Tool-Integrated Reasoning via Hierarchical Multi-Agent Reinforcement Learning

Yiwei Dai[1,*], Hengyi Cai[2,*], Yili Wang[1,*], Zexu Sun[2], Shuaiqiang Wang[2]
Yuchen Li[2], Xin Wang[1,†], Yi Chang[1,†], Dawei Yin[2]

[1]School of Artificial Intelligence, Jilin University
[2]Baidu Inc.
daiyw25@mails.jlu.edu.cn, {xinwang,yichang}@jlu.edu.cn

Abstract

Tool-Integrated Reasoning (TIR) enables large language models to solve complex tasks by invoking external tools during reasoning. Existing methods typically rely on either single-agent reinforcement learning for unified optimization or multi-agent systems for explicit role decomposition. However, single-agent RL often entangles high-level planning with low-level execution, while multi-agent systems may suffer from role mismatch when different roles are optimized independently. To address this limitation, we introduce the principle of *decoupled inference with hierarchical alignment*: high-level planning and low-level execution should remain separated during inference, while their policies should be aligned through shared task feedback during training. We instantiate this principle as **HATR**, a **H**ierarchically **A**ligned framework for **T**ool **R**easoning. At inference time, **HATR** represents each reasoning turn as a tree-structured decision process, where a high-level planning role determines the action branch and step goal, and a low-level execution role realizes this decision through tool use or internal reasoning. During training, **HATR** performs hierarchical multi-agent reinforcement learning by sampling online trajectories, deriving preference signals from task-level rewards, and alternately optimizing policies across decision levels. This cross-level optimization aligns roles across the multi-agent hierarchy while preserving their functional boundaries. Experiments on mathematical reasoning and question-answering benchmarks across multiple backbones show that **HATR** consistently improves task performance over single-agent and multi-agent baselines.

1 Introduction

Large language models (LLMs) [Yang et al., 2025, Liu et al., 2024, Abdin et al., 2024, Chen et al., 2025b, Li et al., 2026, Wang et al., 2026] have demonstrated strong reasoning abilities in mathematical reasoning and question answering. Despite this progress, many complex tasks still require capabilities that are difficult to obtain from parametric generation alone, such as access to up-to-date information, precise symbolic computation and interaction with external environments. Tool-Integrated Reasoning (TIR) addresses this limitation by enabling LLM agents to invoke external tools, including retrievers, calculators and code interpreters, during multi-step problem solving [Jia et al., 2025, Dong et al., 2025a, Li et al., 2025d,c, Zhao et al., 2026]. By incorporating external tool feedback into the reasoning process, TIR extends LLMs from purely text-based inference toward more grounded and verifiable problem solving.

[*]Equal contribution.
[†]Corresponding authors.
[‡]Work done during an internship at Baidu.

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

Reliable TIR requires not only access to external tools, but also effective control over tool-related decisions. A model must determine when tool invocation is necessary, select an appropriate tool, formulate valid tool inputs and integrate the resulting observations into subsequent reasoning. Existing methods generally follow two paradigms. Single-agent approaches optimize planning and execution within a unified policy [Li et al., 2025b, Qian et al., 2025, Feng et al., 2025]. Although this formulation supports end-to-end optimization, it couples high-level control decisions with low-level tool operations, which may limit interpretability and controllability in long-horizon reasoning. Multi-agent approaches instead decompose tool reasoning into specialized roles, such as planners and executors [Li et al., 2025d, Hu et al., 2025]. While this decomposition provides clearer functional structure, the resulting components are often optimized independently or kept partially fixed, which can lead to role mismatch between planning intent and execution behavior.

Such role mismatch is especially harmful in multi-step tool reasoning, where different roles repeatedly condition on each other across steps. When roles are optimized under different objectives or behavior distributions, their decisions can become misaligned across turns. A typical manifestation is that high-level goals may be difficult to realize, while local actions may fail to support global intent. This reveals a deeper *inconsistency between inference-time structure and training-time optimization*: role decomposition improves interpretability during inference, but separated roles must still be aligned under shared task objectives during training. Without such alignment, errors can accumulate across reasoning steps and undermine the reliability of tool use.

To address this issue, we introduce the principle of *decoupled inference with hierarchical alignment*, which bridges inference-time role separation and training-time coordination. The principle keeps planning and execution structurally separated during inference, so that high-level control decisions remain explicit and interpretable. During training, it aligns different roles through shared task feedback, allowing their policies to adapt across decision levels. This design preserves the benefits of role decomposition while reducing mismatch introduced by independently optimized components.

We instantiate this principle in **HATR**, a hierarchical multi-agent framework for tool-integrated reasoning. We focus on a minimal planner–executor instantiation of multi-agent TIR, which separates high-level control from low-level execution while retaining the core coordination problem of multi-agent tool reasoning. During inference, **HATR** decomposes each reasoning turn into a tree-structured interaction, where a high-level planning role determines the reasoning branch and step goal, and a low-level execution role realizes it through tool execution or internal reasoning. This makes control decisions explicit while preserving the functional boundary between planning and execution. During training, **HATR** performs hierarchical policy optimization by sampling online trajectories, constructing preference signals from task and tool-use feedback and alternately updating policies across decision levels. This allows roles to adapt to the behavior distribution of their counterparts, promoting cross-level alignment without collapsing them into a single policy. Experimental results show that **HATR** consistently improves performance over single- and multi-agent baselines.

Our contributions are summarized as follows:

- We identify **role mismatch** as a structural inconsistency in tool-integrated reasoning, arising from the gap between inference-time role decomposition and training-time optimization.
- We propose **HATR**, which realizes decoupled inference with hierarchical alignment: role decomposition is preserved at inference time, while policies are progressively aligned under shared task feedback during training.
- We conduct extensive experiments on math and QA benchmarks, showing that **HATR** improves cross-level coordination and task performance across multiple backbones.

2 Related Works

Tool-Integrated Reasoning (TIR) enables language models to interact with external tools [Jia et al., 2025]. We categorize prior work into three groups. (1) Supervised fine-tuning (SFT) trains TIR systems to imitate tool-use trajectories from manual or model-generated demonstrations [Shim et al., 2025, Wang et al., 2025b, Li et al., 2025a], but limits the exploration to these behaviors, restricting discovery of alternative strategies. (2) Single-agent RL methods use a unified model for reasoning and execution, guiding tool invocation via reward design, staged optimization [Li et al., 2025b, Feng et al., 2025, Qian et al., 2025, Liu et al., 2026b], step-level feedback [Yu et al., 2025],

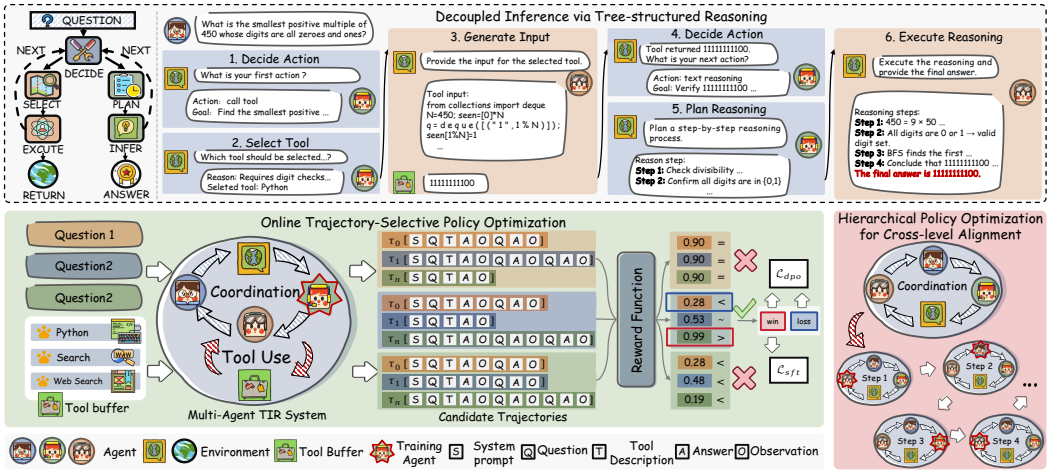


Figure 1: Overview of **HATR**. **HATR** preserves planner–executor decomposition at inference time and aligns role-specific policies through shared task feedback during training.

efficiency improvements [Wang et al., 2025a, Song et al., 2025], retrieval-augmented reasoning [Chen et al., 2025a, Jin et al., 2025, Zhang et al., 2025, Chen et al., 2026, Liu et al., 2026a], or multi-tool support [Dong et al., 2025a]. Unstable trajectory quality in online RL motivates entropy-based dynamic sampling in ARPO [Dong et al., 2025b] and episode filtering in SimpleTIR [Xue et al., 2025]. Related studies further highlight that preference pairs differ in their training utility for reasoning alignment [Wu et al., 2026]. Our approach leverages the role-playing ability of large models to explicitly separate high-level planning from execution. (3) Multi-agent TIR systems implement specialized agents for search or reasoning [Zheng et al., 2025, Lu et al., 2025], but roles are usually fixed and optimization is offline, limiting coordination. OWL [Hu et al., 2025] learns a shared planner from feedback but does not support online multi-agent training, while AgentFlow [Li et al., 2025d] improves multi-turn tool use by fine-tuning the planner while keeping other modules fixed. In contrast, our approach targets role mismatch through online hierarchical multi-agent reinforcement learning, alternately optimizing different roles under shared task feedback to improve cross-level coordination.

