

Why Multi-Step Tool-Use Reinforcement Learning Collapses and How Supervisory Signals Fix It

Yupu Hao^{1,2}, Zhuoran Jin^{1,2}, Huanxuan Liao^{1,2}, Kang Liu^{1,2}, Jun Zhao^{1,2*}

¹The Key Laboratory of Cognition and Decision Intelligence for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China

²School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
{haoyupu2023, liaohuanxuan2023}@ia.ac.cn, {zhuoran.jin, kliu, jzhao}@nlpr.ia.ac.cn

Abstract

Tool use enables large language models (LLMs) to perform complex tasks, and recent agentic reinforcement learning (RL) methods show promise for enhancing model capabilities. However, RL alone often leads to instability or limited gains in tool-use tasks. In our experiments, some models exhibit catastrophic collapse, where performance abruptly drops and tool-invocation structures fail. The analysis reveals that these failures stem from unexpected probability spikes in specific control tokens, disrupting structured execution, yet the underlying tool-use capability remains intact, merely obscured by specific formats. To address this, we systematically investigate a diverse set of supervisory signals, including off-policy supervision, hint-based guidance, erroneous example supervision, and others, applied under both synchronous and interleaved training schemes. We find that interleaving supervised fine-tuning (SFT) with RL substantially improves stability, but exhibits degraded performance under format and content out-of-distribution (OOD) evaluation. We also analyze the impact of learning rates and generalization across settings. These results highlight the importance of understanding RL failures and demonstrate how diverse supervisory signals can guide exploratory learning, enabling robust training of LLMs for complex, multi-step tool-use tasks. Our Code is available at <https://github.com/hypasd-art/Tool-RL-Box>.

1 Introduction

Tool use plays an increasingly important role in large language models (LLMs) (Li et al., 2023; Mialon et al., 2023; Li et al., 2026). By leveraging external tools, LLMs can function as intelligent agents capable of autonomously performing complex interactive tasks (Qu et al., 2025; Yao et al.,

2024). However, the multi-step and structured nature of tool-based interactions, together with the diversity of tool feedback, introduces substantial challenges for improving model capabilities. To mitigate these issues, prior works have focused on improving tool-use performance through optimized interaction frameworks (Qin et al., 2024), large-scale synthesis of high-quality trajectories (Hao et al., 2025b; Liu et al., 2025b; Prabhakar et al., 2025b), and refined supervised or unsupervised training methodologies (Liu et al., 2025b; Lu et al., 2025). Particularly recent advances in agentic RL (Lu et al., 2025; Wang et al., 2025; Mai et al., 2025) have motivated growing interest in RL as a more principled framework for agentic interaction, as agentic RL has demonstrated strong performance across a variety of complex tasks (Zhang et al., 2025b; Wu et al., 2025).

However, in practice, RL-based optimization often suffers from **training instability** and **limited performance gains** in tool-use settings. As shown in Figure 1, erroneous response frequencies can rise sharply at certain stages. In our experiments, we observe a surprising failure mode where model performance can abruptly collapse to near zero, accompanied by a breakdown of valid tool-invocation structures. Through detailed analysis, we find that these failures are not caused by a loss of reasoning ability, but are instead driven by unexpected probability amplification of specific control tokens, which gradually distorts structured generation into degenerate execution patterns.

These observations indicate that RL alone is insufficient for stable, long-horizon structured generation. We find supervised fine-tuning on high-quality tool-use trajectories provides a strong initialization, improving early performance and preventing collapse. Nonetheless, prior work suggests that models trained with SFT can be limited in terms of generalization and performance compared to RL (Chu et al., 2025). Therefore, we

