# ADVEVO-MARL: SHAPING INTERNALIZED SAFETY THROUGH ADVERSARIAL CO-EVOLUTION IN MULTI-AGENT REINFORCEMENT LEARNING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

LLM-based multi-agent systems excel at planning, tool use, and role coordination, but their openness and interaction complexity also expose them to jailbreak, prompt-injection, and adversarial collaboration. Existing defenses fall into two lines: (i) self-verification that asks each agent to pre-filter unsafe instructions before execution, and (ii) external guard modules that police behaviors. The former often underperforms because a standalone agent lacks sufficient capacity to detect cross-agent unsafe chains and delegation-induced risks; the latter increases system overhead and creates a single-point-of-failure—once compromised, system-wide safety collapses, and adding more guards worsens cost and complexity. To solve these challenges, we propose AdvEvo-MARL, a co-evolutionary multi-agent reinforcement learning framework that internalizes safety into task agents. Rather than relying on external guards, AdvEvo-MARL jointly optimizes attackers (which synthesize evolving jailbreak prompts) and defenders (task agents trained to both accomplish their duties and resist attacks) in adversarial learning environments. To stabilize learning and foster cooperation, we introduce a public baseline for advantage estimation: agents within the same functional group share a group-level mean-return baseline, enabling lower-variance updates and stronger intra-group coordination. Across representative attack scenarios, AdvEvo-MARL consistently keeps attack-success rate (ASR) below 20%, whereas baselines reach up to 38.33%, while preserving—and sometimes improving—task accuracy (up to +3.67% on reasoning tasks). These results show that safety and utility can be jointly improved without relying on extra guard agents or added system overhead.

## 1 INTRODUCTION

LLM-based agents exhibit advanced capabilities in software engineering (Pan et al., 2025), computer use (Ning et al., 2025), and scientific discovery (Shao et al., 2025). Building on this progress, multi-agent systems (MAS) coordinate specialized agents with diverse expertise to harness collective intelligence for solving increasingly complex real-world problems. However, as MAS become more capable, they also face growing safety challenges (Raza et al., 2025). On one hand, MAS inherit vulnerabilities from single agents, particularly their susceptibility to jailbreak attacks, where malicious actors attempt to bypass safety guardrails. On the other hand, the complex interaction dynamics among agents, along with the presence of potentially unauthorized or adversarial agents, significantly expand the attack surface beyond that of isolated systems (He et al., 2025).

To mitigate these risks, researchers mainly explore two broad categories of defense: (i) empowering each agent to locally verify the benignness of its inputs before generating responses (tse Huang et al., 2025), and (ii) deploying external inspector agents to monitor and regulate information flow throughout interactions (Xiang et al., 2025). While these approaches are effective to some extent, they suffer from notable limitations. External guard agents introduce a single point of failure—once compromised, the system is left defenseless—and scaling up the number of guards quickly incurs prohibitive computational costs, rendering them impractical for large-scale deployments (Chennabasappa et al., 2025). Meanwhile, individual agents have limited capacity to detect or resist sophisticated, cross-agent attacks, making self-verification in isolation insufficient (Zhu et al., 2025). A natural intuition is to embed safety awareness within task agents through targeted safety training. Yet

training agents individually overlooks the collaborative dynamics required for effective multi-agent defense, and conventional safety training based on static datasets often leads to overfitting and poor generalization against adaptive adversaries (Geissler et al., 2024).

To address these challenges, we introduce AdvEvo-MARL, a co-evolutionary multi-agent reinforcement learning (MARL) framework that embeds safety awareness directly within task agents. The core idea is to jointly evolve attackers, which generate increasingly sophisticated jailbreak prompts, and defenders, which must both resist these attacks and fulfill their assigned tasks. AdvEvo-MARL initializes training with a curated pool of adversarial prompts derived from representative attack strategies. Since attackers lack prior knowledge of effective jailbreak tactics, we first warm them up using carefully designed seed prompts from this pool by supervised fine-tuning (SFT). During following MARL, attackers rewrite and refine these prompts to create more potent adversarial inputs, while defenders are simultaneously optimized to withstand these evolving threats and maintain task performance. To further stabilize training and foster coordination, we introduce a **public baseline** for advantage estimation: agents within the same functional group (e.g., attackers or defenders) share the group's mean return as their baseline. This mechanism enables agents to learn from peer behaviors, reduces variance in policy updates, and strengthens intra-group cooperation. With training, attackers evolve beyond static attack templates, while defenders acquire more robust and generalizable safety behaviors. This co-evolutionary process drives continuous safety enhancement, mitigating the risk of overfitting to fixed attack distributions and enabling resilience against adaptive adversaries.

Experiments on three representative MAS attack scenarios—agent manipulation, message corruption, and user instruction hijacking—demonstrate the effectiveness of AdvEvo-MARL in enhancing system robustness. Further task benchmarks show minimal performance degradation, and in some cases even improved task capabilities, underscoring the potential of AdvEvo-MARL as a standardized framework for building MAS that are both safe and capable.

In summary, our main contributions are three-folds:

- We propose AdvEvo-MARL, a novel multi-agent reinforcement learning framework that internalizes safety awareness within each agent through adversarial co-evolution. In this evolving paradigm, attackers and defenders iteratively compete and improve, leading to increasingly robust strategies on both sides.
- We introduce a public baseline mechanism for advantage estimation, where agents within the same functional group (e.g., attackers or defenders) use the group's mean return as a baseline. This design promotes collaborative learning among agents and enables more stable policy updates during training.
- Experiments across multiple representative MAS attack settings demonstrate consistent safety gains—achieving up to a maximum of 18.33% improvement. Further evaluations on standard task benchmarks reveal minimal degradation and, in several cases even enhanced task performance, underscoring AdvEvo-MARL's effectiveness in simultaneously promoting multi-agent system safety and task utility.

## 2 Related Work

Our work builds on two main research lines. The first examines safety in MAS, where adversarial threats such as agent manipulation and message corruption motivates defenses like self-verification, guard agents, and peer inspection. The second explores multi-agent reinforcement learning (MARL), which has enabled coordinated training and has recently been applied to LLM-based systems. These perspectives motivate our proposed AdvEvo-MARL, which unifies safety and MARL by co-evolving attackers and defenders to embed intrinsic safety awareness into agents.