3 Preliminary

3.1 Problem Setup

Let $\Pi = \{\pi_i\}_{i=1}^{N}$ denote a multi-agent system (MAS), where each π_i is an instruction-following large language model that generates actions based on the evolving dialogue context. The MAS interacts with a tool buffer $\mathcal{T}$, where each tool $t \in \mathcal{T}$ takes an input argument u, queries external information sources, and returns a textual observation o. Given an input $x \in \mathcal{X}$, the MAS produces an output $\hat{y} \in \mathcal{Y}$ through multi-turn interaction under a maximum turn budget H. We denote the induced decision process by $\mathcal{F}(\cdot\,; \Pi, \mathcal{T}, H)$. We formulate the task as a MAS, such that

$$\mathcal{F}(x; \Pi, \mathcal{T}, H) \to \hat{y}, \tag{1}$$

which produces the system output for a given x. In this work, we study a minimal planner–executor instantiation of multi-agent TIR, which separates high-level control from low-level execution while retaining the core coordination problem of multi-agent tool reasoning.

3.2 Multi-Agent POMDP Formulation

We formalize the tool-integrated multi-agent reasoning process as a finite-horizon partially observable Markov decision process (POMDP):

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, \mathcal{O}, \mathcal{R}, \Pi, H). \tag{2}$$

Here, $\mathcal{S}$ denotes the latent reasoning state space and $s_t \in \mathcal{S}$ is the state at turn t. The action space $\mathcal{A}$ contains internal reasoning actions and optional tool-invocation actions, and the transition kernel P captures state updates conditioned on these actions. The observation space $\mathcal{O}$ includes textual

observations accessible to the agents, such as dialogue context and feedback returned by tools in $\mathcal{T}$. The agent set $\Pi = \{\pi_i\}_{i=1}^{N}$ consists of N cooperative agents, and H is the maximum episode length. At turn t, each agent π_i receives a partial observation $\tau_{i,0:t}$ summarizing all information available before making the next decision. The joint policy then generates a new joint action:

$$\pi(\mathbf{a}_{t+1} \mid \boldsymbol{\tau}_{0:t}) = \prod_{i=1}^{N} \pi_i(a_{i,t+1} \mid \tau_{i,0:t}), \tag{3}$$

where $\mathbf{a}_{t+1} = (a_{1,t+1}, \ldots, a_{N,t+1})$ is the newly generated joint action and $\boldsymbol{\tau}_{0:t} = (\tau_{1,0:t}, \ldots, \tau_{N,0:t})$ is the collection of agent histories available before this action.

The reward is computed for each agent based on its own trajectory and the reference answer y:

$$r(\tau_{i,0:t_{\text{end}}}, y) \in \mathcal{R}. \tag{4}$$

The learning objective is to maximize the expected terminal reward across agents:

$$\max_{\Pi} \ \mathbb{E}_{\Pi,P}\left[\frac{1}{N}\sum_{i=1}^{N} r(\tau_{i,0:t_{\text{end}}}, y)\right]. \tag{5}$$

3.3 Direct Preference Optimization

Direct Preference Optimization (DPO) [Rafailov et al., 2023] is an offline preference-based fine-tuning method that directly updates a target policy π_θ using pairwise comparisons. DPO constructs a closed-form objective based on likelihood ratios between the target policy and a fixed reference policy π_{ref}, eliminating the need for reward models or value estimation.

Given a preference triplet (x, z^+, z^-), where x is the prompt and z^+ and z^- denote the win and loss responses, respectively, the DPO objective is

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,z^+,z^-)\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{\pi_\theta(z^+|x)}{\pi_\theta(z^-|x)} - \beta\log\frac{\pi_{\text{ref}}(z^+|x)}{\pi_{\text{ref}}(z^-|x)}\right)\right], \tag{6}$$

where $\sigma(\cdot)$ is the sigmoid function and $\beta > 0$ controls the preference margin. The objective increases the likelihood of preferred responses while regularizing updates through the reference model, providing a stable and efficient formulation for offline preference learning.

4 Method

4.1 Overview

As illustrated in Figure 1, **HATR** realizes decoupled inference with hierarchical alignment through three components: tree-structured reasoning, online trajectory-selective optimization and hierarchical policy optimization. The framework targets role mismatch in multi-agent TIR by preserving role decomposition during inference and aligning role policies during training. We study a minimal planner–executor instantiation, where the planner models high-level control and the executor models low-level realization. During inference, **HATR** organizes each reasoning turn as a tree-structured decision process that makes branch selection and step goals explicit before execution. During training, **HATR** samples online trajectories, converts shared task feedback into trajectory-level preferences and alternately optimizes policies across decision levels.

4.2 Decoupled Inference via Tree-structured Reasoning

To preserve role separation during inference, **HATR** structures each reasoning turn as a tree-guided multi-agent process. The planner first determines the high-level branch, while the executor realizes the selected branch through tool execution or internal reasoning. This design exposes structural decisions before execution and avoids mixing planning control with low-level generation.

System Overview. **HATR** decomposes problem solving into a sequence of turns indexed by t. Each turn expands a directed tree

$$\mathcal{B}^{(t)} = \big(\mathcal{V}^{(t)}, \mathcal{E}^{(t)}\big), \tag{7}$$

whose nodes represent atomic actions and whose edges encode precedence relations. This structure exposes planning decisions before execution and forms a hierarchical scaffold aligned with the POMDP formulation.

Atomic Action Space. Atomic actions are partitioned as

$$\mathcal{A}_{\text{atom}} = \underbrace{\{\texttt{DECIDE}, \texttt{SELECT}, \texttt{PLAN}\}}_{\mathcal{A}_{\text{pla}}} \cup \underbrace{\{\texttt{EXECUTE}, \texttt{INFER}, \texttt{ANSWER}\}}_{\mathcal{A}_{\text{exe}}} \cup \underbrace{\{\texttt{RETURN}\}}_{\mathcal{A}_{\text{env}}}. \tag{8}$$

Planner actions define high-level structure, executor actions realize execution, and `RETURN` is generated by the environment.

Tool Buffer. **HATR** maintains a tool buffer

$$\mathcal{T} = \{t_j\}_{j=1}^{K}$$

where each tool t_j is identified by a name d_{name} and is associated with a description d_{des}, an input specification d_{in}, an output specification d_{out} and an example d_{ex}. These descriptors define how the tool receives arguments, how its output is interpreted and how the tool is used in typical scenarios.

A deterministic function retrieves the descriptor set given a tool name

$$f(d_{\text{name}}) = (d_{\text{name}}, d_{\text{des}}, d_{\text{in}}, d_{\text{out}}, d_{\text{ex}}). \tag{9}$$

During a `SELECT` action the planner π_{pla} outputs a tool name, and the executor π_{exe} applies the function f to obtain the corresponding input and output specifications and example which guide the construction of valid tool parameters.

Multi-Agent Roles under Partial Observability. Two cooperative agents operate under partial observability. At the beginning of turn t, the planner receives its private history $\tau^{\text{pla}}_{0:t-1}$ and performs the atomic action `DECIDE`:

$$\big(g_t, a_t^{\text{pla}}\big) \sim \pi_{\text{pla}}\big(\cdot \mid \tau^{\text{pla}}_{0:t-1}, \texttt{DECIDE}\big), \tag{10}$$

where g_t is the subgoal and $a_t^{\text{pla}} \in \{\texttt{SELECT}, \texttt{PLAN}\}$ chooses the branch. The textual output of `DECIDE` is appended to the dialogue, producing updated histories $\tau^{\text{pla},+}_{0:t}$.

