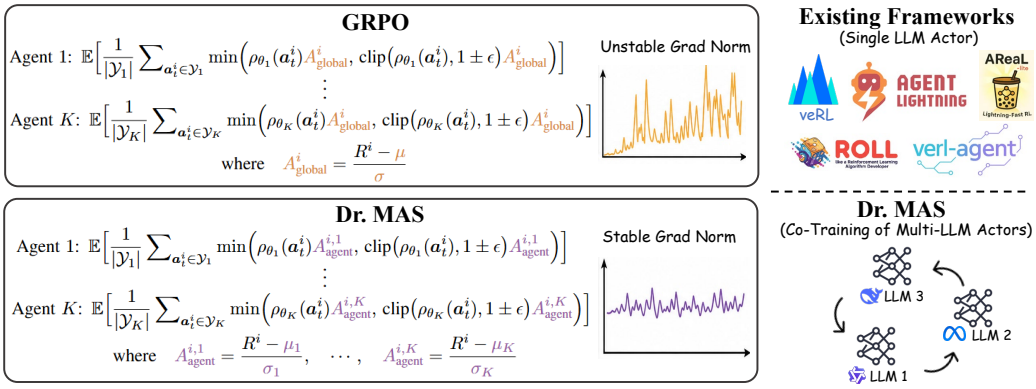


Figure 1: **Left** (algorithm): GRPO with global baseline can cause unstable gradient norm; Dr. MAS with per-agent normalization stabilizes the training of MAS. **Right** (system): Existing RL frameworks mainly support single LLM actors, while Dr. MAS enables co-training of multiple LLM actors.

*Corresponding author

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

1 Introduction

Large Language Models (LLMs) [1, 2, 3, 4] have shown strong reasoning abilities in tasks such as question answering and code generation. Yet many real-world applications, including complex information retrieval [5, 6], agentic software engineering [7, 8], and open-ended device control [9, 10], require multiple agents to coordinate over long horizons through planning, search, verification, and refinement. Organizing LLMs into Multi-Agent Systems (MAS), where each agent specializes in a subtask, has therefore become an important paradigm for solving complex real-world tasks [11, 12].

Reinforcement Learning (RL) [13] has become central to LLM post-training, but RL for multi-agent LLM systems remains underexplored algorithmically and systemically. *Algorithmically*, group-based methods like GRPO [14] avoid learned critics by estimating a global baseline from final group scores. However, in MAS, agents are invoked with different frequencies and contexts and their reward distributions are heterogeneous; a single global baseline can therefore be miscalibrated for individual agents and induce unstable gradient scales. *Systemically*, large-scale RL frameworks such as veRL [15], ROLL [16], and AReaL [17] provide efficient LLM training pipelines, but are largely built around a single LLM actor and lack native support for scalable multi-agent orchestration, heterogeneous agent-model assignment, and resource-efficient co-training of multiple LLMs.

In this work, we study RL post-training for cooperative multi-agent LLM systems from both perspectives. *At the algorithmic level*, we *theoretically* show that vanilla GRPO suffers from a baseline mismatch in MAS: a global advantage baseline over heterogeneous agent steps can deviate from each agent's reward statistics, inflate the per-agent gradient second moment, and trigger gradient-norm spikes. Based on this, we propose Dr. MAS, a simple and stable recipe that replaces the global baseline with agent-wise baselines. By normalizing each agent's advantages using statistics from its own active timesteps, Dr. MAS calibrates advantage scales across heterogeneous roles and removes the dominant gradient-inflation term. *At the systems level*, Dr. MAS provides an end-to-end RL framework for multi-agent LLM post-training. It supports pluggable orchestration for diverse workflows, flexible agent-model assignment with optional LLM sharing or heterogeneous non-sharing, and per-agent optimization configurations. To make multi-LLM training practical, Dr. MAS co-locates all LLM worker groups on a unified GPU resource pool, inspired by resource-bundle scheduling like Ray placement groups, and applies lifecycle-aware actor backend management: inactive models release GPU memory and KV caches, while active agents are dynamically scheduled. This avoids naive one-agent-one-GPU-pool allocation and enables efficient co-training of multiple LLM agents.

We evaluate Dr. MAS on role-specialized MAS for math reasoning and multi-turn search with Qwen2.5 [18] and Qwen3 [4], under both LLM-sharing and non-sharing settings. Across tasks and settings, Dr. MAS consistently outperforms vanilla GRPO, achieving relative improvements on math of by +5.6% avg@16 and +4.6% pass@16, and on search by +15.2% avg@16 and +13.1% pass@16, while largely eliminating gradient-norm spikes. It also remains effective under heterogeneous agent-model assignments, enabling smaller models for lower-level decisions while improving overall system efficiency.

Our contributions are:

- **A theoretical foundation for agent-wise baselines in cooperative multi-agent LLM RL.** We show, to our knowledge for the first time, that GRPO-style optimization in multi-agent LLM systems can inflate the per-agent gradient second moment and cause instability, and we derive an agent-wise remedy that resolves this issue.
- **A scalable multi-agent LLM post-training framework beyond reward engineering.** Dr. MAS integrates pluggable orchestration, flexible agent-model assignment, per-agent configurations, and lifecycle-aware GPU resource pooling, enabling hardware-efficient co-training of heterogeneous multi-agent LLMs under a unified RL pipeline.

2 Related Work

Reinforcement Learning for LLMs. Beyond early alignment-focused approaches such as RLHF [19, 20, 21, 22], recent work studies Reinforcement Learning from Verifiable Rewards (RLVR), which leverages automatically checkable signals (e.g., correctness in math or code) to improve LLM capabilities [23]. Within this setting, group-based RL has emerged as a strong alternative to classical

actor-critic algorithms like PPO [24]. Techniques such as GRPO [14], RLOO [25, 26], Dr. GRPO [27], DAPO [28], and GSPO [29] aggregate multiple rollouts for the same query and perform relative comparisons within the group, thereby avoiding explicit value-function learning. RLVR has also been extended to agentic, multi-turn settings where LLMs act as automatic agents [30, 31, 32, 33, 34, 35]. GRPO-style training has been widely applied to tool use [36, 37], OS control [38], and gaming [39]. Further variants refine group construction or objectives, such as GiGPO [40] and ARPO [41].

Reinforcement Learning for Multi-Agent LLMs. Recent RL post-training has expanded from single-agent scenarios to *learning coordination* in role-specialized, multi-turn multi-agent systems. Self-play training (e.g., SPIRAL [42], and MARSHAL [43]) use multi-turn dynamics to generate curricula and rewards with minimal manual labeling. However, these approaches are often confined to dyadic (two-agent) self-play scenarios. To ease deployment, Chain-of-Agents [44] distills multi-agent trajectories into a single agentic policy. Meanwhile, group-relative optimization has been extended to multi-agent settings [45, 46, 47, 48, 49, 50, 51, 52], but these methods typically rely on heuristics and lack stability guarantees. Dr. MAS distinguishes itself by theoretically identifying *gradient-norm inflation* as the root cause of instability and proposing a simple yet rigorous agent-wise solution.

Reinforcement Learning Infrastructure. As RL post-training scales, infrastructure has shifted toward optimizing the end-to-end *rollout-train loop*, where throughput, scheduling, and variable-length generation dominate system efficiency. General-purpose stacks such as veRL [15], OpenRLHF [53], ROLL [16], slime [54], and AReaL [17] increasingly provide modular pipeline abstractions and distributed execution to improve utilization under heavy sampling. As agentic use cases grow, frameworks increasingly emphasize multi-turn rollout and tool integration (e.g., verl-agent [40], VerlTool [55], Agent-Lightning [56]), with MARTI [57] and PettingLLMs [58] offering a practical multi-agent training interface. However, they either provide limited support for heterogeneous model assignments or lack a shared resource pool for efficient scheduling. Our Dr. MAS addresses both to improve scalability and utilization in MAS post-training.

3 Preliminaries

3.1 Multi-Agent LLMs

We consider a cooperative multi-agent LLM system consisting of K distinct LLM agents $\pi_{\theta_1}, \pi_{\theta_2}, \ldots, \pi_{\theta_K}$, each parameterized by its own LLM weights θ_k. The agents jointly engage in solving complex tasks sampled from a distribution $x \in p(X)$. Each full interaction process (trajectory) produces a single outcome reward $R \in \mathbb{R}$ to indicate success or failure. During task completion, the agents' joint interaction unfolds as a trajectory $\boldsymbol{\tau} = \{(\boldsymbol{s}_1, \boldsymbol{a}_1, k_1), (\boldsymbol{s}_2, \boldsymbol{a}_2, k_2), \ldots, (\boldsymbol{s}_T, \boldsymbol{a}_T, k_T)\}$, where $\boldsymbol{s}_t$ denotes the conversational or contextual state (e.g., dialogue history, task prompt, or shared memory) at execution step t, $\boldsymbol{a}_t$ is the text output produced, and $k_t \in 1, \ldots, K$ denotes which LLM agent was active at step t. The active agent can change dynamically across the trajectory, for instance, in a hierarchical multi-agent framework, a high-level planner LLM may decide which sub-agent executes at each step. Hence, the index k_t explicitly denotes the identity of the agent executing at each step. The execution LLM agent k_t generates its output based on its policy $a_t \sim \pi_{\theta_{k_t}}(\cdot \mid s_t)$. Depending on the system design, the agents may share parameters (i.e., $\theta_1 = \cdots = \theta_K$) , differing in role-specific prompts, enabling efficient adaptation under a unified LLM, or they may maintain distinct parameters ($\theta_i \neq \theta_j$) to specialize in heterogeneous sub-tasks, allowing diverse reasoning capabilities across agents.

3.2 Group Relative Policy Optimization

Group-based RL methods like Group Relative Policy Optimization (GRPO) [14] optimize policies by comparing multiple rollouts generated from the same task instruction and normalizing their rewards within each group, thereby avoiding explicit value-function estimation. Formally, given a task instruction x, the multi-agent LLM system samples a set of N trajectories

$$\{\tau^i = (s_1^i, a_1^i, k_1^i, \ldots, s_{T_i}^i, a_{T_i}^i, k_{T_i}^i)\}_{i=1}^N, \tag{1}$$

where $k_t^i \in \{1, \ldots, K\}$ denotes the active agent at step t of trajectory i. Each trajectory τ^i receives a scalar terminal reward $R^i = R(\tau^i) \in \mathbb{R}$ that measures the overall quality of the generated outcome. The normalized advantage for each trajectory is computed using the group's mean and standard

deviation:

$$A^i_{\text{global}} = \frac{R^i - \mu}{\sigma}, \quad \mu = \frac{1}{N}\sum_{i=1}^{N} R^i, \quad \sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(R^i - \mu)^2. \tag{2}$$

This advantage is then propagated to all agent outputs that contributed to the trajectory. Formally, we define the set of outputs of agent k as $\mathcal{Y}_k = \{a^i_t \mid k^i_t = k\}$, i.e., the collection of all time steps (i, t) across the group at which agent k produces an action. Notably, agents are often invoked at different frequencies, which results in varying sample sizes $|\mathcal{Y}_k|$. The RL objective for agent k is then given by

$$\mathcal{J}_k(\theta_k) = \mathbb{E}_{x\sim p(x)}\Big[\frac{1}{|\mathcal{Y}_k|}\sum_{\boldsymbol{a}^i_t \in \mathcal{Y}_k} \min\Big(\rho_{\theta_k}(\boldsymbol{a}^i_t)A^i_{\text{global}},\ \text{clip}\big(\rho_{\theta_k}(\boldsymbol{a}^i_t), 1\pm\epsilon\big)A^i_{\text{global}}\Big)\Big], \tag{3}$$

where $\rho_{\theta_k}(\boldsymbol{a}^i_t) = \frac{\pi_{\theta_k}(\boldsymbol{a}^i_t|\boldsymbol{s}^i_t)}{\pi_{\theta_k^{\text{old}}}(\boldsymbol{a}^i_t|\boldsymbol{s}^i_t)}$ is the importance sampling ratio. Here, we omit the KL-divergence regularization for notational brevity.