### 2.1 Safety in Multi-Agent Systems.

LLMs are known to exhibit safety vulnerabilities, especially when exposed to adversarial attacks. Equipping agents with external tools or memory systems further expands the attack surface (Raza et al., 2025; Chen et al., 2024). While multi-agent systems (MAS) built upon such agents demonstrate strong task-solving capabilities, they are also vulnerable to a wide range of threats, most commonly: (1) manipulating agents to induce malicious behaviors (Yu et al., 2024), and (2) corrupting communication

messages or workflow execution (He et al., 2025; Zhang et al., 2024). To mitigate these risks, several defense strategies have been proposed. Some works leverage *self-verification*, encouraging each agent to assess the benignness of its inputs before responding (Fan & Li, 2025; tse Huang et al., 2025), while others employ a dedicated *guard agent* to monitor and rectify message flows (tse Huang et al., 2025). Another line of research collects safety-oriented interaction trajectories and trains graph neural networks to detect and correct unsafe responses (Wang et al., 2025). Furthermore, decentralized defenses have also been explored, where agents inspect one another to form peer-based protection (Fan & Li, 2025). Although these approaches provide partial safeguards, they face key limitations. Individual agents often lack the capacity to detect sophisticated attacks, while centralized guard agents introduce a single point of failure and impose computational overhead in complex systems. In contrast, we advocate embedding safety awareness directly into each agent through reinforcement learning, enabling intrinsic defense capabilities and fundamentally improving the robustness of MAS.

### 2.2 Multi-Agent Reinforcement Learning.

Reinforcement learning (RL) has proven effective in post-training LLMs (Shao et al., 2024; Team et al., 2025), with methods such as Proximal Policy Optimization (PPO) and Group Relative Policy Optimization (GRPO) yielding substantial performance gains (Shao et al., 2024). More recently, RL has also been applied to enhance agentic behaviors in language-based systems (Jin et al., 2025). Multi-agent reinforcement learning (MARL), exemplified by algorithms like MAPPO and QMIX (Kang et al., 2023; Rashid et al., 2020), extends RL to coordinated multi-agent settings (Liu et al., 2025). Several recent studies adapt MARL to LLM-based systems: one line of work applies MARL to improve collaborative agent behaviors in structured game environments (Park et al., 2025); another develops hierarchical MAS with high-level planners and low-level executors using parameter sharing to enhance meta-reasoning (Wan et al., 2025); yet another treats each Retrieval-Augmented Generation (RAG) module as an agent, applying MARL to jointly optimize task performance (Chen et al., 2025). **However**, most methods train a single backbone model with **shared parameters across agents**, limiting true agent-level diversity. In contrast, our framework trains **multiple distinct backbone models** collaboratively under RL, enabling genuine co-evolution. Building on these advances, our work explores MARL as a vehicle to improve MAS safety. By co-evolving attackers and defenders in an adversarial learning environment, we embed safety awareness directly into task agents through continuous interaction and adaptation, fostering robust and generalizable defense capabilities.

## 3 Preliminary

We formulate the interaction among learning agents as a partially observable Markov game:

$$\mathcal{G} = (\mathcal{S}, \{\mathcal{A}_i\}_{i=1}^{N}, P, \{\mathcal{O}_i\}_{i=1}^{N}, \gamma, \mathcal{T}), \tag{1}$$

where $\mathcal{S}$ denotes the state space, $\mathcal{A}_i$ represents the action space of agent $i$, $P$ is the state transition function, $\mathcal{O}_i$ is the observation function for agent $i$, $\gamma$ is the discount factor, and $\mathcal{T}$ is the finite time horizon. Each agent $i \in \{1, \ldots, N\}$ follows a stochastic policy $\pi_i(a_i \mid o_i)$, conditioned on local observation $o_i \sim \mathcal{O}_i(s)$, and jointly contributes to the environment evolution via the composite action $a = (a_1, \ldots, a_N)$. In the context of LLMs-based agents, instead of treating each token as an action, we define the action of an agent as generating a complete response that consists of a token sequence.

The agents are partitioned into two disjoint sets: attackers $\mathcal{A}$ and defenders $\mathcal{D}$, $\mathcal{A} \cap \mathcal{D} = \emptyset$ and $\mathcal{A} \cup \mathcal{D} = \{1, \ldots, N\}$. The attackers attempt to compromise system's safety guardrail, while the defenders must resist adversarial attacks and preserve task performance. All agents interact over the course of $T$ steps. At the end of each episode, the system produces a final output $y = \Phi(\tau)$, where $\tau = (s_0, a_0, s_1, \ldots, s_T)$ denotes the complete trajectory induced by the multi-agent interaction. This output is then evaluated by the environment or a trusted judge to form a global reward $G(\tau)$, upon which each agent receives its own local reward $r_i$. The learning goal is to co-evolve attackers and defenders under shared dynamics and finally induce a stable and robust equilibrium between attacker and defender populations. This is captured by the following game-theoretic objective, where $\{\pi_k\}_{k=1}^{N}$ denotes the joint policy of all N agents:

$$\max_{\{\pi_j\}_{j\in\mathcal{D}}} \min_{\{\pi_i\}_{i\in\mathcal{A}}} \mathbb{E}_{\tau\sim\{\pi_k\}_{k=1}^{N}} \left[ \sum_{j\in\mathcal{D}} r_j(\tau) - \sum_{i\in\mathcal{A}} r_i(\tau) \right]. \tag{2}$$