Tool Branch. If $a_t^{\text{pla}} = \texttt{SELECT}$, the planner chooses a tool identifier

$$t_j \sim \pi_{\text{pla}}\big(\cdot \mid \tau^{\text{pla},+}_{0:t}, g_t, \texttt{SELECT}\big), \tag{11}$$

and retrieves descriptors $f(t_j)$. Next, the executor binds executable parameters

$$u_t \sim \pi_{\text{exe}}\big(\cdot \mid \tau^{\text{exe}}_{0:t-1}, g_t, f(t_j), \texttt{EXECUTE}\big). \tag{12}$$

The environment executes the tool call and returns

$$o_t = t_j(u_t), \tag{13}$$

which is appended to the dialogue and incorporated into $\tau^{\text{pla}}_{0:t+1}$. This design allows observations to be naturally integrated into dialogue history, avoiding the need for explicit loss masking during training.

Internal Reasoning Branch. If $a_t^{\text{pla}} = \texttt{PLAN}$, the planner generates a meta plan

$$p_t \sim \pi_{\text{pla}}\big(\cdot \mid \tau^{\text{pla},+}_{0:t}, g_t, \texttt{PLAN}\big), \tag{14}$$

and the executor π_{exe} produces an output

$$o_t \sim \pi_{\text{exe}}\big(\cdot \mid \tau^{\text{exe}}_{0:t}, g_t, p_t, \texttt{INFER}\big). \tag{15}$$

The output o_t is always appended to the dialogue and incorporated into the planner's observation for turn $t+1$. If the executor π_{exe} emits a final answer

$$\hat{y} \sim \pi_{\text{exe}}\big(\cdot \mid \tau^{\text{exe}}_{0:t}, g_t, \texttt{ANSWER}\big), \tag{16}$$

the answer is appended in the same manner, and the episode terminates without initiating a new turn.

4.3 Online Trajectory-Selective Policy Optimization

After decoupled inference defines how trajectories are generated, we need a policy-level optimizer that can learn from the current multi-agent system. **HATR** samples multiple online trajectories for each input and constructs preference pairs from their rewards. To reduce noisy comparisons from invalid tool calls, redundant actions, or accumulated reasoning errors, we retain only high-margin trajectory pairs for online DPO updates, while using high-reward trajectories for supervised anchoring.

Reward Design. The reward is computed at the completion of each rollout and evaluates both the final prediction and the behavioral properties of the generated trajectory. Given the produced answer y and ground truth $y^\star$, correctness is defined as

$$R_{\mathrm{corr}}(y, y^\star) \in [0, 1], \tag{17}$$

which combines symbolic matching and numerical agreement. Long-horizon tool-integrated reasoning introduces additional variability, so the reward also accounts for tool efficiency, output repetition and trajectory length through, $R_{\mathrm{tool}}(\tau), \quad R_{\mathrm{rep}}(\tau), \quad R_{\mathrm{len}}(\tau)$, where R_{tool} reflects an adaptive preference over the frequency and success rate of tool calls, R_{rep} measures lexical redundancy and R_{len} imposes a nonlinear penalty as the output approaches a predefined maximum length. These components are aggregated as

$$R_{\mathrm{aux}} = \alpha\,(2R_{\mathrm{tool}} - 1) - \beta\,R_{\mathrm{rep}} + \gamma\,R_{\mathrm{len}}, \tag{18}$$

and are normalized to obtain $\widetilde{R}_{\mathrm{aux}} \in [0, 1]$. The final reward interpolates between correctness and auxiliary feedback with an adaptive mixing coefficient ϵ that increases when correctness is low:

$$R(y, \tau) = (1 - \epsilon)\,R_{\mathrm{corr}}(y, y^\star) + \epsilon\,\widetilde{R}_{\mathrm{aux}}. \tag{19}$$

This reward formulation captures the structural characteristics of **HATR** and provides informative preference signals for online DPO under long-horizon, tool-dependent rollouts.

Preference Construction. For each input x, the current multi-agent system generates n trajectories under the current policy, denoted by $\mathcal{T}(x) = \{\tau_1, \ldots, \tau_n\}$, where each trajectory τ_i receives a scalar reward $R(\tau_i)$. The win and loss trajectories are identified as

$$\tau^+ = \arg\max_{\tau_i \in \mathcal{T}(x)} R(\tau_i), \tag{20}$$

$$\tau^- = \arg\min_{\tau_i \in \mathcal{T}(x)} R(\tau_i). \tag{21}$$

To avoid degenerate preference signals, we apply a filtering operator $\Phi(\cdot)$ that discards uninformative pairs. Specifically, a preference pair (τ^+, τ^-) is retained by Φ only if it satisfies

$$R(\tau^+) > \eta \quad \text{and} \quad R(\tau^+) \neq R(\tau^-), \tag{22}$$

where η is a reward threshold. The resulting preference set is defined as

$$\mathcal{P}(x) = \Phi\big(\{(\tau^+, \tau^-)\}\big). \tag{23}$$

Mini-batches $\mathcal{B}$ sampled from $\bigcup_x \mathcal{P}(x)$ are used to perform online DPO updates.

Training Objective. For the selected target policy, training combines online preference optimization with supervised anchoring. The DPO loss increases the likelihood of the preferred trajectory τ^+ over the non-preferred trajectory τ^-, where both trajectories are constructed from online rollouts under the current multi-agent system. The SFT loss further imitates the high-reward trajectory τ^+ to prevent the policy from drifting away from reliable behaviors. The combined objective is

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{DPO}} + \mathcal{L}_{\text{SFT}}. \tag{24}$$

This joint formulation allows the policy to leverage relative preference signals from online trajectory comparisons while retaining strong supervision from high-quality rollouts.

4.4 Hierarchical Policy Optimization for Cross-level Alignment

Trajectory-selective optimization updates a selected policy, but cross-role alignment requires deciding which role to update and under what counterpart behavior. **HATR** performs hierarchical policy optimization. Training proceeds in iterations indexed by k, where online trajectories are generated by the current multi-agent system. At each iteration, one policy is optimized while the other remains fixed, allowing each role to adapt to its current counterpart without collapsing the two policies.

Let $\{\pi_i^{(k)}\}_{i=0}^{N}$ denote the policies of all agents at iteration k. The update at iteration k takes the form

$$\pi_i^{(k+1)} \leftarrow \mathrm{Update}\big(\pi_i^{(k)} \mid \{\pi_j^{(k)}\}_{j\neq i}\big), \tag{25}$$

where the index i is selected according to the hierarchical training schedule. Here $\mathrm{Update}(\cdot)$ denotes the trajectory-selective policy optimization procedure defined in the previous subsection. This formulation aligns each policy under the behavior of its current counterpart. As a result, high-level planning and low-level execution can be progressively aligned under shared task signals, without collapsing their roles into a single policy.

5 Experiment

Our experiments are designed to answer the following questions: **Q1.** Does the proposed method achieve consistent performance improvements across math and QA benchmarks? **Q2.** Does hierarchical policy optimization mitigate role mismatch and promote cross-level alignment? **Q3.** How does hierarchical alignment affect tool-use behavior during reasoning? **Q4.** How does cross-level alignment emerge during training?

5.1 Experiment setting

Data Construction. We construct our experimental data from several publicly available math and question answering datasets, including **NuminaMath-TIR** [Li et al., 2024], **HotpotQA** [Yang et al., 2018], **2WikiMultiHopQA** [Ho et al., 2020], **SimpleDeepSearch** [Sun et al., 2025] and **WebWalker** [Wu et al., 2025]. To obtain data suitable for reinforcement learning, we first evaluate all instances using Qwen2.5-7B-Instruct [Qwen et al., 2025] under a pass@1 setting and filter out instances that can be solved correctly in a single attempt. We then apply DeepSeek-V3 [Liu et al., 2024] with the proposed **HATR** pipeline, discarding instances that cannot be solved within five turns. This filtering ensures that training focuses on non-trivial instances that require multi-step reasoning and tool interaction. From the remaining pool, we randomly sample 3,000 instances for training, maintaining a 2:1 ratio between reasoning and QA examples. Details are in Appendix A.1.

Tool Settings. We equip the model with three external tools (**Search**, **Python**, **WebSearch**). The tool configuration follows Tool-Star [Dong et al., 2025a]. Search uses the Bing Search API to retrieve relevant documents from the internet. Python supports code execution for numerical computation and symbolic reasoning. WebSearch retrieves the top ten relevant web pages through Bing search and produces a concise summary using a locally deployed Qwen2.5-72B-Instruct [Qwen et al., 2025].