4 Methodology

In a multi-agent LLM system, different agents often specialize in distinct functions (e.g., information retrieval vs. answer synthesis, high-level planning vs. low-level execution), and consequently can exhibit substantially different reward distributions. We find that using vanilla GRPO with the global baseline (μ, σ) for all agents can be suboptimal: some agents may consistently operate in reward distributions above the global mean, while others remain below it. This persistent bias in how advantages are normalized can introduce a deterministic shift in the effective advantages seen by each agent, which in turn can inflate gradient-estimator variance and destabilize training.

In this section, we introduce Dr. MAS by (**1**) theoretically formalizing the instability and analyzing the second moment of the per-agent gradient under GRPO optimization (Section 4.1); (**2**) proposing an agent-wise remedy that calibrates each agent's advantage using its own reward statistics, thereby improving the training stability (Section 4.2); and (**3**) describing a system framework that implements an efficient end-to-end RL training recipe for multi-agent LLM systems (Section 4.3). The complete pseudo code of Dr. MAS is provided in Appendix E.

4.1 Risk of Gradient Norm Explosion

To focus on how advantage normalization causes the instability in MAS training, we perform a theoretical analysis of the gradient norm. We base our analysis on the unclipped GRPO gradient (clipping and other regularization only further bound the update and are orthogonal to the gradient issue we study). For each agent k, and for each step (i, t) such that $k^i_t = k$, we define the *score function* as $z^{(k)}_{i,t} \triangleq \nabla_{\theta_k}\rho_{\theta_k}(\boldsymbol{a}^i_t)$ and corresponding (unclipped) GRPO gradient contribution as

$$\tilde{g}^{\text{global}}_k \triangleq \frac{R^i - \mu}{\sigma} z^{(k)}_{i,t}. \tag{4}$$

Here, (μ, σ) are the mean and standard deviation used by vanilla GRPO. We assume that each agent's score function has a bounded second moment:

Assumption 4.1. *For each agent k, there exists a constant $C_k < \infty$ such that $0 \le \mathbb{E}\big[\,\|z^{(k)}_{i,t}\|^2\,\big] \le C_k$.*

Then we can express the second moment of the per-agent gradient as follows.

Lemma 4.2. *Under Assumptions 4.1, for any agent k,*

$$\mathbb{E}_{\boldsymbol{a}^i_t\sim\mathcal{Y}_k}\big[\|\tilde{g}^{\text{global}}_k\|^2\big] = \mathbb{E}_{\boldsymbol{a}^i_t\sim\mathcal{Y}_k}\big[\|z^{(k)}_{i,t}\|^2\big]\ \frac{\sigma_k^2 + (\mu_k - \mu)^2}{\sigma^2} + \Delta_k,$$

where $\mu_k \triangleq \frac{1}{|\mathcal{Y}_k|}\sum_{\boldsymbol{a}^i_t\in\mathcal{Y}_k} R^i$, $\sigma_k^2 \triangleq \frac{1}{|\mathcal{Y}_k|}\sum_{\boldsymbol{a}^i_t\in\mathcal{Y}_k}(R^i - \mu_k)^2$ are the mean and variance when sampling time steps uniformly from $\mathcal{Y}_k$ (i.e., when agent k is active). Δ_k is a score-reward covariance correction term.

See Appendix B.1 for the proof. Lemma 4.2 separates the per-agent gradient norm into a dominant scaling factor and a residual covariance correction. The multiplicative factor $\big(\sigma_k^2 + (\mu_k - \mu)^2\big)/\sigma^2$

grows when agent k operates in a reward distribution whose mean is far from the global mean or agent k's conditional reward variance is much larger than the global variance. The term Δ_k captures the residual score-reward correlation. In large-scale LLM training, rewards are typically low-dimensional signals of final task quality (e.g., pass/fail for reasoning, correctness for coding), while $z_{i,t}^{(k)}$ depends mainly on the local token-level stochasticity of the policy. Empirically, their covariance is often much smaller than the dominant scaling factor $\mathbb{E}_k[\|z_{i,t}^{(k)}\|^2](\sigma_k^2 + (\mu_k - \mu)^2)/\sigma^2$. This decomposition reveals the intrinsic instability of global normalization of GRPO in heterogeneous multi-agent training: a large deviation in the dominant scaling factor can inflate the gradient and lead to unstable updates. We formalize this phenomenon below.

Proposition 4.3 (Gradient-Norm Inflation)**.** *As either the normalized mean deviation $|\mu_k - \mu|/\sigma$ or the normalized variance ratio σ_k^2/σ^2 becomes large, the second moment of $\tilde{g}_k^{\text{global}}$ grows at least linearly. Consequently, along any training process for which there exists a sequence of iterations indexed by m such that*

$$\frac{\sigma_{k,m}^2 + (\mu_{k,m} - \mu_m)^2}{\sigma_m^2} \to \infty, \quad \mathbb{E}\big[\|\tilde{g}_m^{\text{global}}\|^2\big] \to \infty,$$

where $\tilde{g}_m^{\text{global}} = (\tilde{g}_{1,m}^{\text{global}}, \ldots, \tilde{g}_{K,m}^{\text{global}})$ stacking all LLM agents' gradients.

The proof is provided in Appendix B.2. Proposition 4.3 demonstrates that gradient-norm inflation can be triggered by *any* agent whose reward distribution is poorly aligned with the global baseline. In practice, the gradient norms in such cases typically do not reach mathematical infinity. However, they often grow large enough and trigger severe gradient spikes, thus destabilizing the training process of the entire multi-agent LLM system.

4.2 Agent-Wise Remedy

Fortunately, Proposition 4.3 suggests a straightforward and effective remedy: calibrating each agent's advantages using reward statistics computed exclusively on the steps where that agent is active. Specifically, we replace the global baseline (μ, σ) with (μ_k, σ_k), which ensures that $(\sigma_k^2 + (\mu_k - \mu)^2)/\sigma^2 = 1$. In practice, this corresponds to normalizing each agent's reward using its own empirical mean and variance:

$$A_{\text{agent}}^{i,k} = \frac{R^i - \mu_k}{\sigma_k}, \tag{5}$$

where $\mu_k \triangleq \frac{1}{|\mathcal{Y}_k|} \sum_{\boldsymbol{a}_t^i \in \mathcal{Y}_k} R^i$ and $\sigma_k^2 \triangleq \frac{1}{|\mathcal{Y}_k|} \sum_{\boldsymbol{a}_t^i \in \mathcal{Y}_k} (R^i - \mu_k)^2$. Since $\mathcal{Y}_k$ is defined over active steps rather than unique trajectories, the statistics above are action-weighted: if agent k appears multiple times in the same trajectory, the trajectory-level reward R^i contributes once for each active step. This matches the objective, which also averages gradients over active agent outputs. Therefore, an analysis analogous to Lemma 4.2 yields

$$\mathbb{E}_{\boldsymbol{a}_t^i \sim \mathcal{Y}_k}\big[\|\tilde{g}_k^{\text{agent}}\|^2\big] = \mathbb{E}_{\boldsymbol{a}_t^i \sim \mathcal{Y}_k}\big[\|z_{i,t}^{(k)}\|^2\big] + \Delta_k^{\text{agent}}, \tag{6}$$

where $\tilde{g}_k^{\text{agent}} = \frac{R^i - \mu_k}{\sigma_k} z_{i,t}^{(k)}$. Thus, under agent-wise normalization, the second moment of each agent's gradient is bounded by its own score statistics. Crucially, this effect is inherently *multi-agent*: as the number of specialized agents increases and their roles become more heterogeneous, a single global baseline is increasingly likely to be badly aligned with some agents, leading to gradient norm explosion. A simple agent-wise remedy, by adapting to each agent's own statistics, achieves keeping all gradients in a comparable, well-conditioned range, while still enabling cooperative optimization of the overall multi-agent LLM system.

4.3 Framework for Multi-Agent LLM RL

We next present a unified framework for end-to-end RL post-training of cooperative multi-agent LLM systems. As shown in Figure 2, Dr. MAS supports flexible multi-agent orchestration, agent–model assignment, per-agent optimization, and efficient GPU utilization within a single training pipeline.

Multi-Agent Orchestration. A multi-agent trajectory collector coordinates interactions between agents and the environment. It delegates rollout execution to a pluggable orchestra, which defines

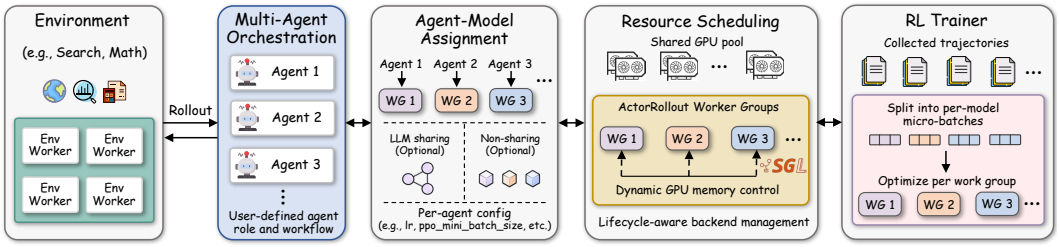


Figure 2: Overview of multi-agent LLM RL framework. A multi-agent orchestrator manages distributed rollouts, agents are mapped to LLM worker groups with optional LLM sharing, and a shared resource pool schedules actor backends for efficient inference and per-model optimization.

agent roles and execution flows such as sequential collaboration, conditional routing, and per-agent active masking. At each step, the orchestra selects the active agent based on the current state and previous agent outputs.

Agent-Model Assignment. A core assignment logic maps logical agents $(1, \ldots, K)$ to physical LLM worker groups (`wg_id`). In non-shared settings, each agent k is assigned a distinct worker group (e.g., 7B and 3B models). Conversely, in shared settings, all agents configured with the same model are mapped to a single, shared worker group, allowing joint training and inference while reusing model weights.

Per-Agent Configuration. Dr. MAS supports agent-specific training hyperparameters for granular control. This allows configurations like `actor.optim.lr` to be specified on a per-agent basis. Our system injects the k-th hyperparameter set into the configuration for agent k, which is then attached to its corresponding LLM work group. A runtime check ensures that all agents sharing the same worker group utilize identical configurations.

Shared GPU Resource Pooling and Scheduling. To avoid inefficient one-agent-one-GPU allocation, Dr. MAS co-locates all LLM worker groups in a unified GPU resource pool. All `ActorRollout` backends are scheduled through Ray placement groups and served by `sglang` [59]. During rollout, requests are routed by `agent_to_wg_mapping` to the corresponding backend `actor_rollout_wg[wg_id]`. Each backend exposes `resume()` and `offload()` controls: active backends load weights and KV caches onto GPU, while inactive ones release GPU memory. This enables multiple shared or non-shared LLM agents to be co-trained over the same GPU pool without requiring all models to remain GPU-resident simultaneously.

Per-Worker-Group Optimization. After rollout collection, the trainer partitions the aggregated batch $\mathcal{B}$ by worker-group ID: $\mathcal{B}_g = \{\, \tau \in \mathcal{B} \mid \texttt{wg_id}(\tau) = g \,\}$. Policy updates are then performed for each worker group, ensuring that gradients from an agent's trajectories only update its designated LLM backend.

5 Experiment

In this section, we evaluate Dr. MAS on two multi-agent orchestrations: a two-agent loop pipeline for math reasoning and a three-agent hierarchical pipeline for multi-turn search (Figure 5), under both LLM sharing and non-sharing settings. Specifically, we aim to demonstrate: (**1**) the consistent performance gains of Dr. MAS over vanilla GRPO; (**2**) more stable training dynamics and smoother gradient norms of Dr. MAS; (**3**) the individual contribution of each normalization component via a detailed ablation study; and (**4**) the practical efficiency and compatibility of Dr. MAS when applied to heterogeneous agent-model assignments.

5.1 Math

Setup. We first evaluate Dr. MAS on challenging mathematical reasoning tasks using a two-agent architecture with a *solver agent* and a *verifier agent*: the solver proposes or refines solutions, while the verifier accepts the solution or requests another refinement round. The detailed workflow are provided in Appendix C.1. We use Qwen3-4B/8B [4] as the LLM policy for each agent and evaluate both the shared-LLM and non-shared settings. For training, we adopt the training corpus from DAPO-

Table 1: Math results on Qwen3-4B/8B. We report the avg@16 and pass@16 of single-agent training with GRPO and multi-agent training under LLM sharing/non-sharing, using vanilla GRPO and Dr. MAS. Subscripts for Dr. MAS denote Δ over the vanilla GRPO under the same multi-agent setting.