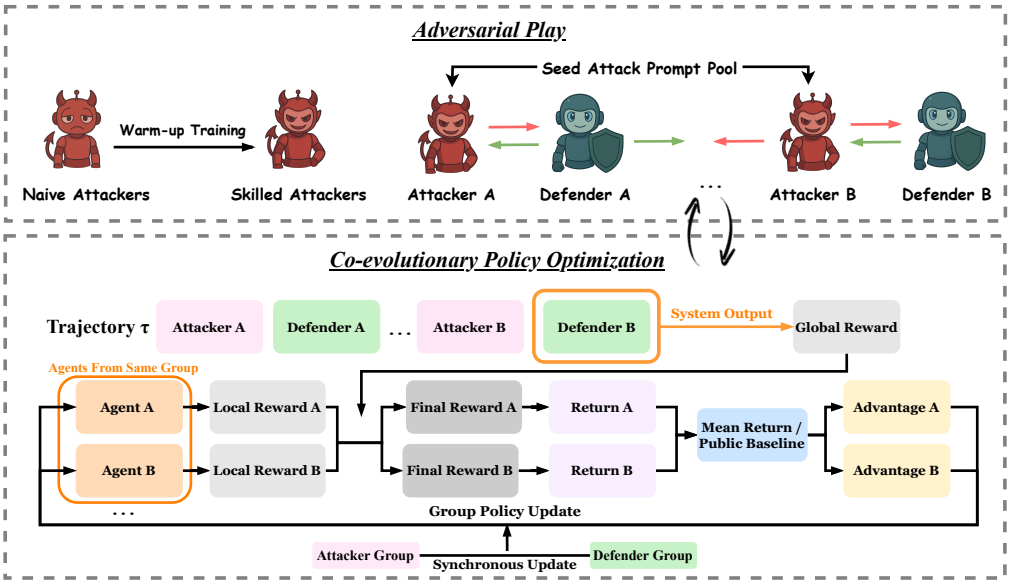



Figure 1: **Framework**. AdvEvo-MARL begins by warming up attacker agents through supervised fine-tuning to embed prior knowledge of jailbreak behaviors. Then, attackers and defenders learn to co-evolve via adversarial multi-agent reinforcement learning. During policy updates, agents within the same functional group (i.e., attackers or defenders) leverage a public baseline which is computed as the mean return of their respective group to estimate their individual advantages for optimization.

# 4 METHODOLOGY

In this section, we introduce AdvEvo-MARL, a multi-agent reinforcement learning framework designed to improve the safety of multi-agent systems. We first provide an overview of AdvEvo-MARL, then detail the attacker warm-up procedure, and finally present the adversarial RL pipeline with public-baseline-based advantage estimation.

## 4.1 OVERVIEW

As shown in fig. 1, AdvEvo-MARL unfolds in two stages. First, an *attacker warm-up* phase uses supervised fine-tuning to inject prior knowledge of jailbreak strategies, preventing trivial or ineffective attacks at the start of training. Upon this initialization, we introduce an *adversarial co-evolutionary RL stage* where attackers and defenders are jointly optimized through repeated interactions, enabling defenders to acquire robust and adaptive safety behaviors against evolving threats. To stabilize learning and encourage group-consistent updates, agents within the same role leverage a *public baseline* for advantage estimation, reducing variance and promoting effective collaboration.

## 4.2 BOOTSTRAPPING ADVERSARIAL GENERATION VIA ATTACKER WARM-UP

As attackers lack a prior understanding of jailbreak behaviors and adversarial prompting techniques, we first conduct warm-up training before MARL. We construct dataset $D_{adv}$ consisting of paired samples of the form $(x_{behavior}, x_{attack})$, where $x_{behavior}$ is the trivial harmful questions, and $x_{attack}$ is the re-written attack prompts using certain jailbreak techniques. Specifically, we begin by sampling 1,000 harmful behaviors from existing public datasets, ensuring broad coverage across diverse categories of harmful content. We then apply representative jailbreak strategies to generate corresponding adversarial attack prompts, obtaining an initial jailbreak prompt dataset $D_{init}$. Given the original questions and their associated attack variants, we employ an advanced reasoning model to synthesize multi-step reasoning traces that illustrate how to construct effective adversarial prompts. To ensure quality, we filter out invalid reasoning trajectories that are contradictory, off-topic, or vague using a LLM-as judge method. The resulting dataset $D_{adv}$ contains approximately 4,000 high-quality training samples. AdvEvo-MARL leverages imitation learning to equip attackers with jailbreak knowledge from the curated $D_{adv}$, thereby accelerating exploration in the early stages of training.

### 4.3 AdvEvo-MARL: Safe and Capable Multi-Agent Systems via co-evo RL

To build a safe and capable MAS, we embeds safety awareness directly into agents through adversarial co-evolution, enabling them to withstand evolving attacks while maintaining strong task performance. Importantly, we **trains multiple backbone models** collaboratively under RL, **rather than** relying on a **single shared-parameter model**, ensuring genuine co-evolution across diverse agents.

**Training Algorithm** Following the attacker warm-up stage, both attackers and defenders are jointly optimized within a co-evolutionary multi-agent reinforcement learning process. All agents are trained using REINFORCE++ to improve both system safety and task performance (Hu, 2025). To facilitate collaborative learning and stabilize policy updates, we introduce a public baseline for advantage estimation.

Specifically, during each rollout episode, the advantage for each agent is computed relative to the mean return of all agents within the same role group (i.e., attackers or defenders), rather than being estimated solely from its own return trajectory. Formally, for episode $\tau$ we define:

$$b^A(\tau) = \frac{1}{|\mathcal{A}|} \sum_{i \in \mathcal{A}} r_i^A(\tau), \qquad b^D(\tau) = \frac{1}{|\mathcal{D}|} \sum_{j \in \mathcal{D}} r_j^D(\tau), \tag{3}$$

where $b^A$ and $b^D$ denote the mean return value of attackers and defenders respectively. The resulting advantage estimate for any agent $k \in \{1, \ldots, N\}$ is then given by

$$\hat{A}_k(\tau) = r_k(\tau) - \begin{cases} b^A(\tau), & \text{if } k \in \mathcal{A}, \\ b^D(\tau), & \text{if } k \in \mathcal{D}. \end{cases} \tag{4}$$

Finally, the training loss for agent $k$ is defined as:

$$\begin{aligned} \mathcal{L}_{\text{REINFORCE++}}(\theta_k) = &- \mathbb{E}_t \left[ \min \left( r_{t,k}(\theta_k) \hat{A}_{t,k}, \text{clip}\left(r_{t,k}(\theta_k), 1-\varepsilon, 1+\varepsilon\right) \hat{A}_{t,k} \right) \right] \\ &+ \beta_{\text{KL}} \mathbb{E}_t \left[ \text{KL}\left(\pi_{\theta_k}(\cdot \mid x_{t,k}) \| \pi_{\text{ref},k}(\cdot \mid x_{t,k})\right) \right], \end{aligned} \tag{5}$$

where $r_{t,k}(\theta_k) = \frac{\pi_{\theta_k}(a_{t,k}|x_{t,k})}{\pi_{\theta_k^{\text{old}}}(a_{t,k}|x_{t,k})}$ denotes the importance sampling ratio, clipping $clip$ restricts updates magnitude, and the KL term measures divergency between learned policy $\pi_{\theta_k}$ and reference policy $\pi_{ref,k}$ to regulate training.