Benchmarks. We conduct evaluations on a broad range of benchmarks covering math and QA. The mathematical benchmarks span competition-style and curriculum-based settings, including AIME24 [Zhang and Math-AI, 2024], AIME25 [Zhang and Math-AI, 2025], MATH500 [Lightman et al., 2023], GSM8K [Cobbe et al., 2021], AMC23[1], Gaokao Math Cloze [Zhong et al., 2023], Gaokao Math QA [Zhong et al., 2023], Gaokao2023-EN[2], Gaokao2024-I/II [3], Gaokao2024-Mix [4] and OlympiadBench [5]. For question answering, we consider datasets that require multi-hop reasoning and information aggregation, including WebWalker [Wu et al., 2025], HotpotQA [Yang et al., 2018], 2WikiMultiHopQA [Ho et al., 2020], MuSiQ [Trivedi et al., 2022] and Bamb [Press et al.,

[1] https://huggingface.co/datasets/math-ai/amc23

[2] https://huggingface.co/datasets/MARIO-Math-Reasoning/Gaokao2023-Math-En

[3] https://github.com/llmeval/Llmeval-Gaokao2024-Math

[4] https://github.com/QwenLM/Qwen2.5-Math

[5] https://huggingface.co/datasets/math-ai/olympiadbench

2023]. Evaluation across all benchmarks is performed using pass@1 accuracy, with DeepSeek-v3.1[6] employed as the LLM judge to assess answer correctness.

Models. **HATR** adopts a decoupled architecture. The planner is fixed to **Qwen3-4B-Instruct-2507** [Yang et al., 2025] across all experiments. For the executor, we consider multiple backbone models, including **Qwen3-4B-Instruct-2507** [Yang et al., 2025], **DeepSeek-R1-Distill-Qwen-1.5B** [Guo et al., 2025] and **Phi-4-mini-instruct** [Abdin et al., 2024]. The official thinking mode of all models is disabled. Additional details are in Appendix A.2.

Baselines. We compare with two groups of baselines. Training-free baselines include pure reasoning methods (CoT and BoX) and the agentic baseline ReAct [Yao et al., 2022], which uses the same tools as ours but has no tree-structured role decomposition or RL training. Training-based baselines include ReAct-DPO and SingleAgent. ReAct-DPO applies DPO [Rafailov et al., 2023] training to the ReAct agent. SingleAgent follows our reasoning structure but uses a single LLM trained with DPO to perform both planning and execution through role-playing. In addition, as a supplementary comparison, we evaluate several larger open-source models, including Qwen3-14/32B [Yang et al., 2025], Ministral-8B-Instruct-2410[7] and DeepSeek-R1-0528-Qwen3-8B [Guo et al., 2025], under two prompt settings and report the best performance. We focus on isolating the effect of hierarchical alignment under a controlled optimization setting. Therefore, all training-based baselines are implemented with DPO-style optimization. We do not include PPO- or GRPO-based baselines because they introduce additional differences in training objectives, rollout dynamics and tool configurations, which make direct attribution to hierarchical alignment less controlled. Detailed settings are in Appendix A.4.

5.2 Main results (Q1)

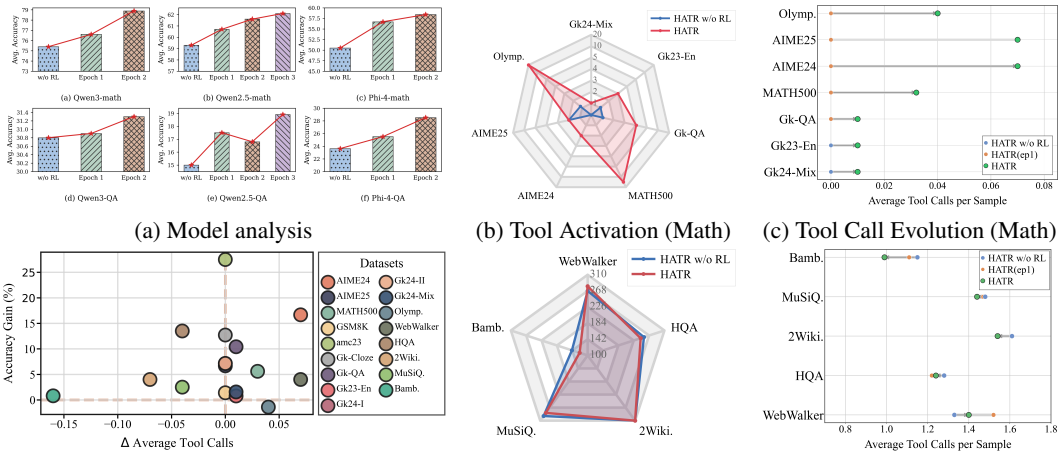
Table 1: Accuracy of **HATR** on math datasets and models. Best in bold, second-best underlined.

Method	AIME24	AIME25	MATH500	GSM8K	AMC23	Gk Cloze	Gk QA	Gk23 En	Gk24 I	Gk24 II	Gk24 Mix	Olymp.	Avg.
Qwen3-14B	23.33	23.33	87.80	**96.20**	72.50	78.81	81.30	79.48	78.57	50.00	73.63	56.20	66.76
Qwen3-32B	36.67	26.67	86.60	95.40	57.50	75.42	85.19	78.44	78.57	64.29	70.88	53.60	67.44
Ministral-3-8B	3.33	0.00	50.40	83.80	25.00	45.76	53.85	50.65	42.86	35.71	40.66	23.40	37.95
DeepSeek-R1-8B	33.33	33.33	88.80	78.00	70.00	**90.68**	87.75	81.30	85.71	71.43	72.53	52.40	70.44
Qwen3-4B-Instruct-2507													
+ boxed	36.67	33.33	88.40	95.00	80.00	83.05	83.19	80.78	85.71	78.57	69.23	54.00	72.33
+ cot	36.67	20.00	88.00	95.20	75.00	82.20	82.22	82.60	78.57	71.43	70.33	53.00	69.60
+ ReAct	36.67	33.33	65.00	81.25	60.00	52.54	66.10	64.94	**100.00**	64.29	61.54	39.20	60.41
+ ReAct-DPO	36.67	40.00	66.40	80.00	67.50	69.49	67.81	60.52	**100.00**	**85.71**	60.44	42.20	64.73
+ SingleAgent	33.33	33.33	89.40	91.40	**85.00**	80.51	87.75	80.26	85.71	78.57	70.33	54.40	72.50
+ HATR w/o RL	43.33	**50.00**	89.80	90.80	**85.00**	80.51	88.60	81.82	85.71	71.43	**82.42**	54.80	75.35
+ HATR	**60.00**	46.67	**90.80**	91.80	**85.00**	81.36	**90.60**	**83.64**	**100.00**	78.57	79.12	**59.00**	**78.88**
DeepSeek-R1-Distill-Qwen-1.5B													
+ boxed	23.33	20.00	81.20	85.20	**65.00**	75.42	80.34	74.81	64.29	50.00	63.74	39.60	60.24
+ cot	30.00	25.00	**82.80**	77.60	**65.00**	**78.81**	79.20	**76.36**	**71.43**	50.00	61.54	41.60	61.61
+ HATR w/o RL	16.67	**30.00**	76.20	**86.60**	50.00	76.27	77.49	71.17	64.29	**57.14**	63.74	41.80	59.28
+ HATR	**33.33**	20.00	77.20	86.20	**65.00**	73.73	**81.97**	70.13	**71.43**	**57.14**	**64.84**	**44.44**	**62.12**
Phi-4-mini-instruct													
+ box	0.00	3.33	34.80	68.80	20.00	34.75	44.16	37.92	28.57	28.57	28.57	8.80	28.19
+ cot	3.33	0.00	34.00	58.20	15.00	29.66	37.32	34.03	35.71	50.00	24.18	8.80	27.52
+ HATR w/o RL	13.33	13.33	68.20	84.60	40.00	56.78	61.54	64.42	57.14	50.00	57.14	**39.00**	50.46
+ HATR	**30.00**	**20.00**	**73.80**	**86.00**	**67.50**	**69.49**	**71.95**	**65.19**	**64.29**	**57.14**	**58.70**	37.60	**58.47**

Math Results. As shown in Table 1, **HATR** improves performance across most mathematical benchmarks and backbone models. Compared with HATR w/o RL, **HATR** achieves further gains, showing that role decoupling alone is insufficient and that hierarchical alignment is needed for reliable multi-step tool reasoning. This suggests that the benefit of **HATR** comes not only from separating planning and execution, but also from cross-role training. The improvements are more evident on challenging benchmarks such as AIME and OlympiadBench, where solving problems requires decomposition, tool-use decisions and reliable execution. Pure reasoning baselines remain competitive on simpler datasets, but degrade on harder ones, suggesting that static prompting becomes less reliable when tool-related decisions are required.