*Corresponding Author

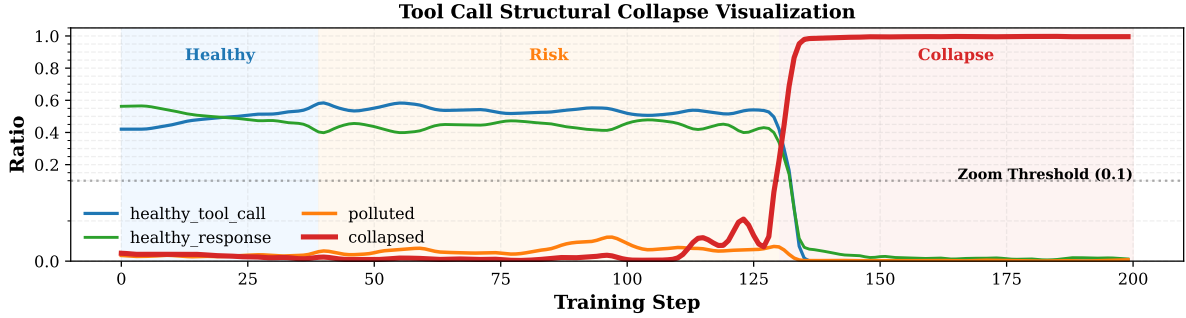


Figure 1 The changes of token appearing frequency in the RL training process of model Qwen2.5-1.5B-Instruct. Polluted indicates that irrelevant special tokens appear in the output, whereas Collapsed indicates that the output is completely invalid or incorrect.

aim to **investigate how different supervisory signals interact with RL optimization and whether supervision can fundamentally stabilize multi-turn agentic training, attempting to combine them to mitigate their shortcomings.** Yet, a key open question remains: *how should supervisory signals be integrated into RL training, and which forms of supervision are most effective for enhancing stability?*

To address this issue, we systematically investigate a broad family of supervisory signals, including SFT then RL, off-policy supervision, and erroneous trajectory supervision, under both synchronous and interleaved training paradigms. Additionally, we introduce Hint-based guidance by prepending correct hints before the model generates responses to help generation, and remove these hints during optimization. We also propose Process Reflection Supervision, which reflections are extracted from intermediate reasoning steps in the training process for the summarization the strengths and weaknesses of each steps alongside original error trajectories to construct the SFT data, further improves both stability and final performance. Our results show that simply mixing supervision with RL is insufficient: synchronous methods often suffer from distribution mismatch, while interleaved training with supervised fine-tuning (SFT) significantly improves stability but may degrade performance under format and content-level out-of-distribution (OOD) evaluation.

We further analyze key factors affecting stability and generalization, including learning rate sensitivity and cross-setting transfer behavior. Across these studies, we find that supervisory signals play a central role in controlling structural behavior during RL, beyond what scalar rewards alone can

achieve.

Overall, our work highlights that agentic RL failure is primarily a *structural collapse problem* rather than a capability limitation, and demonstrates that carefully designed supervisory signals can effectively regulate token-level execution patterns, enabling more stable and robust tool-use learning in LLMs.

Our work makes three key contributions:

- **We identify structural collapse as a fundamental failure mode in multi-turn agentic RL.** We show that RL on tool-use tasks does not primarily degrade reasoning ability, but induces a *structural collapse* in which generation degenerates into malformed control-token sequences, breaking tool-invocation structure while preserving underlying task competence.
- **We uncover a token-level mechanism underlying RL instability.** We find that RL disproportionately amplifies special control tokens, leading to a redistribution of policy mass. This reveals that instability arises from *control-token dynamics* rather than capability degradation, providing a mechanistic explanation for collapse.
- **We conduct a systematic study of supervisory signals for stabilizing multi-turn agentic RL.** We systematically explore six supervisory configurations across two model families and categorize them into synchronous and interleaved paradigms. We further introduce Process Reflection Supervision, which converts intermediate trajectory information into textual supervisory signals. Empirically, interleaved training consistently im-

proves stability and generalization, while synchronous methods are prone to distribution mismatch.