**Reward Modeling** To care distinct objectives of attackers and defenders, we design separate reward mechanisms for each agent type. Attackers receive rewards based on whether the final system output achieves the intended malicious goal, as evaluated by a global reward signal. In contrast, defenders are responsible for both resisting jailbreak attempts and fulfilling their assigned tasks. Relying solely on the global reward, however, can introduce misaligned incentives for defenders: an individual agent may receive misleading feedback due to the behavior of others. For instance, when some agents generate unsafe responses but the aggregated system output remains benign.

To address this issue, we assign rewards based on both individual agent's response and the final system output. Therefore, the rewards of defenders are evaluated at both the local response level and the global system level, as a combination of task performance and safety compliance. All agents also receive a formatting reward that enforces their outputs to put reasoning process between `<think>` and `</think>` and enclose final response with `<response>` and `</response>` tags. The overall reward is formulated as:

$$R_k = \begin{cases} \gamma_f \cdot f \; - \; \alpha_s \cdot s, & \text{if } k \in \mathcal{A}, \\ \alpha_s \cdot s \; + \; \beta_t \cdot t \; + \; \gamma_f \cdot f, & \text{if } k \in \mathcal{D}, \end{cases} \tag{6}$$

where $s$, $t$, and $f$ represent the rewards for safety, task utility, and format compliance respectively. For both safety and task performance, a reward of $1$ is assigned if the output is safe or correct, and $-1$ otherwise. For formatting, a reward of $0.5$ is given if the response satisfies the pre-defined structure, and $-0.1$ otherwise. In practice, we prioritize safety in the first half of training ($\alpha_s = 1$, $\beta_t = 0.5$), and reverse the weights afterward to emphasize task performance.

# 5 EXPERIMENTS

Experiments cross 3 representative multi-agent attack scenarios and 3 task-specific benchmarks to assess its effectiveness in enhancing both safety and task utility. We first describe the experimental setup. Then we report results on red team attacks to demonstrate the robustness of our approach against adversarial threats. Next, we present task evaluations to assess the model's general task performance. Finally, we conduct ablation studies to validate the design choices of AdvEvo-MARL.

## 5.1 EXPERIMENTAL SETUP

**Multi-agent systems.** To ensure a comprehensive evaluation under varying communication structures, we consider three representative system topologies in our experiments. (1) Chain mode: agents interact sequentially. Each agent can only observe the message from its immediate predecessor. (2) Tree mode: a hierarchical structure where two child agents exchange messages and a parent agent summarizes the communication history to produce a final output. (3) Complete mode: a fully connected topology where each agent can send and receive messages to and from all other agents. All experiments are conducted with three agents. Unless otherwise specified, we use QWen2.5 instruction-tuned models (3B and 7B) as the backbone foundation models.

**Attack methods.** We choose three widely adopted attack strategies mainly focusing on jailbreak attacks and harmful information propagation within MAS. (1) *NetSafe* (Yu et al., 2024), alters agent behavior by injecting 'dark traits' into profile configurations. One agent is randomly selected as malicious attacker in each episode. (2) *AutoInject* (tse Huang et al., 2025), randomly injects adversarial prompts into communication messages between agents. (3) UserHijack, manipulate user instructions to insert targeted adversarial content, simulating compromised user input.

**Baselines.** We compare AdvEvo-MARL against several baseline methods. (1) Vanilla instruction-tuned QWen2.5 series 3B and 7B models without safety training as backbone models. (2) Challenger, a self-verification strategy where individual agent verify the benignness of its input before generating a response. All agents are equipped with this self-checking mechanism in our setting. (3) Inspector, introduces external guard agents to detect and correct malicious messages. We only deploy one inspector agent to monitor all message flows during interaction.

**Datasets.** In training, we sample 4,000 problems from levels 3–5 of MATH-500 dataset (Lightman et al., 2024) to serve as regular task prompts for defenders. In adversarial training, we use the described seed pool for attack rewriting. For **system safety evaluation**, we follow the original NetSafe protocol and adopt its official evaluation dataset. Meanwhile, we construct a 300 adversarial prompts pool by sampling JailbreakBench, Wild Jailbreak, and Strong Reject. These prompts are used in both AutoInject and UserHijack settings. For **general task evaluation**, we select 3 prevailing benchmarks: (1) *mathematical reasoning*: AIME'24 & AIME'25 (AIME, 2025), challenging high-school mathematics requiring deep thinking and creative problem-solving, each containing 30 questions in total; (2) *coding*: LiveCodeBench (v6, 2025.01 - 2025.05) (Jain et al., 2024), collecting coding problems from live online platforms, providing a realistic, dynamic, challenging environment for coding capability evaluation; (3) *general reasoning*: GPQA-diamond (Rein et al., 2024), 100 graduate-level Q&A problems encompassing physics, chemistry, biology and other scientific domains.

**Metrics.** We employ three metrics to comprehensively evaluate both the robustness and utility of multi-agent systems. (1) Attack success rate (ASR): the proportion of evaluation samples where the system ultimately produces a harmful response. (2) Contagion rate (PR): the ratio of agents that exhibit unsafe behaviors at any point during the interaction episode, reflecting the system's process-level safety. (3) Task performance: we adopt accuracy (Acc) for mathematical and general reasoning tasks, and Pass@1 for coding tasks.

## 5.2 MAIN RESULTS

Table 1 presents a comprehensive comparison of ASR and CR across a range of models, system topologies, and adversarial settings. Among all open-source baselines, AdvEvo-MARL consistently achieve the lowest ASR and CR across nearly all configurations, demonstrating superior robustness against adversarial compromise in multi-agent systems.