[6] https://huggingface.co/deepseek-ai/DeepSeek-V3.1

[7] https://huggingface.co/mistralai/Ministral-8B-Instruct-2410



(a) Model analysis (b) Tool Activation (Math) (c) Tool Call Evolution (Math)

(d) Accuracy Gain (e) Tool Activation (QA) (f) Tool Call Evolution (QA)

Figure 2: Model and tool-use analysis of the HATR framework.

QA Results. As shown in Table 2, **HATR** achieves higher average performance across the evaluated model settings. Compared with **HATR** w/o RL, **HATR** further improves performance, indicating that hierarchical alignment provides benefits beyond role decomposition. This matters for QA tasks because retrieval-based reasoning requires coordinated decisions about when to search, use evidence and terminate. The gains over ReAct-style baselines suggest that tool access alone is insufficient for reliable multi-step retrieval. Instead, roles must be aligned to translate high-level search intent into effective evidence acquisition and answer generation. Together, these results support pairing inference-time role decoupling with training-time alignment.

Table 2: Accuracy of **HATR** across QA.

Method	WebWalker	HQA	2Wiki	MusiQ	Bamb	Avg.
Qwen3-14B	4.50	27.50	32.50	**12.50**	48.80	25.16
Qwen3-32B	3.00	30.50	31.50	12.00	**53.60**	26.12
Ministral-3-8B	2.50	23.00	22.50	8.50	32.80	17.86
DeepSeek-R1-8B	0.00	4.50	10.50	3.50	13.60	6.42
Qwen3-4B-Instruct-2507						
+ box	2.50	24.50	27.00	9.50	34.40	19.58
+ cot	2.50	25.00	26.50	8.50	31.20	18.74
+ ReAct	8.00	31.25	32.50	10.00	35.20	23.39
+ ReAct-DPO	8.50	31.88	30.00	12.00	40.80	24.64
+ SingleAgent	3.50	30.00	25.50	8.50	35.20	20.54
+ HATR w/o RL	9.00	42.00	**42.50**	10.50	49.60	30.72
+ HATR	**11.00**	**43.50**	41.50	12.00	48.80	**31.36**
DeepSeek-R1-Distill-Qwen-1.5B						
+ box	0.00	0.50	6.25	0.00	0.80	1.51
+ cot	0.50	0.00	0.00	0.50	0.00	0.20
+ HATR w/o RL	2.34	21.00	20.50	6.50	24.80	15.03
+ HATR	**2.50**	**28.00**	**26.00**	**7.81**	**30.40**	**18.94**
Phi-4-mini-instruct						
+ box	1.00	13.50	17.00	4.00	12.00	9.50
+ cot	1.50	14.00	9.00	2.00	16.80	8.66
+ HATR w/o RL	7.50	31.50	29.50	8.00	41.60	23.62
+ HATR	**11.50**	**45.00**	**33.50**	**10.50**	**42.40**	**28.58**

5.3 Ablation Study (Q2)

To address Q2, we analyze stage-wise accuracy during hierarchical training in Figure 2a. Across most model-task settings, performance improves after hierarchical training, suggesting that role decomposition alone is insufficient and that cross-level alignment is necessary to make planner decisions compatible with executor behavior. The intermediate stages show that optimizing a single role can temporarily shift the behavior distribution of the interacting system, while subsequent cross-level optimization helps the two roles readapt to each other. This suggests that role mismatch cannot be fully addressed by optimizing a single role in isolation. Such non-monotonic trends provide evidence of role mismatch during intermediate stages, where improving one policy may temporarily reduce its compatibility with the other. These results support our hypothesis that role decomposition alone is insufficient, and that hierarchical alignment is necessary for reliable multi-step tool reasoning.

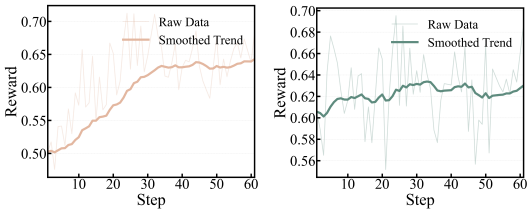
5.4 Tool Usage Analysis (Q3)

To address Q3, we analyze how hierarchical alignment affects tool-use behavior using Δ average tool calls and accuracy gain. Figures 2b and 2e show that **HATR** changes tool activation patterns across math and QA tasks, indicating that hierarchical training reshapes tool-use decisions. Figures 2c and 2f further show that tool-call frequency does not uniformly increase after training. Instead, it varies across datasets and stages, and Figure 2d shows cases where accuracy improves with fewer average tool calls. This suggests that the gains come from better coordination between planning intent

and execution behavior, rather than more tool invocation. These results indicate that hierarchical alignment improves tool-use control by making executor actions better match planner-level decisions.

5.5 Training Dynamics (Q4)

To address Q4, Figure 3 shows planner and executor rewards. Under executor-first alternating training, the executor stabilizes earlier, while the planner improves more gradually with higher variance. This reflects the difficulty of adapting high-level control to evolving executor behavior. Later, both curves become more stable, suggesting that alternating updates reduce role mismatch and improve cross-level coordination.



(a) Executor (b) Planner

Figure 3: Training reward curves on **Qwen3**.

6 Conclusion

We identify role mismatch as a key challenge in tool-integrated reasoning, arising from the gap between inference-time role separation and training-time optimization. To address this issue, we propose decoupled inference with hierarchical alignment, which preserves role decomposition during inference while aligning policies during training. We instantiate this principle in **HATR**, a hierarchical multi-agent framework for tool reasoning. Experiments across math and QA benchmarks show that **HATR** consistently improves performance over single-agent and multi-agent baselines.

Acknowledgments and Disclosure of Funding

This work was supported by a grant from New Generation Artificial Intelligence-National Science and Technology Major Project (No. 2026ZD0127902), the National Natural Science Foundation of China (No.U2341229 and No.62406125), the Reform Commission Foundation of Jilin Province (No.2024C003), the New Cornerstone Science Foundation through the XPLORER PRIZE, the Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China (JYB2025XDXM903), the National Natural Science Foundation of China under grants (No.62372211), the China National Postdoctoral Program for Innovative Talents (No. BX2026297), and the China Postdoctoral Science Foundation (No. 2026M791630).

References

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.

Mingyang Chen, Linzhuang Sun, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z Pan, Wen Zhang, Huajun Chen, et al. Learning to reason with search for llms via reinforcement learning. *arXiv preprint arXiv:2503.19470*, 2025a.

Xinran Chen, Yuchen Li, Hengyi Cai, Zhuoran Ma, Xuanang Chen, Haoyi Xiong, Shuaiqiang Wang, Ben He, Le Sun, and Dawei Yin. Multi-agent proactive information seeking with adaptive llm orchestration for non-factoid question answering. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 4341–4352, 2025b.

Xuanhe Chen, Yuchen Li, Youyi Bi, Shuaiqiang Wang, Linghe Kong, and Dawei Yin. Reflectrag: Enhancing retrieval-augmented generation with grpo-optimized iterative reflection. *Neurocomputing*, 695:134047, 2026. doi: 10.1016/j.neucom.2026.134047.

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

Guanting Dong, Yifei Chen, Xiaoxi Li, Jiajie Jin, Hongjin Qian, Yutao Zhu, Hangyu Mao, Guorui Zhou, Zhicheng Dou, and Ji-Rong Wen. Tool-star: Empowering llm-brained multi-tool reasoner via reinforcement learning. *arXiv preprint arXiv:2505.16410*, 2025a.

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*, 2025b.

Jiazhan Feng, Shijue Huang, Xingwei Qu, Ge Zhang, Yujia Qin, Baoquan Zhong, Chengquan Jiang, Jinxin Chi, and Wanjun Zhong. Retool: Reinforcement learning for strategic tool use in llms. *arXiv preprint arXiv:2504.11536*, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, 2020.

Mengkang Hu, Yuhang Zhou, Wendong Fan, Yuzhou Nie, Bowei Xia, Tao Sun, Ziyu Ye, Zhaoxuan Jin, Yingru Li, Qiguang Chen, et al. Owl: Optimized workforce learning for general multi-agent assistance in real-world task automation. *arXiv preprint arXiv:2505.23885*, 2025.