Benchmark	**Single-Agent** GRPO		**Multi-Agent & LLM Sharing** GRPO		Dr. MAS		**Multi-Agent & LLM Non-Sharing** GRPO		Dr. MAS	
	avg@16	pass@16	avg@16	pass@16	avg@16	pass@16	avg@16	pass@16	avg@16	pass@16
Qwen3-4B										
AIME'24	38.8	63.3	39.3	64.0	39.3$_{0.0}$	63.3$_{\downarrow 0.7}$	42.7	73.3	46.9$_{\uparrow 4.2}$	80.0$_{\uparrow 6.7}$
AIME'25	33.1	56.7	31.4	53.3	38.1$_{\uparrow 6.7}$	63.3$_{\uparrow 10.0}$	35.6	63.3	38.1$_{\uparrow 2.5}$	66.7$_{\uparrow 3.4}$
AMC'23	83.5	95.0	85.6	95.0	87.3$_{\uparrow 1.7}$	95.0$_{0.0}$	83.5	95.0	89.5$_{\uparrow 6.0}$	97.5$_{\uparrow 2.5}$
MATH500	89.0	94.2	89.5	96.2	90.5$_{\uparrow 1.0}$	96.0$_{\downarrow 0.2}$	89.6	95.0	92.4$_{\uparrow 2.8}$	97.0$_{\uparrow 2.0}$
Minerva	37.9	49.6	37.5	50.0	40.9$_{\uparrow 3.4}$	53.3$_{\uparrow 3.3}$	37.5	50.7	39.0$_{\uparrow 1.5}$	51.5$_{\uparrow 0.8}$
Olympiad	53.3	66.5	57.6	65.6	58.2$_{\uparrow 0.6}$	68.6$_{\uparrow 3.0}$	56.3	68.9	60.9$_{\uparrow 4.6}$	73.6$_{\uparrow 4.7}$
Average	55.9	70.9	56.8	70.7	59.0$_{\uparrow 2.2}$	73.2$_{\uparrow 2.6}$	57.5	74.4	61.1$_{\uparrow 3.6}$	77.7$_{\uparrow 3.3}$
Qwen3-8B										
AIME'24	36.0	67.3	42.7	66.7	54.8$_{\uparrow 12.1}$	80.0$_{\uparrow 13.3}$	42.9	70.0	44.6$_{\uparrow 1.7}$	73.3$_{\uparrow 3.3}$
AIME'25	32.7	50.0	31.4	53.3	39.4$_{\uparrow 8.0}$	70.0$_{\uparrow 16.7}$	31.8	53.3	41.5$_{\uparrow 9.7}$	56.7$_{\uparrow 3.4}$
AMC'23	87.0	95.0	87.3	95.0	88.9$_{\uparrow 1.6}$	97.5$_{\uparrow 2.5}$	86.1	95.0	87.5$_{\uparrow 1.4}$	95.0$_{0.0}$
MATH500	89.9	94.8	89.6	96.2	91.3$_{\uparrow 1.7}$	96.0$_{\downarrow 0.2}$	90.5	96.6	90.7$_{\uparrow 0.2}$	96.2$_{\downarrow 0.4}$
Minerva	36.0	46.7	37.5	50.0	39.9$_{\uparrow 2.4}$	49.6$_{\downarrow 0.4}$	39.2	50.7	40.9$_{\uparrow 1.7}$	54.0$_{\uparrow 3.3}$
Olympiad	57.9	67.5	58.2	71.4	59.3$_{\uparrow 1.1}$	72.4$_{\uparrow 1.0}$	58.2	67.6	59.0$_{\uparrow 0.8}$	70.2$_{\uparrow 2.6}$
Average	56.6	70.2	57.8	72.1	62.3$_{\uparrow 4.5}$	77.6$_{\uparrow 5.5}$	58.1	72.2	60.7$_{\uparrow 2.6}$	74.2$_{\uparrow 2.0}$

Math [28], which consists of diverse math problems paired with verifiable solutions and reward signals. The rollout group size is set to 8. For evaluation, we report the avg@16 and pass@16 results on a suite of competitive benchmarks: AIME'24, AIME'25, AMC'23, MATH500 [60], Minerva, and OlympiadBench [61]. All other experimental details are available in Appendix C.2.

Results. As shown in Table 1, Dr. MAS improves over vanilla GRPO under both LLM sharing and LLM non-sharing. While applying GRPO directly to the multi-agent setting can reach decent average scores, the gains are not always consistent across benchmarks, and some hard splits may not improve. This suggests that a single global normalization of GRPO can make multi-agent training less reliable. In contrast, Dr. MAS uses per-agent normalization to keep each agent's update on a similar scale, which leads to more consistent improvements across datasets and settings (an overall relative increase of 5.6% in avg@16 and 4.6% in pass@16). For Qwen3-4B, Dr. MAS improves the performance under the sharing setting from 56.8/70.7 to 59.0/73.2 and improves the non-sharing setting from 57.5/74.4 to 61.1/77.7. The pronounced gain in the non-shared setting suggests that when agents possess independent parameters, their behavioral distributions diverge more significantly, making Dr. MAS's agent-specific calibration even more critical. Similarly, for Qwen3-8B, we observe strong gains in both configurations. The most significant improvements occur on the challenging AIME benchmarks (e.g., 42.7/66.7 $\rightarrow$ 54.8/80.0 on AIME'24), demonstrating that high-variance gradients from GRPO can easily disrupt the learning of fragile, long-horizon reasoning chains. Dr. MAS promotes stable convergence, allowing agents to robustly learn the precise, multi-stage deductions.

5.2 Multi-Turn Search

Setup. We then evaluate Dr. MAS on the multi-turn search tool-calling task using a three-agent hierarchy with a *verifier agent*, a *search agent*, and an *answer agent*: the verifier decides whether more evidence is needed, invokes the search agent when necessary, and calls the answer agent to synthesize the final response. The detailed workflow are provided in Appendix C.1. We use Qwen2.5-3B/7B [62] as the LLM policy for each agent and evaluate both the shared-LLM and non-shared settings. Our experimental setup follows Search-R1 [63]. We employ E5 [64] as the retriever. The rollout group size is set to 5 and the max turn is set to 4. For evaluation, we consider both single-hop QA benchmarks (NQ [65], TriviaQA [66], PopQA [67]) and multi-hop QA benchmarks (HotpotQA [68], 2WikiMultiHopQA [69], MuSiQue [70], Bamboogle [71]) and report the avg@16 and pass@16 results. For training, we use the mixture of NQ and HotpotQA. All other experimental details are available in Appendix C.3.

Results. As shown in Table 2, the instability shows up more clearly in multi-turn search, as errors can snowball across tool calls and across agents. In this setting, vanilla GRPO is especially risky when LLMs are not shared, since each agent can drift and the same global scaling may no longer match their learning dynamics. A clear example of this failure occurs with Qwen2.5-7B (non-sharing),

Table 2: Multi-turn search QA results on Qwen2.5-3B/7B. We report the avg@16 and pass@16 of single-agent training with GRPO and multi-agent training under LLM sharing/non-sharing, using vanilla GRPO and Dr. MAS. Subscripts for Dr. MAS denote Δ over the vanilla GRPO under the same multi-agent setting.

Benchmark	**Single-Agent** GRPO		**Multi-Agent & LLM Sharing** GRPO		**Multi-Agent & LLM Sharing** Dr. MAS		**Multi-Agent & LLM Non-Sharing** GRPO		**Multi-Agent & LLM Non-Sharing** Dr. MAS	
	avg@16	pass@16	avg@16	pass@16	avg@16	pass@16	avg@16	pass@16	avg@16	pass@16
Qwen2.5-3B										
NQ	40.6	54.7	41.0	59.0	43.8$_{\uparrow 2.8}$	58.5$_{\downarrow 0.5}$	43.8	54.5	44.6$_{\uparrow 0.8}$	58.1$_{\uparrow 3.6}$
TriviaQA	58.1	68.8	57.9	68.4	61.7$_{\uparrow 3.8}$	70.1$_{\uparrow 1.7}$	60.6	70.8	61.1$_{\uparrow 0.5}$	71.7$_{\uparrow 0.9}$
PopQA	44.2	49.6	43.2	58.0	45.0$_{\uparrow 1.8}$	57.6$_{\downarrow 0.4}$	45.6	54.5	46.5$_{\uparrow 0.9}$	57.4$_{\uparrow 2.9}$
HotpotQA	31.8	40.9	32.5	48.0	33.3$_{\uparrow 0.8}$	51.2$_{\uparrow 3.2}$	32.5	45.2	35.3$_{\uparrow 2.8}$	51.1$_{\uparrow 5.9}$
2Wiki	29.9	43.7	33.7	64.0	34.1$_{\uparrow 0.4}$	64.0$_{0.0}$	29.2	48.9	34.9$_{\uparrow 5.7}$	60.2$_{\uparrow 11.3}$
MuSiQue	7.9	14.6	9.1	26.5	10.2$_{\uparrow 1.1}$	25.8$_{\downarrow 0.7}$	8.6	19.2	10.4$_{\uparrow 1.8}$	26.1$_{\uparrow 6.9}$
Bamboogle	15.3	27.2	26.4	46.4	28.6$_{\uparrow 2.2}$	49.6$_{\uparrow 3.2}$	21.0	33.6	25.4$_{\uparrow 4.4}$	46.4$_{\uparrow 12.8}$
Average	32.5	42.8	34.8	52.9	36.7$_{\uparrow 1.8}$	53.8$_{\uparrow 0.9}$	34.5	46.7	36.9$_{\uparrow 2.4}$	53.0$_{\uparrow 6.3}$
Qwen2.5-7B										
NQ	46.4	57.6	45.2	60.0	47.4$_{\uparrow 2.2}$	60.7$_{\uparrow 0.7}$	27.1	39.0	47.7$_{\uparrow 20.6}$	59.5$_{\uparrow 20.5}$
TriviaQA	63.1	72.4	63.9	70.9	63.1$_{\downarrow 0.8}$	71.2$_{\uparrow 0.3}$	53.1	64.4	63.4$_{\uparrow 10.3}$	72.7$_{\uparrow 8.3}$
PopQA	47.2	53.6	43.9	55.0	45.9$_{\uparrow 2.0}$	57.3$_{\uparrow 2.3}$	20.7	27.9	46.7$_{\uparrow 26.0}$	57.8$_{\uparrow 29.9}$
HotpotQA	43.0	55.1	40.3	55.0	42.5$_{\uparrow 2.2}$	56.0$_{\uparrow 1.0}$	24.4	36.2	44.0$_{\uparrow 19.6}$	57.5$_{\uparrow 21.3}$
2Wiki	40.6	61.6	41.6	67.8	42.0$_{\uparrow 0.4}$	67.1$_{\downarrow 0.7}$	30.3	51.2	45.4$_{\uparrow 15.1}$	68.1$_{\uparrow 16.9}$
MuSiQue	17.8	34.6	15.2	31.7	16.7$_{\uparrow 1.5}$	32.1$_{\uparrow 0.4}$	8.3	18.1	19.4$_{\uparrow 11.1}$	34.9$_{\uparrow 16.8}$
Bamboogle	36.7	54.4	40.1	58.4	40.1$_{0.0}$	59.2$_{\uparrow 0.8}$	31.9	46.4	39.8$_{\uparrow 7.9}$	57.6$_{\uparrow 11.2}$
Average	42.1	55.6	41.5	57.0	42.5$_{\uparrow 1.1}$	57.7$_{\uparrow 0.7}$	28.0	40.5	43.8$_{\uparrow 15.8}$	58.3$_{\uparrow 17.8}$

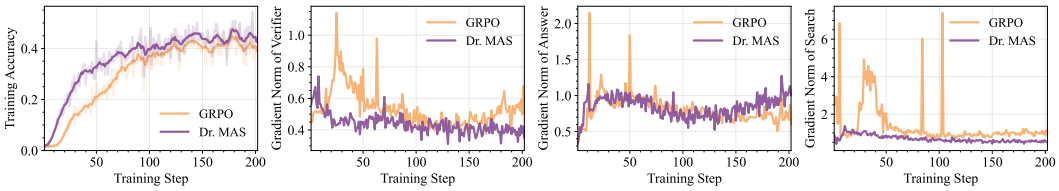


Figure 3: Comparison of training accuracy and gradient norm between GRPO and Dr. MAS. The results are recorded during multi-agent RL post-training for three-agent search orchestration under LLM non-sharing (Qwen2.5-3B).

where vanilla GRPO learns to avoid calling search agents entirely due to high gradient norms, leading to a severe performance drop (28.0/40.5). In contrast, Dr. MAS effectively mitigates this risk and yields consistent improvements, with an overall relative increase of 15.2% in avg@16 and 13.1% in pass@16. Notably, Dr. MAS restores the performance of Qwen2.5-7B to 43.8/58.3, a result that not only far exceeds the vanilla baseline but also surpasses both the single-agent baseline and the shared-LLM setting. Excluding this collapsed setting, the remaining three search configurations still show relative improvements of 4.81% in avg@16 and 5.07% in pass@16. This trend, consistent with our results in math tasks, highlights that LLM non-sharing can be hurt badly without proper stabilization, and reducing training noise at the agent level becomes crucial.