2 Related Works

2.1 Tool Learning

Recent advances have extended LLMs with tool-use capabilities (Hao et al., 2025a; Prabhakar et al., 2025a; Qin et al., 2024), enabling interaction with external APIs and environments beyond text generation (Lu et al., 2025). Through tool invocation, models can perform task execution (Li et al., 2023; Trivedi et al., 2024), information retrieval (Jin et al., 2025), and planning (Xie et al., 2024), improving their ability to solve complex tasks efficiently. Prior works have enhanced tool-use capabilities through reasoning frameworks (Qin et al., 2024; Yao et al., 2023), supervised fine-tuning (Lu et al., 2025; Zhang et al., 2025d; Li et al., 2025), and reinforcement learning (Hu, 2025; Shao et al., 2024). Although RL improves tool-invocation performance (Qian et al., 2025; Jiang et al., 2025; Xue et al., 2025), its effectiveness strongly depends on the base models prior tool knowledge. Smaller or weakly initialized models often fail to benefit from RL, while high-quality SFT data alone can already provide strong performance. This motivates incorporating supervisory signals into RL to stabilize multi-turn interactions and improve tool-use learning.

2.2 Expert Trajectories in Reinforcement Learning

RL improves models through trajectory exploration and reward-based updates (Shao et al., 2024; Srivastava and Aggarwal, 2025), but can stagnate when sampling quality is poor. Recent works mitigate this by incorporating expert or ground-truth trajectories into RL (Fu et al., 2025; Liu et al., 2025a). For instance, LUFFY (Yan et al., 2025) replaces part of sampled trajectories with expert ones and reweights low-probability tokens, while ReLIFT (Ma et al., 2025) alternates RL with targeted SFT. Other methods use partial correct answers (Zhang et al., 2025a,c; Huang et al., 2025; Zhan et al., 2025) or retrieved experiences (Chen et al., 2025; Fang et al., 2025). However, these methods are mainly studied in single-turn reasoning tasks (e.g., mathematics), and their effectiveness in multi-turn tool-use settings remains under-explored.

3 Preliminary

We represent multi-turn tool utilization trajectory τ by LLMs as a sequence of actions and feedbacks: $\tau = (a_1, r_1, a_2, r_2, \dots, a_T, r_T)$. Each action $a_t \in \mathcal{A}$ is drawn from an action space \mathcal{A} that includes both **feasible tool invocations** and **natural language responses**. At each step, the feedback $r_t \in \mathcal{R}$ is given based on current action and the preceding trajectory:

$$r_t = R(a_t | a_1, r_1, \dots, a_{t-1}, r_{t-1}) \in \mathcal{R} \quad (1)$$

where \mathcal{R} denotes the feedback space, encompassing both **environment responses** (e.g., tool execution outputs) and **user responses**.

Conditioned on the accumulated interaction history, the model samples the next action:

$$a_t \sim P(a_t | a_1, r_1, \dots, a_{t-1}, r_{t-1}) \quad (2)$$

4 Method

In this section, we first analyze the *training instability and limited performance gains* observed under pure reinforcement learning settings. By examining model-generated trajectories during training, we identify the underlying causes of these failures. In contrast, introducing supervised fine-tuning before RL substantially improves stability and performance, highlighting the importance of supervisory signals. Motivated by this observation, we further conduct a systematic study of different supervisory signals and integration strategies in agentic RL, analyzing their effects on stability, structural behavior, and generalization.

4.1 Experiment Setup

Dataset. We conduct experiments on BFCL-V3 (Patil et al., 2025), a multi-turn tool-use benchmark where models must invoke multiple tools across interactive environments to answer user queries and follow-up questions. We focus on four challenging settings: *Base*, *Miss Func*, *Miss Param*, and *Long Context*. For training, we randomly sample 100 instances from each of the first three settings, resulting in 300 questions following previous work (Lu et al., 2025), which proves the performance can be improved by small amount of data, while reserving the remaining data for evaluation. Due to excessive context length, *Long Context* is excluded from training. We also augment supervised fine-tuning with converted ToolACE