Xinda Jia, Jinpeng Li, Zezhong Wang, Jingjing Li, Xingshan Zeng, Yasheng Wang, Weinan Zhang, Yong Yu, and Weiwen Liu. Fast, slow, and tool-augmented thinking for llms: A review. *arXiv preprint arXiv:2508.12265*, 2025.

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.

Chengpeng Li, Mingfeng Xue, Zhenru Zhang, Jiaxi Yang, Beichen Zhang, Bowen Yu, Binyuan Hui, Junyang Lin, Xiang Wang, and Dayiheng Liu. START: Self-taught reasoner with tools. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 13523–13564, Suzhou, China, November 2025a. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.683. URL `https://aclanthology.org/2025.emnlp-main.683/`.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. Numinamath tir. `[https://huggingface.co/AI-MO/NuminaMath-TIR](https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf)`, 2024.

Xuefeng Li, Haoyang Zou, and Pengfei Liu. Torl: Scaling tool-integrated rl. *arXiv preprint arXiv:2503.23383*, 2025b.

Yuchen Li, Hengyi Cai, Rui Kong, Xinran Chen, Jiamin Chen, Jun Yang, Haojie Zhang, Jiayi Li, Jiayi Wu, Yiqun Chen, et al. Towards ai search paradigm. *arXiv preprint arXiv:2506.17188*, 2025c.

Yuchen Li, Jiamin Chen, Xinran Chen, Zhiyu Li, Haojie Zhang, Rui Kong, Jiayi Li, Xinyu Ma, Hengyi Cai, Lixin Su, et al. Retain to refine: Adaptive online question answering via query routing and long-short memory. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pages 2312–2322, 2026.

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. In *NeurIPS 2025 Workshop on Efficient Reasoning*, 2025d.

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*, 2023.

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.

Wenhan Liu, Xinyu Ma, Yutao Zhu, Yuchen Li, Daiting Shi, Dawei Yin, and Zhicheng Dou. Agentic-r: Learning to retrieve for agentic search. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 15987–16005. Association for Computational Linguistics, 2026a. doi: 10.18653/v1/2026.findings-acl.785. URL https://aclanthology.org/2026.findings-acl.785/.

Xiaojian Liu, Han Xu, Jianqiang Xia, Zhixuan Li, Ke Xu, Yiwei Dai, Xinran Chen, Changwo Wu, and Yuchen Li. Scope-rl: Optimizing reasoning paths before and after success. *arXiv preprint arXiv:2607.11506*, 2026b. doi: 10.48550/arXiv.2607.11506. URL https://arxiv.org/abs/2607.11506.

Rui Lu, Zhenyu Hou, Zihan Wang, Hanchen Zhang, Xiao Liu, Yujiang Li, Shi Feng, Jie Tang, and Yuxiao Dong. Deepdive: Advancing deep search agents with knowledge graphs and multi-turn rl. *arXiv preprint arXiv:2509.10446*, 2025.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711, 2023.

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.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

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. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Jeonghoon Shim, Gyuhyeon Seo, Cheongsu Lim, and Yohan Jo. Tooldial: Multi-turn dialogue generation method for tool-augmented language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*, 2025.

Shuang Sun, Huatong Song, Yuhao Wang, Ruiyang Ren, Jinhao Jiang, Junjie Zhang, Fei Bai, Jia Deng, Wayne Xin Zhao, Zheng Liu, et al. Simpledeepsearcher: Deep information seeking via web-powered reasoning trajectory synthesis. *arXiv preprint arXiv:2505.16834*, 2025.

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.

Hongru Wang, Cheng Qian, Wanjun Zhong, Xiusi Chen, Jiahao Qiu, Shijue Huang, Bowen Jin, Mengdi Wang, Kam-Fai Wong, and Heng Ji. Acting less is reasoning more! teaching model to act efficiently. *arXiv preprint arXiv:2504.14870*, 2025a.

Pei Wang, Yanan Wu, Noah Wang, Jiaheng Liu, Xiaoshuai Song, ZY Peng, Ken Deng, Chenchen Zhang, Junran Peng, Ge Zhang, et al. Mtu-bench: A multi-granularity tool-use benchmark for large language models. In *The Thirteenth International Conference on Learning Representations*, 2025b.

Yu Wang, Yuchen Li, Rui Kong, Xinran Chen, Jiamin Chen, Hengyi Cai, Shuaiqiang Wang, Jiashu Zhao, Yulun Zhang, Zhonghao Lyu, et al. Swrouter: Similarity-contractive window routing for multi-turn large language model conversations. *arXiv preprint arXiv:2609.11414*, 2026.

Hui Wu, Hengyi Cai, Jinman Zhao, Xinran Chen, Ziheng Li, Zhejun Zhao, Shuaiqiang Wang, Yuchen Li, and Dawei Yin. Not all preferences deserve gradients: Understanding gradient utility in offline reasoning alignment. In *Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing*, 2026. URL `https://arxiv.org/abs/2602.01207`.

Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Linhai Zhang, Yulan He, Deyu Zhou, Pengjun Xie, et al. Webwalker: Benchmarking llms in web traversal. In *Workshop on Reasoning and Planning for Large Language Models*, 2025.

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. In *NeurIPS 2025 Fourth Workshop on Deep Learning for Code*, 2025.

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

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2369–2380, 2018.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*, 2022.

Yuanqing Yu, Zhefan Wang, Weizhi Ma, Shuai Wang, Chuhan Wu, Zhiqiang Guo, and Min Zhang. Steptool: Enhancing multi-step tool usage in llms via step-grained reinforcement learning. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 3952–3962, 2025.

Shaokun Zhang, Yi Dong, Jieyu Zhang, Jan Kautz, Bryan Catanzaro, Andrew Tao, Qingyun Wu, Zhiding Yu, and Guilin Liu. Nemotron-research-tool-n1: Tool-using language models with reinforced reasoning. *arXiv preprint arXiv:2505.00024*, 2025.

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2024, 2024.

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2025, 2025.

Zhejun Zhao, Yuchen Li, Alley Liu, Yuehu Dong, Xiaolong Wei, Lixue Zheng, Pingsheng Liu, Dongdong Shen, Long Xia, Jiashu Zhao, and Dawei Yin. Tura: Tool-augmented unified retrieval agent for ai search. *arXiv preprint arXiv:2508.04604*, 2026.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. *arXiv preprint arXiv:2504.03160*, 2025.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. Agieval: A human-centric benchmark for evaluating foundation models, 2023.

A Experimental details

A.1 Data Statistics

Table 3 reports the overall dataset sizes together with the number of instances incorrectly solved by Qwen-7B-Instruct [Qwen et al., 2025] under the pass@1 setting, covering NuminaMath-TIR [Li et al., 2024], HotpotQA [Yang et al., 2018], 2WikiMultiHopQA [Ho et al., 2020], SimpleDeepSearch [Sun et al., 2025], and WebWalker [Wu et al., 2025]. The *Counts* column shows the total number of instances in each dataset, while *Incorrect* indicates cases that cannot be solved correctly in a single attempt by the base model. Across these datasets, incorrectly solved instances are generally more challenging and often require multi-step reasoning or effective tool usage. We therefore use these error cases as training data for reinforcement learning, as learning from difficult instances is essential for improving decision-making and tool-integrated reasoning performance.

Table 3: Dataset statistics and the number of instances incorrectly solved by Qwen-7B-Instruct under the pass@1 setting.

Dataset	Task Type	Counts	Incorrect
NuminaMath-TIR	Text	72.4K	17.53K
HotpotQA	Text	8.4K	6.84K
2WikiMultiHopQA	Text	0.87K	0.66K
SimpleDeepSearch	TIR	0.87K	0.67K
WebWalker	Text	0.68K	0.67K
Total	–	83.22K	26.37K

A.2 Benchmark and Evaluation Details

For benchmarks with large test sets, we follow a fixed subsampling protocol during evaluation. For mathematical reasoning benchmarks with more than 500 test instances, we evaluate models on a fixed subset of 500 problems sampled from the official test split. For question answering benchmarks with more than 200 test instances, we evaluate models on a fixed subset of 200 examples sampled from the official test split. Subsampling is performed once prior to evaluation, and the same subsets are used for all models and methods.

We additionally specify the context length configurations used for different models. For **Qwen3-4B-Instruct-2507** [Yang et al., 2025] and **Phi-4-mini-instruct** [Abdin et al., 2024], the maximum context length for a single dialogue round is set to 4,096 tokens, while the total context length budget across multi-turn interactions is capped at 40K tokens. For **DeepSeek-R1-Distill-Qwen-1.5B** [Guo et al., 2025], we allow a maximum of 10K tokens per single dialogue round and a total context length of up to 50K tokens for multi-turn conversations. Each example is evaluated with at most 5 turns.