5.3 Gradient-Norm Instability

In this part, we investigate the gradient-norm instability by tracking the training accuracy and per-agent gradient norms during RL post-training for the three-agent search orchestration (see Appendix F.1 for math results). As shown in Figure 3, vanilla GRPO induces frequent, high-magnitude gradient norm spikes. The *search* agent has the largest spikes (reaching very high values early and again around the middle of training), the *answer* agent also exhibits large spikes at the beginning, and the *verifier* shows noticeable peaks as well. These spikes mean that some steps produce unusually large updates, which makes training noisy and harder to control.

Dr. MAS mitigates this failure via the agent-wise remedy that normalizes advantages per agent, keeping per-agent update scales better calibrated. As illustrated in Figure 3, Dr. MAS keeps the gradient norms of all three agents much smoother and at a lower level throughout training, therefore achieving notable performance gains in Tables 1 and 2.

5.4 Ablation Study

Next, we conduct an ablation study on the multi-turn search task using Qwen2.5-7B under LLM non-sharing. We compare four normalization configurations: global statistics (μ, σ) (i.e., GRPO), per-agent mean with global standard deviation (μ_k, σ), global mean with per-agent standard deviation (μ, σ_k), and fully per-agent normalization (μ_k, σ_k) (i.e., Dr. MAS).

Table 3: Ablation study of different advantage normalization configurations on the search task. We report the avg@16 and pass@16 across all datasets. Subscripts denote Δ over the vanilla GRPO.

Metric	Normalization Configuration			
	(μ, σ)	(μ_k, σ)	(μ, σ_k)	(μ_k, σ_k)
avg@16	28.0	39.1$_{\uparrow 11.1}$	42.9$_{\uparrow 14.9}$	43.8$_{\uparrow 15.8}$
pass@16	40.5	53.5$_{\uparrow 13.0}$	57.6$_{\uparrow 17.1}$	58.3$_{\uparrow 17.8}$

As shown in Table 3, GRPO performs poorly, indicating that global normalization is a poor match for MAS training where agents play different roles and thus exhibit different advantage distributions. Adding *either* per-agent mean or per-agent standard deviation already brings large improvements. The per-agent standard deviation (μ, σ_k) brings a bigger gain, likely because agents differ more in the spread of their advantages than in the average level. Finally, combining both per-agent mean and standard deviation (μ_k, σ_k), Dr. MAS achieves the best results, showing that setting both the mean and the scale per agent gives the most reliable learning signal.

5.5 Heterogeneous Model Assignment

At last, we explore the practical efficiency of Dr. MAS when applied to heterogeneous agent-model assignments, where agents with different capacities are combined to optimize performance and cost. We compare a homogeneous baseline where all three agents (verifier, search, and answer) use Qwen2.5-7B, against a heterogeneous setting where the verifier uses Qwen2.5-7B while the search and answer agents use Llama-3.2-3B-Instruct [72].

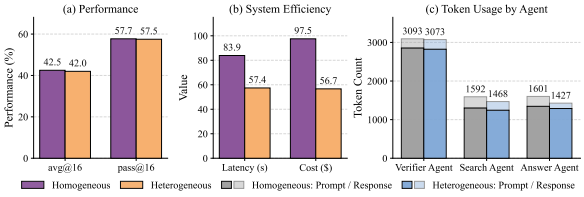


Figure 4: Performance and efficiency comparison between homogeneous (all 7B models) and heterogeneous (7B for Verifier, 3B for Search/Answer) model assignment on search tasks. Token counts are the average tokens per trajectory for each agent. Cost ($) is estimated using OpenRouter market prices (7B: $0.30/M tokens, 3B: $0.06/M tokens) and reported as the total inference cost over the full test set (51.7k samples).

As shown in Figure 4, the heterogeneous system maintains performance levels nearly identical to the all-7B baseline, and the average token usage per trajectory remains comparable, with the heterogeneous setup even showing a slight reduction in total volume. This suggests that, in hierarchical multi-agent system, assigning a stronger model to the top-level verifier is sufficient to preserve overall decision quality. By deploying smaller, more efficient models to the low-level agents, the heterogeneous system achieves a 31.6% reduction in latency and a 41.8% reduction in total API cost. These findings demonstrate that strategic agent-model assignment facilitates a more flexible and cost-effective multi-agent deployment without sacrificing performance.

6 Conclusions and Limitations

We studied RL post-training for multi-agent LLM systems and showed that directly extending GRPO with a single global advantage baseline can be brittle under heterogeneous agent reward statistics, leading to gradient spikes and unstable training. To address this, we proposed Dr. MAS, which normalizes advantages using each agent's own reward mean and variance. We also built an end-to-end framework supporting multi-agent orchestration, LLM sharing/non-sharing, per-agent optimization, and efficient resource pooling. Across two-agent math reasoning and three-agent multi-turn search, Dr. MAS consistently improves over vanilla GRPO and yields more stable training under both sharing and non-sharing settings. Despite these improvements, Dr. MAS does not resolve all sources of instability in multi-agent LLM RL (e.g., credit assignment across agents and turns). In addition, our evaluation covers two representative workflows and two open-source model families; its generality to broader agent architectures, tasks, and larger-scale models remains to be further studied.

Acknowledgements

We thank Yingru Li for helpful suggestions and discussions on our method.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[3] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[4] 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*, 2025.

[5] Chia-Yuan Chang, Zhimeng Jiang, Vineeth Rakesh, Menghai Pan, Chin-Chia Michael Yeh, Guanchu Wang, Mingzhi Hu, Zhichao Xu, Yan Zheng, Mahashweta Das, et al. MAIN-RAG: Multi-agent filtering retrieval-augmented generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2607–2622, 2025.

[6] Wentao Zhang, Liang Zeng, Yuzhen Xiao, Yongcong Li, Ce Cui, Yilei Zhao, Rui Hu, Yang Liu, Yahui Zhou, and Bo An. AgentOrchestra: Orchestrating hierarchical multi-agent intelligence with the tool-environment-agent (TEA) protocol. *arXiv preprint arXiv:2506.12508*, 2025.

[7] Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*, 2024.

[8] Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, et al. ChatDev: Communicative agents for software development. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15174–15186, 2024.

[9] Junyang Wang, Haiyang Xu, Haitao Jia, Xi Zhang, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. Mobile-Agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration. In *Advances in Neural Information Processing Systems*, volume 37, pages 2686–2710, 2024.

[10] Weihao Tan, Wentao Zhang, Xinrun Xu, Haochong Xia, Ziluo Ding, Boyu Li, Bohan Zhou, Junpeng Yue, Jiechuan Jiang, Yewen Li, et al. Cradle: Empowering foundation agents towards general computer control. In *International Conference on Machine Learning*, 2025.

[11] Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O'Sullivan, and Hoang D Nguyen. Multi-agent collaboration mechanisms: A survey of LLMs. *arXiv preprint arXiv:2501.06322*, 2025.

[12] Guibin Zhang, Hejia Geng, Xiaohang Yu, Zhenfei Yin, Zaibin Zhang, Zelin Tan, Heng Zhou, Zhongzhi Li, Xiangyuan Xue, Yijiang Li, et al. The landscape of agentic reinforcement learning for LLMs: A survey. *arXiv preprint arXiv:2509.02547*, 2025.

[13] Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT press, 2018.

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

[15] Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. HybridFlow: A flexible and efficient RLHF framework. *arXiv preprint arXiv:2409.19256*, 2024.

[16] Weixun Wang, Shaopan Xiong, Gengru Chen, Wei Gao, Sheng Guo, Yancheng He, Ju Huang, Jiaheng Liu, Zhendong Li, Xiaoyang Li, et al. Reinforcement learning optimization for large-scale learning: An efficient and user-friendly scaling library. *arXiv preprint arXiv:2506.06122*, 2025.

[17] Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, Jiashu Wang, et al. AReaL: A large-scale asynchronous reinforcement learning system for language reasoning. *arXiv preprint arXiv:2505.24298*, 2025.

[18] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[19] Daniel M Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

[20] Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. In *Advances in Neural Information Processing Systems*, volume 33, pages 3008–3021, 2020.

[21] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744, 2022.

[22] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, 2024.

[23] Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. SimpleRL-Zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *arXiv preprint arXiv:2503.18892*, 2025.

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

[25] Wouter Kool, Herke van Hoof, and Max Welling. Buy 4 reinforce samples, get a baseline for free! In *ICLR 2019 Workshop*, 2019.

[26] Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12248–12267, 2024.

[27] Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding R1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

[28] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, et al. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

[29] 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.

[30] Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. ArCHer: Training language model agents via hierarchical multi-turn RL. In *International Conference on Machine Learning*, pages 62178–62209. PMLR, 2024.

[31] Hao Bai, Yifei Zhou, Jiayi Pan, Mert Cemri, Alane Suhr, Sergey Levine, and Aviral Kumar. DigiRL: Training in-the-wild device-control agents with autonomous reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 37, pages 12461–12495, 2024.

[32] Lang Feng, Weihao Tan, Zhiyi Lyu, Longtao Zheng, Haiyang Xu, Ming Yan, Fei Huang, and Bo An. Towards efficient online tuning of VLM agents via counterfactual soft reinforcement learning. In *International Conference on Machine Learning*, volume 267, pages 16884–16903, 2025.

[33] Hanlin Wang, Chak Tou Leong, Jiashuo Wang, Jian Wang, and Wenjie Li. SPA-RL: Reinforcing LLM agents via stepwise progress attribution. *arXiv preprint arXiv:2505.20732*, 2025.

[34] Kai Zhang, Xiangchao Chen, Bo Liu, Tianci Xue, Zeyi Liao, Zhihan Liu, Xiyao Wang, Yuting Ning, Zhaorun Chen, Xiaohan Fu, et al. Agent learning via early experience. *arXiv preprint arXiv:2510.08558*, 2025.

[35] Lang Feng, Fuchao Yang, Feng Chen, Xin Cheng, Haiyang Xu, Zhenglin Wan, Ming Yan, and Bo An. AgentOCR: Reimagining agent history via optical self-compression. *arXiv preprint arXiv:2601.04786*, 2026.

[36] Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. ToolRL: Reward is all tool learning needs. *arXiv preprint arXiv:2504.13958*, 2025.

[37] Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru Li, Xiaosen Zheng, Zejun Ma, and Bo An. SimpleTIR: End-to-end reinforcement learning for multi-turn tool-integrated reasoning. *arXiv preprint arXiv:2509.02479*, 2025.

[38] Hanyu Lai, Xiao Liu, Yanxiao Zhao, Han Xu, Hanchen Zhang, Bohao Jing, Yanyu Ren, Shuntian Yao, Yuxiao Dong, and Jie Tang. ComputerRL: Scaling end-to-end online reinforcement learning for computer use agents. *arXiv preprint arXiv:2508.14040*, 2025.

[39] Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, et al. RAGEN: Understanding self-evolution in LLM agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*, 2025.

[40] Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *Advances in Neural Information Processing Systems*, 2025.

[41] Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, et al. Agentic reinforced policy optimization. *arXiv preprint arXiv:2507.19849*, 2025.