Table 1: Attack success rate (ASR) and contagion rate (CR) on NetSafe, AutoInject, and UserHijack attack scenarios across chain, tree, and complete graph topology systems. Lower ASR and CR indicate stronger robustness. Best-performing result is highlighted in **bold** and second-best is underlined.

| | | NetSafe | | AutoInject | | | | | | UserHijack | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | AIME | | GPQA | | LiveCodeBench | | AIME | | GPQA | | LiveCodeBench | |
| | | ASR | CR | ASR | CR | ASR | CR | ASR | CR | ASR | CR | ASR | CR | ASR | CR |
| Chain | GPT-3.5 | 10.89% | 11.88% | 3.33% | 3.89% | 3.03% | 3.7% | 5.14% | 5.62% | 15% | 15.56% | 10.61% | 6.79% | 19.24% | 17.76% |
| | GPT-4o-mini | **0%** | **0%** | 3.33% | 3.33% | 5.05% | 5.39% | 1.14% | 1.14% | 3.33% | 7.78% | 4.55% | 8.67% | **2.29%** | **5.64%** |
| | Vanilla-3B | 11.88% | 36.14% | 15% | 19.44% | 19.7% | 22.05% | 16% | 16.57% | 33.33% | 37.78% | 24.24% | 32.59% | 26.29% | 35.27% |
| | Vanilla-7B- | 21.78% | 40.35% | 13.33% | 15% | 21.21% | 21.63% | 7.43% | 8.76% | 25.58% | 25.58% | 17.68% | 21.64% | 22.29% | 28.53% |
| | Challenger-3b | 8.91% | 17.57% | 13.33% | 16.39% | 20.2% | 20.54% | 12.57% | 15.81% | 16.67% | 19.43% | 25.25% | 28.74% | 24% | 21.65% |
| | Inspector-3b | 1.98% | 2.23% | 1.67% | 1.67% | 3.03% | 3.28% | 4.57% | 4.57% | 15% | 9.44% | 11.11% | 18.25% | 14.29% | 16.44% |
| | Challenger-7b | 3.96% | 9.24% | 8.33% | 8.33% | 13.64% | 14.93% | 19.43% | 21.14% | 16.67% | 17.69% | 16.16% | 12.35% | 14.29% | 18.24% |
| | Inspector-7b | 1.98% | 2.72% | **0%** | **0%** | 3.54% | 4.38% | 2.29% | 2.57% | 13.33% | 10.66% | 4.55% | 6.93% | 8% | 7.59% |
| | AdvEvo-MARL-3B (ours) | 6.93% | 35.64% | **0%** | 1.11% | 1.52% | 2.19% | 2.29% | 2.29% | 8.33% | 9.44% | 7.07% | 8.25% | 8.29% | 15.05% |
| | AdvEvo-MARL-7B (ours) | 0.99% | 19.14% | **0%** | 1.11% | **0.51%** | **1.85%** | **0.57%** | **0.19%** | **1.67%** | **0.56%** | **4.04%** | **3.03%** | 6.86% | 6.29% |
| Tree | GPT-3.5 | 8.91% | 9.9% | **0%** | **0%** | **0%** | **0.25%** | 1.71% | 1.71% | 10% | 17.5% | 9.6% | 16.67% | 15.43% | 24.57% |
| | GPT-4o-mini | **0%** | **0%** | 10% | 2.5% | **0%** | 0.38% | 3.43% | 1.57% | 6.67% | **2.92%** | 4.55% | **2.15%** | 6.29% | **3.71%** |
| | Vanilla-3B | 27.72% | 41.34% | 18.33% | 13.75% | 13.64% | 9.34% | 11.43% | 10.29% | 35% | 41.25% | 33.84% | 42.68% | 29.71% | 37.43% |
| | Vanilla-7B | 16.83% | 26.98% | 31.67% | 22.5% | 37.37% | 22.6% | 22.86% | 18.29% | 35% | 33.33% | 26.26% | 28.03% | 29.14% | 29% |
| | Challenger-3b | 22.77% | 38.61% | 3.33% | 1.67% | 4.04% | 1.77% | 4.57% | 3% | 30% | 36.67% | 25.76% | 40.4% | 26.29% | 37.86% |
| | Inspector-3b | 8.91% | 13.86% | 10% | 9.58% | 3.54% | 4.04% | 4.57% | 5.43% | 10% | 30.83% | 9.6% | 30.3% | 10.29% | 28.86% |
| | Challenger-7b | 38.61% | 51.49% | 13.33% | 7.92% | 13.64% | 9.09% | 17.14% | 10.29% | 30% | 34.58% | 24.75% | 26.89% | 21.71% | 27% |
| | Inspector-7b | 8.91% | 12.87% | 3.33% | 4.58% | 4.04% | 3.79% | 1.71% | 3.86% | 5% | 21.25% | 10.61% | 20.45% | 10.29% | 22% |
| | AdvEvo-MARL-3B (ours) | 1.98% | 34.98% | **0%** | 0.56% | 1.01% | 2.86% | 1.71% | 2.29% | 8.33% | 16.11% | **4.04%** | 11.45% | 9.25% | 23.24% |
| | AdvEvo-MARL-7B (ours) | 6.89% | 24.42% | 1.67% | 3.89% | 1.01% | 2.02% | **0%** | **0.19%** | **0%** | 4.44% | 5.05% | 7.24% | **5.14%** | 9.9% |
| Complete | GPT-3.5 | **0%** | 21.53% | **0%** | 22.5% | **0%** | **0.38%** | 1.14% | 2.57% | 26.67% | 42.92% | 18.18% | 37.5% | 34.86% | 51.29% |
| | GPT-4o-mini | **0%** | **1.24%** | **0%** | **0.42%** | 0.51% | 0.51% | **0%** | **0.29%** | **0%** | **0.83%** | **0.51%** | **1.89%** | **1.14%** | **2.43%** |
| | Vanilla-3B | 42.57% | 54.7% | 36.67% | 71.11% | 16.06% | 37.27% | 26.86% | 42.57% | 26.67% | 53.33% | 30.81% | 56.82% | 36% | 65.43% |
| | Vanilla-7B | 33.66% | 43.07% | 33.33% | 67.78% | 31.31% | 43.68% | 25.14% | 39.29% | 28.33% | 48.75% | 29.29% | 48.99% | 34.86% | 60.86% |
| | Challenger-3B | **0%** | 24.5% | 30.69% | 61.67% | 7.58% | 29.55% | 1.14% | 14.43% | 40% | 58.33% | 34.85% | 65.53% | 33.14% | 57.29% |
| | Inspector-3B | 3.96% | 27.23% | 3.33% | 30.83% | 6.03% | 26.89% | 4.57% | 18.14% | 8.33% | 48.75% | 10.61% | 50.25% | 19.14% | 53.14% |
| | Challenger-7B | **0%** | 22.28% | 15% | 58.89% | 22.73% | 37.5% | 15.43% | 18.86% | 38.33% | 53.33% | 33.84% | 50.38% | 29.71% | 46% |
| | Inspector-7B | 2.97% | 17.82% | 5% | 47.78% | 5.57% | 29.47% | 4% | 17.29% | 8.33% | 41.25% | 16.57% | 37.63% | 17.86% | 39.57% |
| | AdvEvo-MARL-3B (ours) | 0.27% | 3.65% | 6.73% | 50.42% | 5.05% | 26.26% | 3.43% | 13.14% | 4% | 47.22% | 17.68% | 33.7% | 16.29% | 34.86% |
| | AdvEvo-MARL-7B (ours) | **0%** | 2.58% | 3.33% | 45.22% | 7.07% | 29.46% | 1.14% | 6.57% | 6.67% | 36.11% | 11.11% | 28.48% | 14.86% | 40.48% |