A.3 Baseline Details

This appendix describes the two prompt designs used for the baseline methods. Both prompts are instruction-based and contain no in-context examples. They differ only in the placement of the reasoning and output-format instructions.

- CoT-style Prompt.

```
SYSTEM: Please reason step by step, and put your final answer within
\boxed{}.
USER: {question}
```

- Box-style Prompt.

```
SYSTEM: You are a helpful assistant.
USER: {question}
Please reason step by step, and put your final answer within \boxed{}.
```

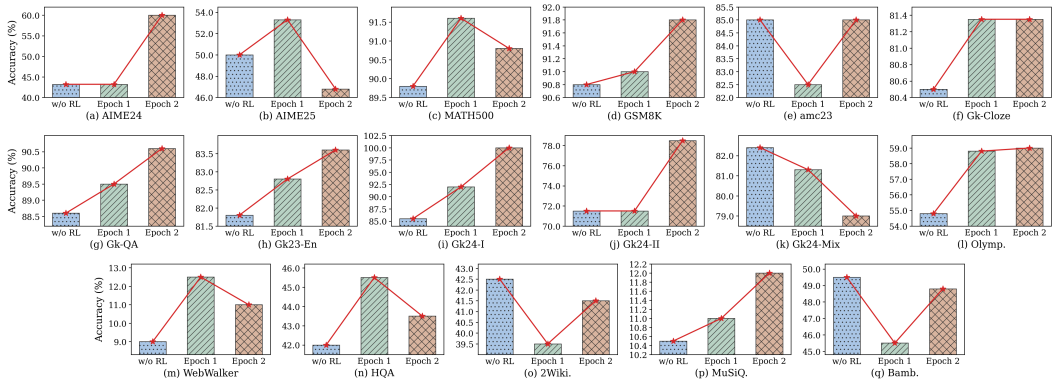


Figure 4: Model analysis of the HATR on **Qwen3**.

A.4 Training Details

All training runs are conducted with 8 A800 GPUs. To improve training efficiency for DeepSeek-R1-Distill-Qwen-1.5B [Guo et al., 2025] and Phi-4-mini-instruct [Abdin et al., 2024], we additionally construct an 8.5K tool-integrated reasoning dataset using DeepSeek-V3 [Liu et al., 2024] for supervised fine-tuning. Auxiliary 6 GPUs are provisioned to host frozen models and tool backends such as WebSearch execution environments to support model interaction.

For all experiments, we use fixed training hyperparameters with α set to 0.22, β set to 0.1, γ set to 0.1, and η is set to 0.50. The training batch size is 48, and the DPO mini-batch size $\mathcal{B}$ is 16. We set the rollout number to 6, restrict the maximum number of rounds to 3, and use a learning rate of 1×10^{-6}. When **Qwen3-4B-Instruct-2507** [Yang et al., 2025] is used as the executor, both the planner and the executor are trained with a maximum response length of 4096 tokens and a maximum prompt length of 9000 tokens. When **DeepSeek-R1-Distill-Qwen-1.5B** [Guo et al., 2025] is used as the executor, the maximum response length is 4096 tokens and the maximum prompt length is 12000 tokens. When **Phi-4-mini-instruct** [Abdin et al., 2024] is used as the executor, the maximum response length is 4096 tokens and the maximum prompt length is 8000 tokens. For all SFT warm-up runs, the maximum sequence length is set to 20444 tokens, and models are trained for 6 epochs with a learning rate of 5×10^{-6} and a weight decay of 0.1, selecting the checkpoint with the minimum validation loss.

B Model Analysis

B.1 Fine-grained Training Dynamics of Hierarchical Reinforcement Learning

This appendix provides a fine-grained analysis of training dynamics in hierarchical reinforcement learning (HRL), complementing the main results with stage-wise performance across models and benchmarks. As shown in Figures 4, 5 and 6, updating execution alone yields limited gains, suggesting that improving a single role in isolation is insufficient. When high-level training is introduced, performance becomes more variable, with temporary regressions on some tasks, indicating mismatch between roles during intermediate stages. After full system-level optimization, these fluctuations are largely reduced and performance becomes more consistent across tasks. Overall, this analysis supports our motivation that cross-level alignment, rather than role separation alone, is critical for reliable multi-step tool reasoning.

B.2 Error Analysis

We define four error types: Task Decomposition Error (planning objectives are incomplete or logically flawed), Logical Inference Error (invalid deductions or contradictions), Tool Error (incorrect tool input, output, or result usage), and Format or Constraint Violation (outputs violating structural, formatting, or task requirements). For math tasks, we sampled 100 failed examples from the Olymp. dataset; for QA tasks, 50 failed examples from the Bamb dataset. GPT-4o was used as the judge to quantify error distributions and provide both quantitative and qualitative insight into system successes

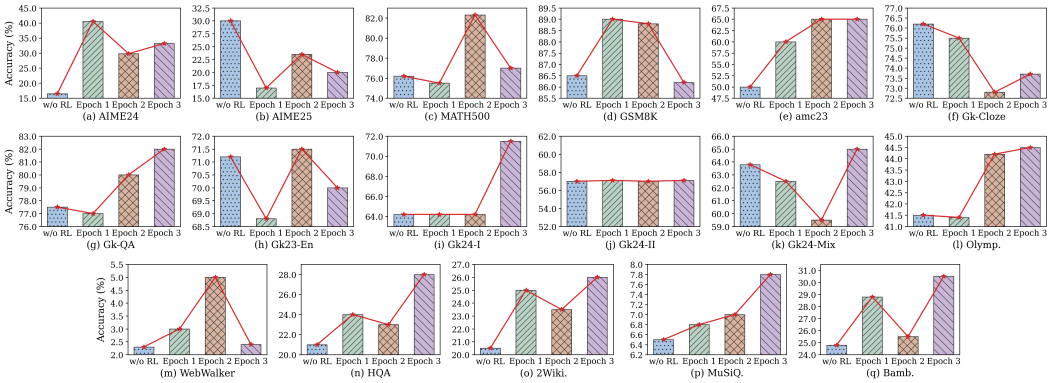


Figure 5: Model analysis of the HATR on **Qwen1.5B**.

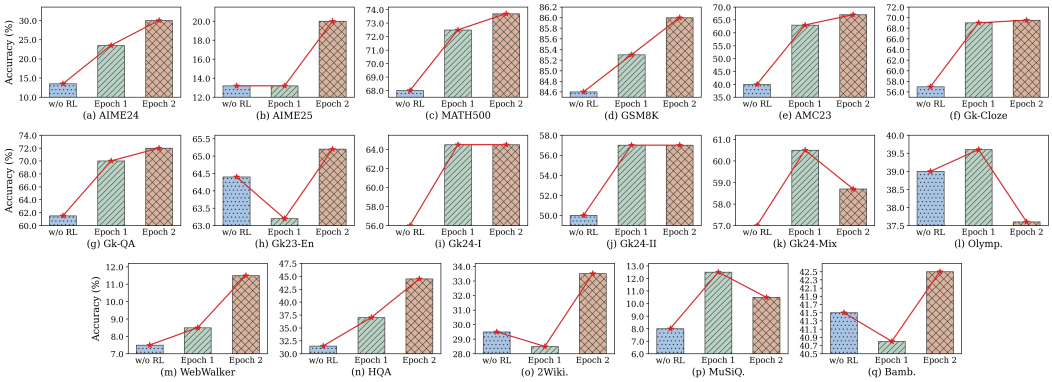


Figure 6: Model analysis of the HATR on **Phi-4**.

and failures. Results in Table 4 show that:(1) After hierarchical RL, Task Decomposition Error, Tool Error, and Format or Constraint Violations are substantially reduced, demonstrating the effectiveness of HRL in improving structured reasoning. (2) Error distributions are largely consistent across different model backbones, indicating the robustness and generalizability of our approach. (3) Tool Errors are more frequent in QA tasks compared to math tasks, likely due to the increased complexity of multi-step reasoning and the inherent variability of online tool execution, rather than any limitation of the proposed method. We encourage future research to explore more robust search tools and incorporate fault-tolerant mechanisms to further mitigate tool-related errors.

B.3 Limitations

While **HATR** yields promising results, the hierarchical reinforcement learning process involves a complex optimization space that requires careful hyperparameter tuning. Furthermore, our current evaluation primarily focuses on math and QA tasks. Further exploration is needed to assess the framework's scalability in open-domain scenarios with a larger and more dynamic set of tools.