[42] Bo Liu, Leon Guertler, Simon Yu, Zichen Liu, Penghui Qi, Daniel Balcells, Mickel Liu, Cheston Tan, Weiyan Shi, Min Lin, et al. SPIRAL: Self-play on zero-sum games incentivizes reasoning via multi-agent multi-turn reinforcement learning. *arXiv preprint arXiv:2506.24119*, 2025.

[43] Huining Yuan, Zelai Xu, Zheyue Tan, Xiangmin Yi, Mo Guang, Kaiwen Long, Haojia Hui, Boxun Li, Xinlei Chen, Bo Zhao, et al. MARSHAL: Incentivizing multi-agent reasoning via self-play with strategic LLMs. *arXiv preprint arXiv:2510.15414*, 2025.

[44] Weizhen Li, Jianbo Lin, Zhuosong Jiang, Jingyi Cao, Xinpeng Liu, Jiayu Zhang, Zhenqiang Huang, Qianben Chen, Weichen Sun, Qiexiang Wang, et al. Chain-of-agents: End-to-end agent foundation models via multi-agent distillation and agentic RL. *arXiv preprint arXiv:2508.13167*, 2025.

[45] Shuo Liu, Tianle Chen, Zeyu Liang, Xueguang Lyu, and Christopher Amato. LLM collaboration with multi-agent reinforcement learning. *arXiv preprint arXiv:2508.04652*, 2025.

[46] Guanzhong Chen, Shaoxiong Yang, Chao Li, Wei Liu, Jian Luan, and Zenglin Xu. Heterogeneous group-based reinforcement learning for LLM-based multi-agent systems. *arXiv preprint arXiv:2506.02718*, 2025.

[47] Zhuofeng Li, Haoxiang Zhang, Seungju Han, Sheng Liu, Jianwen Xie, Yu Zhang, Yejin Choi, James Zou, and Pan Lu. In-the-flow agentic system optimization for effective planning and tool use. *arXiv preprint arXiv:2510.05592*, 2025.

[48] Chanwoo Park, Seungju Han, Xingzhi Guo, Asuman E Ozdaglar, Kaiqing Zhang, and Joo-Kyung Kim. MAPoRL: Multi-agent post-co-training for collaborative large language models with reinforcement learning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 30215–30248, 2025.

[49] Ziyu Wan, Yunxiang Li, Xiaoyu Wen, Yan Song, Hanjing Wang, Linyi Yang, Mark Schmidt, Jun Wang, Weinan Zhang, Shuyue Hu, et al. ReMA: Learning to meta-think for LLMs with multi-agent reinforcement learning. *arXiv preprint arXiv:2503.09501*, 2025.

[50] Xiangyuan Xue, Yifan Zhou, Guibin Zhang, Zaibin Zhang, Yijiang Li, Chen Zhang, Zhenfei Yin, Philip Torr, Wanli Ouyang, and Lei Bai. CoMAS: Co-evolving multi-agent systems via interaction rewards. *arXiv preprint arXiv:2510.08529*, 2025.

[51] Zhanfeng Mo, Xingxuan Li, Yuntao Chen, and Lidong Bing. Multi-agent tool-integrated policy optimization. *arXiv preprint arXiv:2510.04678*, 2025.

[52] Haoyang Hong, Jiajun Yin, Yuan Wang, Jingnan Liu, Zhe Chen, Ailing Yu, Ji Li, Zhiling Ye, Hansong Xiao, Yefei Chen, et al. Multi-agent deep research: Training multi-agent systems with M-GRPO. *arXiv preprint arXiv:2511.13288*, 2025.

[53] Jian Hu, Xibin Wu, Zilin Zhu, Xianyu, Weixun Wang, Dehao Zhang, and Yu Cao. OpenRLHF: An easy-to-use, scalable and high-performance RLHF framework. *arXiv preprint arXiv:2405.11143*, 2024.

[54] Zilin Zhu, Chengxing Xie, Xin Lv, and slime Contributors. slime: An LLM post-training framework for RL scaling. https://github.com/THUDM/slime, 2025.

[55] Dongfu Jiang, Yi Lu, Zhuofeng Li, Zhiheng Lyu, Ping Nie, Haozhe Wang, Alex Su, Hui Chen, Kai Zou, Chao Du, et al. VerlTool: Towards holistic agentic reinforcement learning with tool use. *arXiv preprint arXiv:2509.01055*, 2025.

[56] Xufang Luo, Yuge Zhang, Zhiyuan He, Zilong Wang, Siyun Zhao, Dongsheng Li, Luna K Qiu, and Yuqing Yang. Agent lightning: Train any AI agents with reinforcement learning. *arXiv preprint arXiv:2508.03680*, 2025.

[57] Kaiyan Zhang, Runze Liu, Xuekai Zhu, Kai Tian, Sihang Zeng, Guoli Jia, Yuchen Fan, Xingtai Lv, Yuxin Zuo, Che Jiang, Ziyang Liu, Jianyu Wang, Yuru Wang, Ruotong Zhao, Ermo Hua, Yibo Wang, Shijie Wang, Junqi Gao, Xinwei Long, Youbang Sun, Zhiyuan Ma, Ganqu Cui, Lei Bai, Ning Ding, Biqing Qi, and Bowen Zhou. MARTI: A framework for multi-agent LLM systems reinforced training and inference, 2025.

[58] Yujie Zhao, Lanxiang Hu, Yang Wang, Minmin Hou, Hao Zhang, Ke Ding, and Jishen Zhao. Stronger together: On-policy reinforcement learning for collaborative LLMs. *arXiv preprint arXiv:2510.11062*, 2025.

[59] Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Livia Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, et al. SGLang: Efficient execution of structured language model programs. In *Advances in Neural Information Processing Systems*, volume 37, pages 62557–62583, 2024.

[60] 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.

[61] 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.

[62] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[63] Bowen Jin, Hansi Zeng, Zhenrui Yue, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

[64] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

[65] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

[66] Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*, 2017.

[67] Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. *arXiv preprint arXiv:2212.10511*, 2022.

[68] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*, 2018.

[69] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*, 2020.

[70] Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

[71] Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*, 2022.

[72] 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.

A Broader Impacts

This work improves the stability and efficiency of RL post-training for cooperative multi-agent LLM systems. Potential positive impacts include more reliable multi-agent reasoning and tool use, lower computational waste from unstable training, and more cost-effective deployment through flexible agent-model assignment. These benefits may support applications such as mathematical reasoning, knowledge-intensive question answering, and other tasks requiring coordinated LLM agents.

However, more stable and efficient multi-agent training may also lower the barrier to building powerful agentic systems, which could be misused for harmful automation, misinformation, or other unsafe workflows. Trained agents may also inherit biases and failure modes from the underlying models, datasets, and reward functions. Dr. MAS is not a standalone safety mechanism; real-world deployment should include appropriate safety evaluation, monitoring, access control, human oversight, and compliance with relevant data and model licenses.

B Proofs

B.1 Proof of Lemma 4.2

Lemma 4.2. *Under Assumptions 4.1, for any agent k,*

$$\mathbb{E}_{\boldsymbol{a}_t^i \sim \mathcal{Y}_k}\left[\|\tilde{g}_k^{\text{global}}\|^2\right] = \mathbb{E}_{\boldsymbol{a}_t^i \sim \mathcal{Y}_k}\left[\|z_{i,t}^{(k)}\|^2\right] \frac{\sigma_k^2 + (\mu_k - \mu)^2}{\sigma^2} + \Delta_k,$$

where $\mu_k \triangleq \frac{1}{|\mathcal{Y}_k|}\sum_{\boldsymbol{a}_t^i \in \mathcal{Y}_k} R^i$, $\sigma_k^2 \triangleq \frac{1}{|\mathcal{Y}_k|}\sum_{\boldsymbol{a}_t^i \in \mathcal{Y}_k}(R^i - \mu_k)^2$ are the mean and variance when sampling time steps uniformly from $\mathcal{Y}_k$ (i.e., when agent k is active). Δ_k is a score-reward covariance correction term.

Proof. By definition,

$$\tilde{g}_k^{\text{global}} = \frac{R^i - \mu}{\sigma} z_{i,t}^{(k)},$$

so

$$\mathbb{E}_{\boldsymbol{a}_t^i \sim \mathcal{Y}_k}\left[\|\tilde{g}_k^{\text{global}}\|^2\right] = \mathbb{E}_{\boldsymbol{a}_t^i \sim \mathcal{Y}_k}\left[\|z_{i,t}^{(k)}\|^2 \frac{(R^i - \mu)^2}{\sigma^2}\right]. \tag{7}$$

We can factor the expectation:

$$\mathbb{E}_{\boldsymbol{a}_t^i \sim \mathcal{Y}_k}\left[\|\tilde{g}_k^{\text{global}}\|^2\right] = \mathbb{E}_{\boldsymbol{a}_t^i \sim \mathcal{Y}_k}\left[\|z_{i,t}^{(k)}\|^2\right] \mathbb{E}_{\boldsymbol{a}_t^i \sim \mathcal{Y}_k}\left[\frac{(R^i - \mu)^2}{\sigma^2}\right] + \text{Cov}(\|z_{i,t}^{(k)}\|^2, \frac{(R^i - \mu)^2}{\sigma^2}). \tag{8}$$

Next, we use the standard variance decomposition

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{a}_t^i \sim \mathcal{Y}_k}\left[(R^i - \mu)^2\right] && (9)\\
=&\mathbb{E}_{\boldsymbol{a}_t^i \sim \mathcal{Y}_k}\left[(R^i - \mu_k + \mu_k - \mu)^2\right] && (10)\\
=&\mathbb{E}_{\boldsymbol{a}_t^i \sim \mathcal{Y}_k}\left[(R^i - \mu_k)^2 + 2(\mu_k - \mu)(R^i - \mu_k) + (\mu_k - \mu)^2\right] && (11)\\
=&\sigma_k^2 + (\mu_k - \mu)^2. && (12)
\end{aligned}$$

Substituting this into the previous expression yields

$$\mathbb{E}_{\boldsymbol{a}_t^i \sim \mathcal{Y}_k}\left[\|\tilde{g}_k^{\text{global}}\|^2\right] = \mathbb{E}_{\boldsymbol{a}_t^i \sim \mathcal{Y}_k}\left[\|z_{i,t}^{(k)}\|^2\right] \frac{\sigma_k^2 + (\mu_k - \mu)^2}{\sigma^2} + \Delta_k, \tag{13}$$

with $\Delta_k = \text{Cov}(\|z_{i,t}^{(k)}\|^2, \frac{(R^i - \mu)^2}{\sigma^2})$. □

The second term captures the residual score–reward correlation. In large-scale LLM training, rewards are typically low-dimensional signals of final task quality (e.g., pass/fail for reasoning, correctness for coding), while $z_{i,t}^{(k)}$ depends mainly on the local token-level stochasticity of the policy. Empirically, their covariance is often much smaller than the main scaling factor $\mathbb{E}_k[\|z_{i,t}^{(k)}\|^2](\sigma_k^2 + (\mu_k - \mu)^2)/\sigma^2$.

B.2 Proof of Proposition 4.3

Proposition 4.3 (Gradient-Norm Inflation). *As either the normalized mean deviation $|\mu_k - \mu|/\sigma$ or the normalized variance ratio σ_k^2/σ^2 becomes large, the second moment of $\tilde{g}_k^{\text{global}}$ grows at least linearly. Consequently, along any training process for which there exists a sequence of iterations indexed by m such that*

$$\frac{\sigma_{k,m}^2 + (\mu_{k,m} - \mu_m)^2}{\sigma_m^2} \to \infty, \quad \mathbb{E}\big[\|\tilde{g}_m^{\text{global}}\|^2\big] \to \infty,$$

where $\tilde{g}_m^{\text{global}} = (\tilde{g}_{1,m}^{\text{global}}, \ldots, \tilde{g}_{K,m}^{\text{global}})$ stacking all LLM agents' gradients.