Specifically, our models maintain ASR consistently below 10% in simpler topology systems such as chain and tree, and remain competitive even in the more challenging complete graph topology, with a maximum ASR of 17.68%, where high interconnectivity greatly facilitates adversarial propagation. In contrast, other open-source baselines frequently fail to maintain low ASR across all topologies. For example, in the tree setting, some models experience up to 38.61% system-level compromise, and in the complete graph setup, ASR can rise as high as 65.53%. In certain cases, these models even underperform relative to their vanilla counterparts: under the UserHijack setting, Challenger-7B reaches 38.33% ASR, a 10% increase over its non-defended variant. Notably, in chain and tree topologies, our models achieve low or even near-zero ASR, often matching or outperforming proprietary models (e.g., GPT-4o-mini).

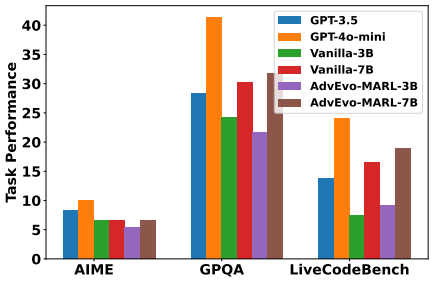



Figure 2: Task benchmark performance. AdvEvo-MARL exhibits minimal degradation and even improved results.

Another key observation is that AdvEvo-MARL maintains low ASR while significantly suppressing CR even as the adversarial setting becomes more aggressive and the communication topology more interconnected. In contrast, many open-source defended baselines exhibit moderate ASR but much higher CR — often ranging from 30% to even 60% in densely connected environments, suggesting insufficient coordination or internal consistency when faced with adversarial attack contagion. Yet our models strive to retain CR below 35% across all evaluation settings. This highlights AdvEvo-MARL's superiority not only to improve individual agents' safety awareness, but also to facilitate collaboration among agents to disrupt adversarial attack spread across the system.

We further evaluate the impact of AdvEvo-MARL on the system's task capabilities across three representative benchmarks. Experimental results in fig. 2 show that our models retain strong task performance, with only a maximum 3% accuracy drop observed among the 3B variants. Notably, the AdvEvo-MARL-7B model being trained exclusively on mathematical tasks, not only preserves its original task competence but even *outperforms* its vanilla counterpart across all datasets, especially those deemed as out-of-distribution. These findings provide clear evidence that safety-oriented training can be achieved without definitely sacrificing task ability. AdvEvo-MARL enables the

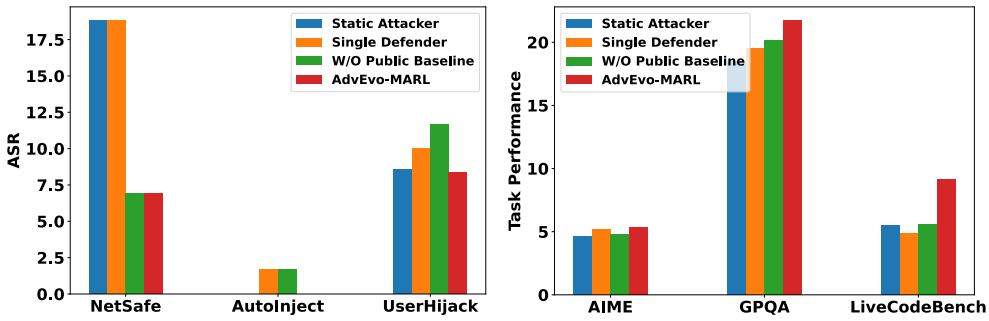



Figure 3: Performance variations under different training configurations. *Left:* robustness performance, AdvEvo-MARL consistently maintains the lowest ASR, *Right:* task performance, AdvEvo-MARL improves task utility across all settings, reaching a maximum 4% gain on LiveCodeBench.

development of agents that are both robust and performant, underscoring its potential as a principled framework for building safe yet capable multi-agent systems.

### 5.3 Dynamic attacks and collaborative defense

To evaluate the effectiveness of dynamic attacker modeling, we compare our MARL-based attacker framework with a static attacker baseline. In the static setting, adversarial prompts are drawn from a fixed pool without adaptation. In contrast, our method enables attackers to continuously generate and refine attack prompts through co-evolution with defenders. As shown in fig. 3, our MARL-based attacker achieves significantly lower ASR under NetSafe threat, revealing a 12% reduction, indicating that defenders trained with evolving attackers exhibit superior robustness. In the AutoInject and UserHijack settings, AdvEvo-MARL also yields marginally lower ASR, suggesting consistent safety improvements across threat models. Evaluations on task datasets also reveal that AdvEvo-MARL outperforms the static attacker baseline across all settings, achieving a maximum 4% performance gain. These results suggest that the presence of dynamic attackers can encourage defenders to develop generalizable task-solving capabilities, highlighting the dual benefits of AdvEvo-MARL for enhancing both safety and utility.