C Broader Impacts.

HATR aims to improve the reliability, interpretability, and controllability of tool-integrated reasoning systems. This may benefit applications that require multi-step reasoning, external information access, and symbolic computation. At the same time, as with other tool-using agents, practical deployment should consider appropriate oversight, especially in sensitive scenarios where incorrect tool use or over-reliance on automation may affect users.

Table 4: Error type distribution across methods, models and tasks.

Method	Model	Task	Task Error	Inference Error	Tool Error	Format Violation
HATR w/o RL	Qwen3-4B	Math	0.17	0.77	0.02	0.04
HATR	Qwen3-4B	Math	0.13	0.86	0.00	0.01
HATR	DeepSeek-1.5B	Math	0.14	0.81	0.01	0.04
HATR	DeepSeek-1.5B	QA	0.14	0.72	0.14	0.00

```
You are an intelligent planning assistant, responsible for breaking down
complex problems into well-defined sub-tasks.
Your responsibilities are:
1.  Determine whether an external tool needs to be invoked;
2.  If so, clearly specify the tool usage objective and select the most
appropriate tool;
3.  If not, construct a clear and logical reasoning path in natural language;
4.  Strictly follow formatting requirements - all decisions must be output in
the required structured format in subsequent prompts.
You will collaborate with other modules to complete the task.  Do not repeat
the original question, and do not reduce the scope of the problem.
```

Figure 7: System prompt of the planner.

D Instructions

Figures 7–13 present the complete prompt templates used to instantiate the planner–executor interaction in our framework. Figure 7 defines the planner's role and its responsibilities for task decomposition, action selection, and goal specification, while Figure 8 provides the original question together with the available tools and their functional descriptions. Figures 9–11 correspond to the three planner actions. Specifically, `DECIDE` determines whether the current step requires tool use or internal reasoning and specifies the step goal, `SELECT` chooses the appropriate tool according to this goal, and `PLAN` constructs a structured reasoning plan. Figures 12 and 13 define the executor actions. The `INFER` prompt guides the executor to follow the proposed plan and produce intermediate reasoning or a final answer, whereas the `EXECUTE` prompt converts the tool-use objective into a valid input for the selected tool. All prompts adopt structured output formats to ensure consistent information transfer between the planner and executor and to support reproducible implementation of the proposed framework across different tasks and tool configurations.

```
Please solve the following problem step by step:
{question}
You may use the following tools to assist you in this task:
[
  {
    "tool_name": "Search",
    "tool_description": "Used for retrieving factual knowledge (e.g., people,
places, definitions), with an emphasis on accuracy and source reliability."
  },
  {
    "tool_name": "WebSearch",
    "tool_description": "Used for web scraping, list comparison, and
organizing structured information."
  },
  {
    "tool_name": "Python",
    "tool_description": "Used for mathematical calculations, simulations,
enumeration, logical verification, graph computation, and other complex
reasoning tasks."
  }
]
You should decide whether to invoke any tools based on the problem, and perform
reasoning or construct inputs in subsequent steps.
```

Figure 8: Question Prompt of Planner

Please determine whether the next step requires tool usage or can be completed via language reasoning.
- If tool usage is needed, output: <ACTION>call_tool</ACTION>
- If it can be solved directly, output: <ACTION>text_reasoning</ACTION>
- Then, use <GOAL>...</GOAL> to clearly describe the analysis or computation objective for this step.
Your output must strictly follow the XML tag format below:
<ACTION>...</ACTION>
<GOAL>...</GOAL>
Requirements for GOAL
- The description must be clear and concise, yet fully informative;
- Do not omit key conditions, variable definitions, or task context;
- Avoid vague phrases such as ‘‘enumerate’’ or ‘‘judge whether’’ without further explanation;
- The GOAL must be self-contained and executable without relying on prior context;
- Encourage tool use where helpful.
Examples
<ACTION>call_tool</ACTION>
<GOAL>Enumerate all bijections f from set $A = \{1, 2, 3, 4, 5, 6\}$ to itself such that $f(f(f(x))) = x$</GOAL>
<ACTION>text_reasoning</ACTION>
<GOAL>Analyze all positive integers $a = 2^n \cdot 3^m$ such that a^6 does not divide 6^a</GOAL>
Do not output anything other than the specified tags above.

Figure 9: DECIDE Prompt of Planner.

Based on the tool usage objective proposed in the previous step, select the most appropriate tool from the available list and briefly explain your reasoning.
Tool Usage Objective:
{goal}
The tool name must be selected from the provided list. Use the following output format:
<REASON>(your explanation) </REASON>
<SELECTED_TOOL>(tool name) </SELECTED_TOOL>

Figure 10: SELECT Prompt of Planner.

Based on the current reasoning objective, plan a structured, step-by-step reasoning process.
Each step should clearly specify the operation to perform and the underlying logic, progressively guiding the task toward the final answer.
Reasoning Objective:
{goal}
Please output strictly in the following format:
<REASONING_STEPS>
Step 1: ...
Step 2: ...
...
Step N: ...
</REASONING_STEPS>
Note: The steps should describe only the reasoning path. Do not include or infer the final answer at this stage.

Figure 11: PLAN Prompt of Planner.

Please execute the following original question, reasoning objective, and reasoning plan step by step. For each step, provide both the operation performed and its result.
If the question involves multiple candidate values (e.g., numbers, options, or combinations), explicitly list and analyze all of them. Skipping relevant data or cases is strictly prohibited.
--
Original Question
{question}
Current Reasoning Objective
{goal}
Reasoning Plan
{plan}
--
Output Format
<THINK>
Step 1: Perform xxx, the result is xxx.
Step 2: ...
...
</THINK>
<ANSWER>\[\\boxed{{final_answer}} \]</ANSWER>
(Only fill this in after the full reasoning is complete)
Instructions
- You must follow the reasoning plan **exactly as given**, step by step -- do not rewrite, skip, merge, or alter any steps;
- Each step must include both the action performed and its result;
- If reasoning is still in progress, output only the <THINK>block without the <ANSWER>tag;
- In the <ANSWER>block, provide the **complete final answer**, not just a label;
- Use \\boxed{{...}} to wrap all answers, whether mathematical, symbolic, or textual;
- The boxed content must be sufficient to match or evaluate against the ground truth;
- **Only output the <ANSWER>tag if the final answer contains sufficient information to reconstruct the Original Question** (key entities, quantities, units, and task intent). Otherwise, output only the <THINK>block and omit <ANSWER>.

Figure 12: INFER/ANSWER Prompt of Executor.

Based on the following original question, tool usage objective and tool description, generate a valid input for the selected tool.
--
Original Question
{question}
Tool Usage Objective
{goal}
Tool Description
{tool_information}
--
Output Format
<TOOL_INPUT>
(Fill in the tool input content here)
</TOOL_INPUT>
Only output the <TOOL_INPUT>block. Do not include any extra statements, quotation marks, comments, explanations, or markdown.

Figure 13: EXECUTE Prompt of Executor.

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 claims made in the abstract and introduction are consistent with the experimental results and the scope of the proposed framework. We clearly state the contributions, assumptions, and limitations and avoid overstating the generalization ability.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss limitations in the main text and provide a more detailed analysis in the appendix B.3.

   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: [N/A]

Justification: This paper does not include formal theoretical results requiring assumptions or proofs.

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 sufficient details about the model architecture, training procedure, datasets, and evaluation protocol to enable reproduction of the main results.

   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: [No]

Justification: We do not release code and data at submission time, but we provide sufficient implementation details to support reproducibility.

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 describe the experimental settings in the main text and appendix A.4.

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: [No]

Justification: We do not report statistical significance metrics such as error bars, as each experiment is conducted with a fixed configuration and a single run for efficiency. We observe consistent improvements across multiple datasets and model backbones, which suggests that the performance gains are robust.

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: We provide information in Appendix A.4.

   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 confirm that this research complies with the NeurIPS Code of Ethics and does not involve any unethical practices.

   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: We discuss both positive impacts and potential risks in Appendix C.

    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: The proposed method does not involve releasing high-risk models or datasets that require specific safeguards.

    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 datasets, models, and codebases used in this work and respect their respective licenses and terms of use.

    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: [N/A]

    Justification: This work does not introduce new datasets, models, or code assets for release.

    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 work does not involve crowdsourcing or 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 work does not involve human subjects and therefore does not require IRB approval.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

    Answer: [Yes]

    Justification: LLMs are a core component of the proposed method, and their usage is fully described in the methodology and experimental sections.

    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.