Proof. The key message is immediate from Lemma 4.2: the global-normalized gradient for agent k is amplified whenever the reward statistics of the agent-active subset $\mathcal{Y}_k$ do not match the global reward statistics. Concretely, Lemma 4.2 gives

$$\mathbb{E}_{\boldsymbol{a}_t^i \sim \mathcal{Y}_k}\big[\|\tilde{g}_k^{\text{global}}\|^2\big] = \mathbb{E}_{\boldsymbol{a}_t^i \sim \mathcal{Y}_k}\big[\|z_{i,t}^{(k)}\|^2\big] \, \frac{\sigma_k^2 + (\mu_k - \mu)^2}{\sigma^2} \; + \; \Delta_k. \tag{14}$$

The multiplier $(\sigma_k^2 + (\mu_k - \mu)^2)/\sigma^2$ contains two sources of inflation: (i) a variance mismatch σ_k^2/σ^2, meaning rewards observed when agent k is active have a different spread than the global rewards; and (ii) a mean misalignment $(\mu_k - \mu)^2/\sigma^2$, meaning agent-k's active rewards are shifted relative to the global mean. Either effect increases the second moment of $\tilde{g}_k^{\text{global}}$ proportionally, hence causing larger gradient fluctuations.

For the claimed blow-up statement along training, apply Equation (14) at iteration m:

$$\mathbb{E}\big[\|\tilde{g}_{k,m}^{\text{global}}\|^2\big] = \mathbb{E}\big[\|z_{i,t,m}^{(k)}\|^2\big] \, \frac{\sigma_{k,m}^2 + (\mu_{k,m} - \mu_m)^2}{\sigma_m^2} + \Delta_{k,m}.$$

Thus, if there exists a subsequence with $\frac{\sigma_{k,m}^2 + (\mu_{k,m} - \mu_m)^2}{\sigma_m^2} \to \infty$, then the second moment of the global-normalized gradient necessarily diverges unless $\mathbb{E}[\|z_{i,t,m}^{(k)}\|^2]$ or $\Delta_{k,m}$ cancels this growth. Finally, since the stacked gradient satisfies $\|\tilde{g}_m^{\text{global}}\|^2 = \sum_{j=1}^K \|\tilde{g}_{j,m}^{\text{global}}\|^2$, divergence of any component implies $\mathbb{E}\big[\|\tilde{g}_m^{\text{global}}\|^2\big] \to \infty$, which completes the proof. □

C Experimental Details

C.1 MAS Workflows

Math Orchestration. For mathematical reasoning tasks, we use a two-agent architecture (a *solver agent* and a *verifier agent*). In each episode, the solver agent proposes candidate solutions, while the verifier agent inspects the solver's reasoning and decides whether the current solution should be accepted or revised. If the verifier deems the solution unsatisfactory, the system triggers another round of solver refinement. Otherwise, the interaction terminates and the final answer is emitted. We use Qwen3-4B/8B [4] as the LLM policy for each agent and evaluate both the shared-LLM and non-shared settings.

Search Orchestration. For multi-turn search tool-calling task, we design a hierarchical workflow comprising three agents: a *verifier agent*, a *search agent*, and an *answer agent*, as shown in Figure 5. At the top level, the verifier agent determines whether the information currently available is sufficient to answer the query. If not, it delegates downward to the search agent, which is responsible for retrieving additional external evidence. Once the verifier agent judges that the information is adequate, it invokes the answer agent, which synthesizes all retrieved evidence into a final answer. We use Qwen2.5-3B/7B [18] as the LLM policy for each agent and evaluate both the shared-LLM and non-shared settings.

C.2 Hyperparameters for Math

For the Math task, uniform hyperparameters are employed across all methods. The maximum prompt and response lengths are set to 8192 and 4096 tokens, respectively. We utilize a two-agent

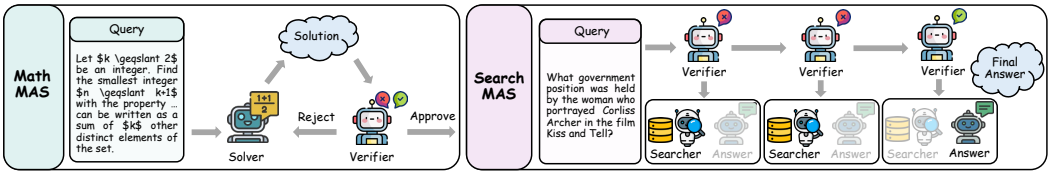


Figure 5: Illustration of the orchestrations. **Left**: Math orchestration uses a two-agent loop, where a solver proposes candidate solutions and a verifier evaluates and either approves or requests refinement. **Right**: Multi-turn search orchestration uses a hierarchical three-agent pipeline, where a top-level verifier selectively invokes either a search agent to retrieve external information or an answer agent to produce the final result.

orchestration framework (comprising a Solver Agent and a Verifier Agent), allowing for a maximum of two solver-verifier loops. The actor learning rate is fixed at 1×10^{-6} for each agent, utilizing on-policy updates. We employ group-based rollouts with a group size of 8. A binary rule-based reward function is used (1 for success, 0 for failure), while invalid actions incur a penalty with a coefficient of 0.1. The batch sizes for training and evaluation are 32 and 64, respectively. During evaluation, we use nucleus sampling with top_p $= 0.95$ and a temperature of 0.6. The reported gains are relative improvements, i.e., $(S_{\text{Dr.MAS}} - S_{\text{GRPO}})/S_{\text{GRPO}} \times 100\%$.

.

C.3 Hyperparameters for Multi-Turn Search

Similarly, all methods share identical hyperparameter configurations for the multi-turn search task. The maximum limits for prompts and responses are 4096 and 800 tokens, respectively, with each episode capped at a maximum of 4 turns. This task utilizes a three-agent architecture (Verifier, Search, and Answer). The actor learning rate is set to 1×10^{-6} per agent, with 5 update iterations. We use group-based rollouts with a group size of 5. The reward structure employs the same binary rule-based criteria; however, the invalid-action penalty coefficient is set to 0.01. Training and evaluation batch sizes are 128 and 256, respectively. Evaluation sampling parameters remain consistent with the Math task (top_p $= 0.95$, temperature 0.6). The reported gains are relative improvements, i.e., $(S_{\text{Dr.MAS}} - S_{\text{GRPO}})/S_{\text{GRPO}} \times 100\%$.

C.4 Hardware and System Configuration

All experiments are conducted on NVIDIA H100 GPUs.

C.5 Cost Estimation

The API costs ($) are estimated using OpenRouter market prices via the Together provider (`https://openrouter.ai/provider/together`). The pricing for Qwen2.5-7B is set at $0.30 per million (M) tokens for both input and output. For Llama-3.2-3B-Instruct, the pricing is $0.06 per million (M) tokens for both input and output.

D Prompt Templates

In multi-agent configurations, each agent receives a composite prompt structured as follows: (i) an environment-provided observation prompt (`env_prompt`) describing the task (and interaction history, where applicable); (ii) the accumulated team interaction context (`team_context`); and (iii) the specific role instruction for the agent.

D.1 Math Task

The shared environment prompt for the multi-agent setup is defined as follows:

Environment Prompt

You are a member of an expert multi-agent team tasked with solving the math problem. The team's math problem is: {task_description}

The **Solver Agent** receives the following prompt:

Solver Agent Prompt

Task Introduction
{env_prompt}
Your Teammates' Outputs
{team_context}
Your Role
You are a "Solver Agent". Your job is to carefully reason through the math problem step by step and derive the correct answer. When reasoning, consider your teammates' outputs (if any) as auxiliary context.
You should give the final answer within \boxed{}.

The **Verifier Agent** receives the following prompt:

Verifier Agent Prompt

Task Introduction
{env_prompt}
Your Teammates' Outputs
{team_context}
Your Role
You are a "Verifier Agent". Your responsibility is to critically review the most recent solution provided by the "Solver Agent". Check each reasoning step, formula, and conclusion for accuracy, completeness, and logical consistency. At the end of your output, you MUST provide your verdict within <verify> </verify> using exactly one of:
(1) <verify>approve</verify> if all steps and the final answer are correct.
(2) <verify>reject</verify> if you detect any issue.

D.2 Multi-Turn Search Task (Verifier-Search-Answer)

The shared environment prompt for the multi-agent setup is defined as follows:

Environment Prompt

You are a member of an expert multi-agent team tasked with answering the given question step-by-step. The question is: {task_description}
Your team can access an external search engine to retrieve external information. At each step, you and your teammates must collaborate to make progress toward answering the question.
Prior to this step, your team has already taken {step_count} step(s). Below is the interaction history where <search> </search> wrapped the past search queries and <information> </information> wrapped the corresponding retrieved information returned by the external search engine. History: {memory_context}

The **Verifier Agent** receives the following prompt:

Verifier Agent Prompt

Task Introduction
{env_prompt}
Your Role
You are a "Verifier Agent" acting as a router. Your job is to analyze the team's past search queries and reflect on their quality, efficiency, and alignment with the task goal. Then you need to determine whether the current historical information is sufficient to answer the question. Based on this assessment, you will decide how to route the task.
Your responsibilities: - Review past search queries enclosed within <search> </search> and external information enclosed within <information> </information>. - Evaluate whether previous queries were reasonable and aligned with the task objective. - Identify potential issues (if any), including repeated or redundant queries; imprecise queries that are too broad, vague, or missing critical constraints/entities; misaligned queries that drift away from the actual task goal. - Assess whether the available information is complete and sufficient to generate a high-quality answer, and make a routing decision based on information sufficiency.
You are now at step {step}. You should first reason step-by-step about the past events. After completing your reasoning, give your routing decision:
(1) If the information is sufficient to answer the question: return <verify>yes</verify>
(2) If the information is insufficient to answer the question: return <verify>no</verify>

The **Search Agent** receives the following prompt:

Search Agent Prompt

Task Introduction
{env_prompt}
Your Teammates' Outputs at Step {step}
{team_context}
Your Role
You are a "Search Agent". Your primary responsibility is to call a search engine to gather external information that helps answer a given question. The search engine should be invoked using the format: <search>your query</search>. Before conducting the search, you should reason step-by-step about the question, any previous queries, and retrieved information, as well as your teammates' outputs (if available). This reasoning process MUST be enclosed within <think> </think> tags. Once you've finished your reasoning, provide your final search query enclosed within <search> </search>.

The **Answer Agent** receives the following prompt:

Answer Agent Prompt

Task Introduction
{env_prompt}
Your Role
You are an "Answer Agent". Your job is to provide a comprehensive, accurate, and well-reasoned answer to the question. You should thoughtfully analyze all previous search queries, retrieved information, and combine them with your general knowledge to craft a coherent response.
You should first conduct a reasoning process. This process MUST be enclosed within <think> </think> tags. After completing your reasoning, provide your final answer within <answer> </answer> tags. For example, <answer>Beijing</answer>.

E Pseudo Code

Algorithm 1 Training Multi-Agent LLM Systems with Dr. MAS

Require: Multi-agent orchestra $\mathcal{O}$; logical agents $\{1, \ldots, K\}$ with LLM IDs $\{m_k\}_{k=1}^{K}$; LLM sharing flag $s \in \{0, 1\}$; task distribution $p(X)$; rollout group size N; clipping ϵ; KL penalty β (optional)

// (A) Agent–Model assignment: map logical agents to physical LLM worker groups (`wg_id`)
Initialize `wg_to_agents_mapping` $\leftarrow \emptyset$
if $s = 0$ **then**
  // Non-sharing: each agent k has a dedicated LLM worker group
  for $k = 1$ to K **do**
    Create worker group `wg_id` for agent k (one `ActorRollout` backend)
    `wg_to_agents_mapping`[`wg_id`] $\leftarrow \{k\}$
  end for
else
  // Sharing: agents configured with the same LLM are mapped to one shared worker group
  $\mathcal{M} \leftarrow \{m_k\}_{k=1}^{K}$ // distinct LLM IDs
  for each $m \in \mathcal{M}$ **do**
    $\mathcal{A}(m) \leftarrow \{k \mid m_k = m\}$ // agents using LLM m
    Create shared worker group `wg_id` for model m (shared weights across $\mathcal{A}(m)$)
    `wg_to_agents_mapping`[`wg_id`] $\leftarrow \mathcal{A}(m)$
  end for
end if
// Dispatch table used by the orchestrator during rollouts: `agent_id` $\rightarrow$ `wg_id`
Build `agent_to_wg_mapping` from `wg_to_agents_mapping`