We further investigate how our dynamic attacker evolves throughout the MARL training process. To quantify this progression, we measure the semantic similarity between generated attack prompts and all seed attacks to obtain diversity scores. Notably, as shown in Figure 4, the diversity of adversarial prompts generated by the attacker, reveals a non-monotonic but ultimately increasing trend over the course of training. Despite fluctuations in early stages, the diversity steadily increases in the later phase, indicating that the attackers learns to produce increasingly varied and novel jailbreak attacks. This increased diversity coincides with enhanced robustness in the trained defenders which suggests a causal link. As attackers evolve and diversify, defenders are less likely to overfit and more capable of generalizing to previously unseen threats. These results underscore that training with a dynamic attacker not only produces stronger adversarial prompts but also drives the emergence of more resilient and generalizable defense behaviors.

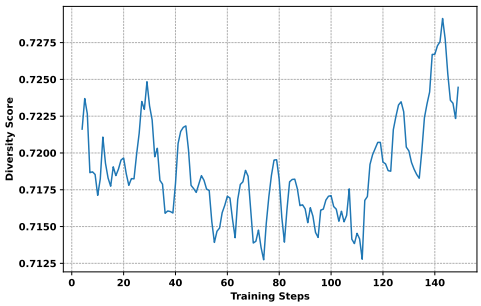
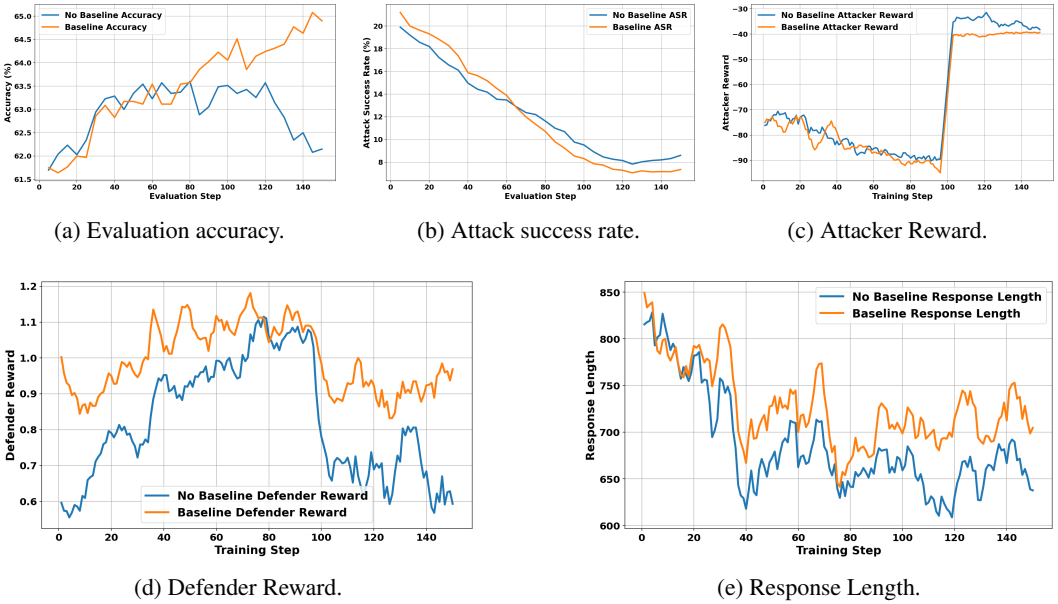



Figure 4: Attacker-generated prompts Diversity.

Another interesting question is whether training defenders in a MAS setting yields benefits over training them individually. Following the setup above, the empirical results in fig. 3 demonstrate that AdvEvo-MARL exhibits the highest system safety and task utility across all evaluated settings. Notably under the NetSafe scenario, our models achieve a 12% gain in robustness and a prominent 4% enhancement in task utility comparatively. These improvements can be attributed to the emergence of collaborative defense behaviors that arise only through joint training. Such coordination and mutual adaptation among agents are difficult to achieve when agents are trained in isolation.

### 5.4 PUBLIC BASELINE BASED MULTI-AGENT REINFORCEMENT LEARNING




(a) Evaluation accuracy. (b) Attack success rate. (c) Attacker Reward.

(d) Defender Reward. (e) Response Length.

Figure 5: Performance comparison of AdvEvo-MARL training with public baseline for advantage estimation (Baseline) and without using public baseline variant (No Baseline).

In this section, we evaluate the effectiveness of our public baseline for advantage estimation in training. As shown in fig. 5, the included public baseline leads to more stable and efficient learning dynamics. Specifically, the public-baseline configuration consistently improves both task utility and system safety, as reflected by steadily improved accuracy and controlled ASR during training and evaluation (fig. 3). In contrast, the standard training setup exhibits non-stationary behavior and even degraded performance in later stages. Moreover, we observe a notable reduction in defender response length under the standard setup, approximately a 13.3% drop during last 50 training steps, indicating a tendency to produce shorter, less informative outputs. This behavior reflects a defensive overcompensation aimed at minimizing risk, but at the cost of task completeness. Finally, as shown in the attacker's reward trajectory, defenders trained with the public baseline are more effective in suppressing the attacker, leading to lower attacker rewards over time, and the defenders also achieve higher rewards via the entire training course. These results provide additional evidence that public-baseline training fosters more robust and generalizable defense policies under adversarial pressure.

## 6 CONCLUSION

We propose AdvEvo-MARL, a multi-agent safety training framework that enhances the robustness and utility of multi-agent systems through co-evolutionary reinforcement learning. By co-training attackers and defenders in a dynamic adversarial environment, AdvEvo-MARL enables agents to continuously adapt to evolving threats, developing stronger and more generalizable defense capabilities. To facilitate collaborative learning and stabilize training, we introduce a public baseline mechanism for advantage estimation, where agents within the same role group (e.g., attackers or defenders) share a common baseline calculated from group-level returns. Extensive experiments across representative attack strategies and task benchmarks demonstrate that AdvEvo-MARL substantially improves system safety while preserving and even enhancing task performance. These results highlight AdvEvo-MARL as a promising and unified framework for building safe and capable multi-agent systems.

## 7 Ethics Statement

Our work aims to enhance safety and task utility of multi-agent systems by explicitly addressing their vulnerabilities through co-evolutionary training of attackers and defenders. We adopt a proactive approach that surfaces how current MAS can be compromised and how defense capabilities can be internalized via reinforcement learning. While our methodology involves generating adversarial attacks, these are used solely for the purpose of strengthening defense mechanisms within multi-agent systems. We believe that open, systematic study of such adversarial attacks is critical for the development of safe and resilient AI systems, and ensuring that MAS have broader positive social impacts.