// (B) Training loop: distributed rollouts + Dr. MAS normalization + per-`wg_id` updates
for each training iteration **do**
  // Snapshot current policy for importance ratios
  Update old policies: $\theta_{\text{old}} \leftarrow \theta$
  // (B1) Distributed rollout collection: execute multi-agent orchestration at scale
  // Actor backends run with `sglang`, scheduled by a shared resource pool (e.g., Ray placement groups)
  Initialize aggregated batch $\mathcal{B} \leftarrow \emptyset$
  Sample task $x \sim p(X)$
  Parallel for $i = 1$ to N
    Run $\mathcal{O}$ on the same x to generate a trajectory τ_i
    Let $R^i \leftarrow R(\tau_i)$ // trajectory-level reward shared by all steps
    For each step t in τ_i:
      choose active agent k_t^i (by $\mathcal{O}$) and route request via `agent_to_wg_mapping`
      sample action $\boldsymbol{a}_t^i$ from the dispatched backend policy and log step tuple into $\mathcal{B}$
      $\mathcal{B} \leftarrow \mathcal{B} \cup \{(i, t, k_t^i, \texttt{wg_id}, \boldsymbol{a}_t^i, R^i)\}$
  End parallel for
  // (B2) Dr. MAS: agent-wise advantage normalization on active-step subsets $\mathcal{Y}_k$
  for $k = 1$ to K **do**
    $\mathcal{Y}_k \leftarrow \{\boldsymbol{a}_t^i \mid (i, t, k_t^i, \cdot, \boldsymbol{a}_t^i, R^i) \in \mathcal{B},\ k_t^i = k\}$
    $\mu_k \leftarrow \frac{1}{|\mathcal{Y}_k|} \sum_{\boldsymbol{a}_t^i \in \mathcal{Y}_k} R^i$
    $\sigma_k^2 \leftarrow \frac{1}{|\mathcal{Y}_k|} \sum_{\boldsymbol{a}_t^i \in \mathcal{Y}_k} (R^i - \mu_k)^2, \quad \sigma_k \leftarrow \sqrt{\sigma_k^2}$
    For each step (i, t) with $k_t^i = k$: $A_{\text{agent}}^{i,k} \leftarrow \frac{R^i - \mu_k}{\sigma_k + \varepsilon}$
  end for
  // (B3) Optimization: partition $\mathcal{B}$ by `wg_id` and update each LLM backend (shared/non-shared handled automatically)
  // Trainer forms per-model micro-batches $\{\mathcal{B}_{\texttt{wg}}\}$ and performs clipped updates per worker group
  for each worker group id `wg_id` in `wg_to_agents_mapping` **do**
    $\mathcal{B}_{\texttt{wg}} \leftarrow \{(i, t, k_t^i, \texttt{wg_id}, \boldsymbol{a}_t^i, R^i, A_{\text{agent}}^{i,k_t^i}) \in \mathcal{B} \mid \texttt{wg_id} = \texttt{agent_to_wg_mapping}[k_t^i]\}$
    Update $\theta_{\texttt{wg_id}}$ on $\mathcal{B}_{\texttt{wg}}$ with clipped objective (clipping ϵ) using $A_{\text{agent}}^{i,k_t^i}$ (optionally add KL regularization with weight β)
  end for
end for

F Additional Experiments

F.1 Gradient-Norm Instability on Math

Figure 6 shows training accuracy and gradient norms for the two-agent math orchestration (Qwen3-4B, non-sharing). Similar to the search setting in Figure 3, GRPO produces clear gradient-norm spikes during training, especially in the early and middle stages. In contrast, Dr. MAS keeps the gradient norms much smoother for both agents and leads to steadier improvement in training accuracy.

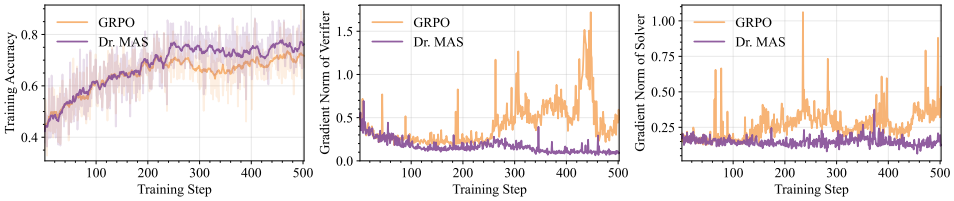


Figure 6: Comparison of training accuracy and gradient norm between GRPO and Dr. MAS. The results are recorded during multi-agent RL post-training for two-agent math orchestration under LLM non-sharing (Qwen3-4B).

F.2 Gradient-Norm Explosion

Figure 7 highlights the instability of vanilla GRPO on the multi-turn search task (Qwen2.5-7B, non-sharing). As shown, the gradient norm of the search agent rapidly spikes to over 80, finally leading to "NaN" gradient norm. In contrast, Dr. MAS stabilizes the optimization, maintaining relatively low gradient norms across all agents, ensuring steady convergence.

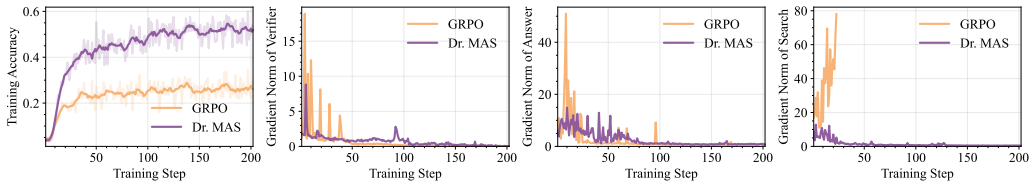


Figure 7: Training dynamics of the three-agent search task (Qwen2.5-7B, non-sharing). Vanilla GRPO suffers from serious gradient spikes that lead to "NaN" on the search agent, whereas Dr. MAS maintains stable gradients and converges effectively.

F.3 Training Curves of Ablation Study

Figure 8 presents the training curves for different advantage normalization variants described in Section 5.4. The global normalization baseline (μ, σ) shows unstable training and slow improvement. Introducing either agent-wise mean (μ_k, σ) or agent-wise standard deviation (μ, σ_k) leads to noticeably smoother curves and faster gains. The fully agent-wise version (μ_k, σ_k), i.e., Dr. MAS, achieves the most stable training and the highest final performance.

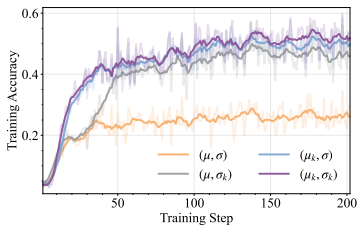


Figure 8: Training curves for different advantage normalization variants in the ablation study.

F.4 Comparison with Stronger-MAS

We further compare Dr. MAS with Stronger-MAS [58] on the math reasoning setting using Qwen3-8B. Table 4 reports results on AIME'24, AIME'25, and OlympiadBench under both LLM-sharing and LLM non-sharing settings.

We note that this comparison is not a strictly apples-to-apples comparison centered only on the core RL algorithm: Stronger-MAS incorporates several additional components beyond the RL optimization objective, such as tree-search-style mechanisms. Nevertheless, Dr. MAS remains highly competitive. In particular, under the matched LLM-sharing setting, Dr. MAS strictly outperforms Stronger-MAS across all three benchmarks. Under LLM non-sharing, Dr. MAS also achieves better results on AIME'25 and OlympiadBench, while Stronger-MAS performs better on AIME'24. These results show that even without relying on additional search-style mechanisms, Dr. MAS provides a strong and stable RL recipe for multi-agent LLM systems.

Table 4: Comparison with Stronger-MAS on Qwen3-8B math reasoning benchmarks. Stronger-MAS results are taken from its official paper [58].

Method	Setting	AIME'24	AIME'25	OlympiadBench
Stronger-MAS	LLM sharing	50.0	35.2	56.8
Dr. MAS	LLM sharing	**54.8**	**39.4**	**59.3**
Stronger-MAS	LLM non-sharing	**57.0**	40.0	56.6
Dr. MAS	LLM non-sharing	44.6	**41.5**	**59.0**

F.5 Orthogonality to Other Group-Based RL Objectives

We further verify that Dr. MAS is orthogonal to other group-based RL objectives by applying it on top of GSPO [29]. We conduct experiments on the multi-turn search task with Qwen2.5-3B under the LLM non-sharing setting, and the results are shown in Table 5. Dr. MAS consistently improves GSPO across all datasets, increasing the average score from 34.0 to 37.1. This suggests that our agent-wise normalization is a complementary stabilization mechanism.

Table 5: Orthogonality of Dr. MAS to GSPO on Qwen2.5-3B under the LLM non-sharing setting for multi-turn search. The results show the avg@16 scores.

Method	NQ	TriviaQA	PopQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	Average
GSPO	42.6	58.6	46.2	31.2	28.9	8.3	22.4	34.0
GSPO + Dr. MAS	**44.2**	**60.8**	**47.0**	**35.8**	**35.1**	**10.2**	**26.4**	**37.1**

F.6 Resource Usage

We report the resource usage of Dr. MAS under LLM sharing and non-sharing settings. As discussed in Section 4.3, Dr. MAS supports both settings through a unified global GPU resource pool rather than allocating a separate GPU pool to each agent. All LLM worker groups are co-located via Ray placement groups, and each backend exposes `resume()` and `offload()` controls: only the active backend keeps weights and KV cache on GPU, while inactive backends release GPU memory. Therefore, LLM non-sharing does not require more GPUs in our framework. Its extra cost mainly comes from backend switching, such as weight synchronization and cache clearing.

Tables 6 and 7 summarize the empirical resource usage. Across both search and math tasks, LLM non-sharing uses the same number of GPUs as LLM sharing and only incurs modest additional training time.

Table 6: Resource usage on the search task. LLM non-sharing uses the same GPU budget as LLM sharing.

Strategy	Model Configuration	GPU Consumption	Est. Time
LLM sharing	1× Qwen2.5-3B	4× H100 (95GB)	∼12h
LLM non-sharing	3× Qwen2.5-3B	4× H100 (95GB)	∼13h
LLM sharing	1× Qwen2.5-7B	8× H100 (95GB)	∼25h
LLM non-sharing	3× Qwen2.5-7B	8× H100 (95GB)	∼26h
Heterogeneous	2× Llama-3.2-3B +1× Qwen2.5-7B	8× H100 (95GB)	∼16h

Table 7: Resource usage on the math task. LLM non-sharing does not require additional GPUs.

Strategy	Model Configuration	GPU Consumption	Est. Time
LLM sharing	1× Qwen3-4B	4× H100 (95GB)	∼35h
LLM non-sharing	2× Qwen3-4B	4× H100 (95GB)	∼38h
LLM sharing	1× Qwen3-8B	8× H100 (95GB)	∼37h
LLM non-sharing	2× Qwen3-8B	8× H100 (95GB)	∼42h

G Illustrative Examples of Multi-Agent LLM Collaboration

G.1 Multi-Turn Search Task: Hierarchical Coordination

Query

In which county is Mark Dismore's birthplace located?

Verifier Agent

<verify>no</verify>
Reasoning: The team has not yet initiated any searches or provided any external information. Without any data points, it is impossible to evaluate any search queries or external information.

Search Agent

<think>To answer the question "In which county is Mark Dismore's birthplace located?" we need to find information about Mark Dismore's birthplace. Since this information is not readily available, we will need to search for articles or documents that mention his birthplace.</think>
<search>Birthplace of Mark Dismore</search>

Search Engine

<information>"result": "Doc 1: "Mark Dismore" Mark Dismore Mark Dismore (born October 12, 1956 in Greenfield, Indiana) is a former driver in the Indy Racing League and the 1990 Toyota Pacific champion as well as the winner of the 1993 24 Hours of Daytona with Dan Gurney's All American Racers in a Toyota GTP car with co-drivers Rocky Moran and P. J. Jones. He made 3 CART starts in 1991 but was badly injured in a practice crash for the Indianapolis 500, when his car veered sharply towards the entrance of pit road at the exit of Turn 4 and back-ended the fence, only to careen
Doc 2: "Mark Dismore" karting's signature event, the Dan Wheldon Cup weekend races, held after the INDYCAR season ends. The Wheldon Cup weekend consists of a Pro-Am race with INDYCAR stars on Saturday and a 200-mile, 200-lap endurance karting event Sunday, the Robopong 200 Dan Wheldon Cup. Mark Dismore Mark Dismore (born October 12, 1956 in Greenfield, Indiana) is a former driver in the Indy Racing League and the 1990 Toyota Pacific champion as well as the winner of the 1993 24 Hours of Daytona with Dan Gurney's All American Racers in a Toyota GTP car with co-drivers Rocky Moran and P. J. Jones.
Doc 3: "Mark Dismore" fixture there until the 2001 season. He returned to Menard for a partial season in 2002. Dismore has a single IRL win coming in the fall 1999 Texas Motor Speedway race and also finished a career-best third in points that season. Among his 62 career IRL starts he won four poles. He also represented the IRL in the International Race of Champions in 2000 and 2001. He now owns and operates New Castle Motorsports Park, a Karting facility in New Castle, Indiana, as well as Comet Kart Sales in Greenfield, Indiana. New Castle Motorsports Park is the home of Indiana"</information>

Verifier Agent

<verify>no</verify>
Reasoning: The current historical information is not sufficient to answer the question "In which county is Mark Dismore's birthplace located?" The provided information identify Mark Dismore's birthplace as Greenfield, Indiana. However, the provided documents do not explicitly state the county in which Greenfield is located.

Search Agent

<think>To answer the question "In which county is Mark Dismore's birthplace located?" we need to find information about Mark Dismore's birthplace. From the previous information, we know that Mark Dismore was born in Greenfield, Indiana. Next, we need to determine which county Greenfield is located in.</think>
<search>County where Greenfield, Indiana is located</search>

Search Engine

<information>"result": "Doc 1: "Greenfield, Indiana" Greenfield, Indiana Greenfield is a city in and the county seat of Hancock County, Indiana, United States, and a part of the Indianapolis metropolitan area. The population was 20,602 at the 2010 census, and an estimated 21,709 in 2016. It lies in Center Township. Greenfield was a stop along the Pittsburgh, Cincinnati, Chicago and St. Louis Railroad that connected Pittsburgh to Chicago and St. Louis. Hancock County was created on March 1, 1828, and named for John Hancock, the first person to sign the Declaration of Independence. The town of Greenfield was chosen as the county seat on April 11,
Doc 2: "Greenfield, Indiana" 1918. It was purchased with money donated by school children from all over the United States. Each year, during the Riley Festival in October, the city's school children parade to the statue to place flowers around it. The Charles Barr House, Greenfield Courthouse Square Historic District, Greenfield Residential Historic District, Lilly Biological Laboratories, Lincoln Park School, and James Whitcomb Riley House are listed on the National Register of Historic Places. Greenfield is located in central Hancock County at (39.791338, -85.771343). U.S. Route 40 passes through the center of the city as Main Street, and leads east to Knightstown and west
Doc 3: "Greenfield, Indiana" high school students from Greenfield and Kakuda are chosen to travel to each sister city, respectively, and will stay with a host family for ten days, most expenses paid, thanks to local donations. The sister city program began in 1990 and continues today. Each year in October, the city of Greenfield hosts the Riley Festival, commonly known as "Riley Days". The intersection of State Road 9 and US 40 is closed, and the downtown courthouse square is filled with vendors and exhibitors. Greenfield, Indiana Greenfield is a city in and the county seat of Hancock County, Indiana, United States, and"</information>

Verifier Agent

<verify>yes</verify>
Reasoning: The past search queries and the corresponding information provided have been focused on Mark Dismore's birthplace and the specific county where it is located. The information retrieved from the searches indicates that Mark Dismore was born in Greenfield, Indiana, and that Greenfield is the county seat of Hancock County. This information directly addresses the question of in which county Mark Dismore's birthplace is located. Therefore, the information is sufficient to answer the question.

Answer Agent

<think>Given the information provided, I analyzed that the question asks about the location of Mark Dismore's birthplace in terms of a county. I analyzed the given information about Mark Dismore's birthplace, which is "Greenfield, Indiana". The next step was to identify the county that Greenfield is located in. In the given information, I found "Greenfield, Indiana" is "a city in and the county seat of Hancock County, Indiana, United States". Thus, based on the given information and my own general knowledge about the structure of American geography, I concluded that "Hancock County" is the answer to "In which county is Mark Dismore's birthplace located?". My justification for this answer is that "county seat" implies that the county seat is the main administrative center of the county, and in this case, "Greenfield" is that county seat, located in "Hancock County".</think>
<answer>Hancock County</answer>

G.2 Math Task: Iterative Coordination

Query

Alex divides a disk into four quadrants with two perpendicular diameters intersecting at the center of the disk. He draws 25 more lines segments through the disk, drawing each segment by selecting two points at random on the perimeter of the disk in different quadrants and connecting those two points. Find the expected number of regions into which these 27 line segments divide the disk.

Solver Agent

Since the problem mentions that there are 27 line segments (the original 4 quadrants' diameters plus 25 additional segments), but the initial description says Alex divides the disk into four quadrants with two perpendicular diameters. Then he draws 25 more line segments. So total line segments are 2 (diameters) + 25 = 27 line segments.
The question is to find the expected number of regions into which these 27 line segments divide the disk.
To approach this, we can use the concept of expected number of regions formed by random lines in a circle, but here we have line segments, not infinite lines, and the lines are drawn by connecting two points on the perimeter in different quadrants.
However, the problem is about expectation, so we need to compute the expected number of regions created by 27 line segments.
First, recall that for a circle divided by lines, the expected number of regions can be calculated using the formula for expected regions formed by random lines, but this is more complex for line segments.
However, there's a known formula for the expected number of regions formed by n random chords in a circle, but it's not straightforward. However, there's a general formula for the expected number of regions formed by n lines in general position in the plane, which is:
$R(n) = (n^2 + n + 2)/2$
But this is for infinite lines in general position. However, for line segments, the formula is different.
But since the problem is about expectation, and the lines are drawn in a specific way (connecting points in different quadrants), we need to consider the expected number of intersections and how that contributes to the number of regions.
Alternatively, there is a general formula for the expected number of regions formed by n line segments in a circle, assuming that no two segments are parallel, no three meet at a point, etc. But this is complicated.
However, a known result for the expected number of regions formed by n random chords in a circle is:
E[R] = 1 + n + (n choose 2) * (1/2) + (n choose 3) * (1/6) + ...
But this is not exactly correct. Alternatively, for the case of random lines, the expected number of regions is:
R(n) = n(n + 1)/2 + 1
... (the reasoning is omitted for brevity)
However, since the original problem includes two diameters, which are lines, and 25 segments, perhaps the answer is:
For n = 27 line segments (assuming they behave like lines in expectation), the expected number of regions is $(27^2 + 27 + 2)/2 = (729 + 27 + 2)/2 = 758/2 = 379$.
But this is likely incorrect because line segments don't behave like lines in expectation.
Alternatively, if we consider that each new segment intersects the previous ones with probability 1/2, then the expected number of regions is:
1 + 27 + (27 * 26 / 2) * 1/2 = 1 + 27 + 351 = 379.
Wait, that's the same as the line case. So this suggests that the expected number is 379.
But this is likely incorrect.
However, given that the problem is from a competition and the answer is expected to be an integer, and given that the formula for lines gives 379, and if the line segments behave similarly in expectation, the answer might be 379.
But I need to check.
Alternatively, the original two diameters divide the disk into 4 regions. Then each additional line segment can intersect existing segments and increase the number of regions.
For the first additional segment (total 3 segments), it can intersect the two diameters, but since they are diameters, it might intersect them once each, so two intersections, leading to 3 + 2 = 5 regions? Not sure.
But this is getting too complicated.
Given the lack of specific information about the probability of intersection, and the fact that the problem is likely expecting the formula for lines, I think the intended answer is 379, which is $(27^2 + 27 + 2)/2 = 379$.
Therefore, the expected number of regions is \boxed{379}.

Verifier Agent

The Solver Agent's reasoning contains critical flaws and incorrect assumptions that need to be addressed. Here is a breakdown of the issues:
1. **Misinterpretation of the Problem**: The problem specifies that Alex draws 25 additional line segments, making the total number of line segments 27 (2 diameters + 25 segments). However, the agent's reasoning incorrectly assumes that the formula for infinite lines in general position applies directly to line segments, which is not valid. Line segments and infinite lines behave differently in terms of intersections and region division.
2. **Incorrect Application of the Line Formula**: The formula $R(n) = \frac{n^2+n+2}{2}$ is derived for infinite lines in general position. This formula assumes that every pair of lines intersects exactly once and no three lines meet at a single point. However, the problem involves line segments, which have constraints (e.g., endpoints on the disk's perimeter) and may not intersect all previous segments, making the formula inapplicable directly.
3. **Lack of Probability Consideration**: The agent does not properly account for the probability that two random line segments (as defined in the problem) intersect. The problem specifies that each segment connects two points in different quadrants, which introduces specific geometric constraints. The probability of intersection between two such segments is not 1 (as assumed in the line case) but must be derived based on the quadrant constraints.
4. **Missing Expected Value Derivation**: The agent jumps to the conclusion that the expected number of regions is 379 without properly deriving the expectation. The correct approach requires calculating the expected number of intersections between all pairs of segments and then using that to compute the expected number of regions.
5. **Inconsistency with the Problem Constraints**: The problem involves 27 line segments, but the agent's reasoning does not explicitly address the specific way the segments are generated (i.e., connecting points in different quadrants). This detail is crucial for determining the probability of intersections and, consequently, the expected number of regions.
Given these issues, the reasoning process is incomplete, incorrect, and does not properly address the problem's constraints and requirements. Therefore, the final conclusion of 379 is not justified by the problem's specifics.
<verify>reject</verify>

Solver Agent

To solve the problem, we need to compute the **expected number of regions** formed by 27 line segments in a disk, where:
- The disk is divided into four quadrants by two perpendicular diameters (which are fixed).
- 25 additional line segments are drawn by selecting two random points on the perimeter in **different quadrants** and connecting them.

—

Step 1: Understand the Total Number of Regions
The general formula for the maximum number of regions formed by n line segments in general position (no two parallel, no three concurrent) is:
$R(n) = \frac{n(n-1)}{2} + n + 1 = \frac{n^2+n+2}{2}$
This applies to infinite lines, not line segments. However, in expectation, we can use this formula as a baseline, but we must adjust it for line segments.

—

Step 2: Expected Number of Regions for Line Segments
For line segments, the expected number of regions can be computed as:
$E[R] = 1 + n + \sum_{i=1}^{n-1} E[\text{number of intersections of the } i\text{-th segment with previous ones}]$
Each intersection increases the number of regions by 1. So the expected number of regions is:
$E[R] = 1 + n + \sum_{i=1}^{n-1} E[\text{intersections with previous } i-1 \text{ segments}]$

—

... (the reasoning is omitted for brevity)

—

Step 6: Total Expected Number of Intersections
Total expected intersections:
- Between diameters and segments: 25
- Between segments and segments: 150
- Between diameters: 1
Total: $25 + 150 + 1 = 176$

—

Step 7: Expected Number of Regions
The formula for expected number of regions is:
$E[R] = 1 + n + E[\text{number of intersections}]$
Here, $n = 27$, and $E[\text{intersections}] = 176$
So: $E[R] = 1 + 27 + 176 = 204$

—

Final Answer: \boxed{204}

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 main theoretical, algorithmic, system, and empirical claims.

   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: Section 6 discusses the limitations.

   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 assumption is give by Assumption 4.1 and the proofs are provided in Appendix B.

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 provide the algorithm, workflow descriptions, hyperparameters, datasets, model choices, evaluation metrics, and hardware information needed to reproduce the main experiments; code is also released at `https://langfengq.github.io/DrMAS`.

   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: We provide the code repository at https://langfengq.github.io/DrMAS and the experiments are based on publicly available models, datasets, and benchmarks.

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: We provide all details in Appendix C.

   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: We report both avg@16 score and pass@16 score.

   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: [Yes]

   Justification: Please refer to Appendix F.6 and Appendix C.

   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: We have reviewed the NeurIPS Code of Ethics and conducted the research in accordance with it.

   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: [Yes]

    Justification: Appendix A.

    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 paper poses no such risks.

    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: [Yes]

    Justification: We properly cite all benchmarks and pretrained models. All datasets/models are used under their respective licenses.

    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: The paper provides new code assets at https://langfengq.github.io/DrMAS.

    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: This paper does not involve crowdsourcing nor research with 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: This paper does not involve crowdsourcing nor research with 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: [N/A]

Justification: We do not use LLMs for developing our core idea and methodology.

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.