## References

AIME. Aime problems and solutions, 2025. URL https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions.

Yiqun Chen, Lingyong Yan, Weiwei Sun, Xinyu Ma, Yi Zhang, Shuaiqiang Wang, Dawei Yin, Yiming Yang, and Jiaxin Mao. Improving retrieval-augmented generation through multi-agent reinforcement learning. *arXiv preprint arXiv:2501.15228*, 2025.

Zhaorun Chen, Zhen Xiang, Chaowei Xiao, Dawn Song, and Bo Li. Agentpoison: Red-teaming llm agents via poisoning memory or knowledge bases. *Advances in Neural Information Processing Systems*, 37:130185–130213, 2024.

Sahana Chennabasappa, Cyrus Nikolaidis, Daniel Song, David Molnar, Stephanie Ding, Shengye Wan, Spencer Whitman, Lauren Deason, Nicholas Doucette, Abraham Montilla, et al. Llamafirewall: An open source guardrail system for building secure ai agents. *arXiv preprint arXiv:2505.03574*, 2025.

Falong Fan and Xi Li. Peerguard: Defending multi-agent systems against backdoor attacks through mutual reasoning. *arXiv preprint arXiv:2505.11642*, 2025.

Florian Geissler, Karsten Roscher, and Mario Trapp. Concept-guided llm agents for human-ai safety codesign. In *Proceedings of the AAAI Symposium Series*, volume 3, pp. 100–104, 2024.

Pengfei He, Yupin Lin, Shen Dong, Han Xu, Yue Xing, and Hui Liu. Red-teaming llm multi-agent systems via communication attacks. *arXiv preprint arXiv:2502.14847*, 2025.

Jian Hu. Reinforce++: A simple and efficient approach for aligning large language models. *arXiv preprint arXiv:2501.03262*, 2025.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code, 2024. URL https://arxiv.org/abs/2403.07974.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, 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.

Hongyue Kang, Xiaolin Chang, Jelena Mišić, Vojislav B Mišić, Junchao Fan, and Yating Liu. Cooperative uav resource allocation and task offloading in hierarchical aerial computing systems: A mappo-based approach. *IEEE Internet of Things Journal*, 10(12):10497–10509, 2023.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Shuo Liu, Zeyu Liang, Xueguang Lyu, and Christopher Amato. Llm collaboration with multi-agent reinforcement learning. *arXiv preprint arXiv:2508.04652*, 2025.

Liangbo Ning, Ziran Liang, Zhuohang Jiang, Haohao Qu, Yujuan Ding, Wenqi Fan, Xiao-yong Wei, Shanru Lin, Hui Liu, Philip S Yu, et al. A survey of webagents: Towards next-generation ai agents for web automation with large foundation models. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 6140–6150, 2025.

Zhenyu Pan, Xuefeng Song, Yunkun Wang, Rongyu Cao, Binhua Li, Yongbin Li, and Han Liu. Do code llms understand design patterns? In *2025 IEEE/ACM International Workshop on Large Language Models for Code (LLM4Code)*, pp. 209–212. IEEE, 2025.

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

Tabish Rashid, Gregory Farquhar, Bei Peng, and Shimon Whiteson. Weighted qmix: Expanding monotonic value function factorisation for deep multi-agent reinforcement learning. *Advances in neural information processing systems*, 33:10199–10210, 2020.

Shaina Raza, Ranjan Sapkota, Manoj Karkee, and Christos Emmanouilidis. Trism for agentic ai: A review of trust, risk, and security management in llm-based agentic multi-agent systems. *arXiv preprint arXiv:2506.04133*, 2025.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Erzhuo Shao, Yifang Wang, Yifan Qian, Zhenyu Pan, Han Liu, and Dashun Wang. Sciscigpt: Advancing human-ai collaboration in the science of science. *arXiv preprint arXiv:2504.05559*, 2025.

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

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, et al. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

Jen tse Huang, Jiaxu Zhou, Tailin Jin, Xuhui Zhou, Zixi Chen, Wenxuan Wang, Youliang Yuan, Michael R. Lyu, and Maarten Sap. On the resilience of llm-based multi-agent collaboration with faulty agents, 2025. URL https://arxiv.org/abs/2408.00989.

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.

Shilong Wang, Guibin Zhang, Miao Yu, Guancheng Wan, Fanci Meng, Chongye Guo, Kun Wang, and Yang Wang. G-safeguard: A topology-guided security lens and treatment on llm-based multi-agent systems. *arXiv preprint arXiv:2502.11127*, 2025.

Zhen Xiang, Linzhi Zheng, Yanjie Li, Junyuan Hong, Qinbin Li, Han Xie, Jiawei Zhang, Zidi Xiong, Chulin Xie, Nathaniel D Bastian, et al. Guardagent: Safeguard llm agents via knowledge-enabled reasoning. In *ICML 2025 Workshop on Computer Use Agents*, 2025.

Miao Yu, Shilong Wang, Guibin Zhang, Junyuan Mao, Chenlong Yin, Qijiong Liu, Qingsong Wen, Kun Wang, and Yang Wang. Netsafe: Exploring the topological safety of multi-agent networks, 2024. URL https://arxiv.org/abs/2410.15686.

Boyang Zhang, Yicong Tan, Yun Shen, Ahmed Salem, Michael Backes, Savvas Zannettou, and Yang Zhang. Breaking agents: Compromising autonomous llm agents through malfunction amplification. *arXiv preprint arXiv:2407.20859*, 2024.

Kunlun Zhu, Jiaxun Zhang, Ziheng Qi, Nuoxing Shang, Zijia Liu, Peixuan Han, Yue Su, Haofei Yu, and Jiaxuan You. Safescientist: Toward risk-aware scientific discoveries by llm agents. *arXiv preprint arXiv:2505.23559*, 2025.

## A THE USE OF LLMS

In this work, large language models are used solely for the purpose of grammar correction and language polishing. All technical contributions including conceptual framework design, algorithm development, model training, experiments and paper writing are original and developed by the authors.