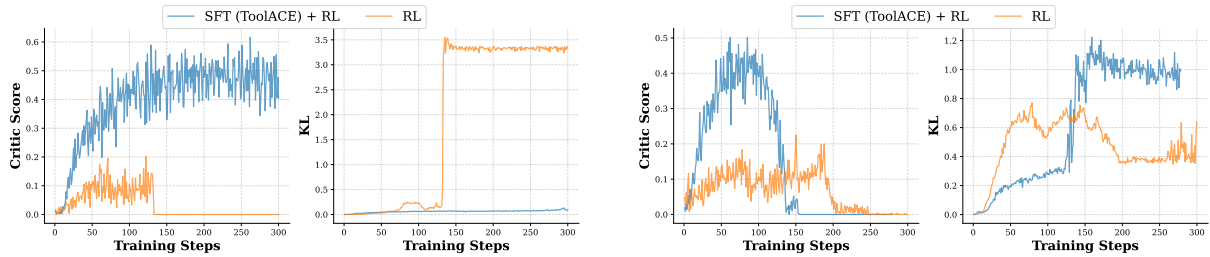


Figure 2 The training dynamics during the training process on the BFCL-V3 dataset. **Left:** Qwen2.5-1.5B-Instruct. **Right:** Qwen3-1.7B.

(Liu et al., 2025b) data to study the impact of training distribution. More details please refer to Appendix B

Model. For our analysis experiments, we use Qwen2.5-1.5B-Instruct (Team et al., 2024) and Qwen3-1.7B (Yang et al., 2025) as base models. During the SFT stage, we decompose each multi-turn interaction into multiple single-turn instances and train the model on these isolated steps. In contrast, during RL, we adopt a full multi-turn tool-use setting, where the model must complete the entire interaction trajectory, including all tool invocations, to receive positive feedback. For Qwen3-1.7B, we use the non-thinking variant. Further implementation details and reward design are provided in Appendix B.

4.2 Problem of catastrophic collapse in the RL

As shown in Figure 2, direct RL on Qwen2.5-1.5B-Instruct often causes **catastrophic collapse**, with sudden reward drops and sustained KL divergence spikes. Qwen3-1.7B exhibits milder instability, but pure RL gains remain limited and may degrade over time. Applying SFT on an additional tool-use dataset before RL stabilizes training: for Qwen2.5-1.5B-Instruct, reward and KL curves evolve smoothly; for Qwen3-1.7B, RL may still occasionally collapse after SFT, but achieves substantially higher performance prior to instability.

4.3 Analysis of the reasons of collapse

By jointly analyzing model-generated trajectories throughout RL training, we identify a consistent pattern of *structural collapse*. Importantly, we find that the observed collapse does not primarily reflect a degradation of the models underlying capability. Instead, the collapse progressively shifts generation toward malformed control-token patterns that corrupt the structural integrity of tool invocation. Specifically, special control tokens used

in tool-calling formats, such as `<tool_call>` and `<|im_end|>`, become increasingly confused during training. As RL optimization proceeds, these tokens are over-amplified and incorrectly combined as shown in Tabel 3, causing the model to drift from valid tool-use trajectories toward degenerate termination patterns under particular prompt formats.

To systematically characterize this phenomenon, we categorize model outputs into four structural states. **Healthy Tool Call** denotes well-formed tool invocations that strictly follow the required schema. **Healthy Response** corresponds to natural-language responses and contains no tool-related artifacts. **Text Pollution** represents an intermediate failure stage where incomplete tool tags or misplaced control tokens begin to appear. Finally, **Collapsed** denotes severe degeneration, where generation converges to meaningless minimal termination sequences such as `<tool_call><|im_end|>` with almost no semantic content.

As shown in Figure 1, Qwen2.5-1.5B-Instruct initially maintains a clear separation between **tool-call** and **text-response** generation pathways, producing structurally valid outputs during early training. However, as RL progresses, fragmented control-token patterns increasingly appear within natural-language responses (see Table 3), indicating progressive corruption of structural boundaries. In later stages, the probability mass becomes dominated by minimal termination sequences `<tool_call><|im_end|>`, leading to complete structural collapse. Detailed definitions and additional results are provided in Appendix A.

These results suggest that instability in multi-turn tool-use RL arises from insufficient structural supervision during exploration. Without constraints that preserve tool-invocation structure, RL disproportionately reinforces certain control tokens, gradually redistributing policy mass toward

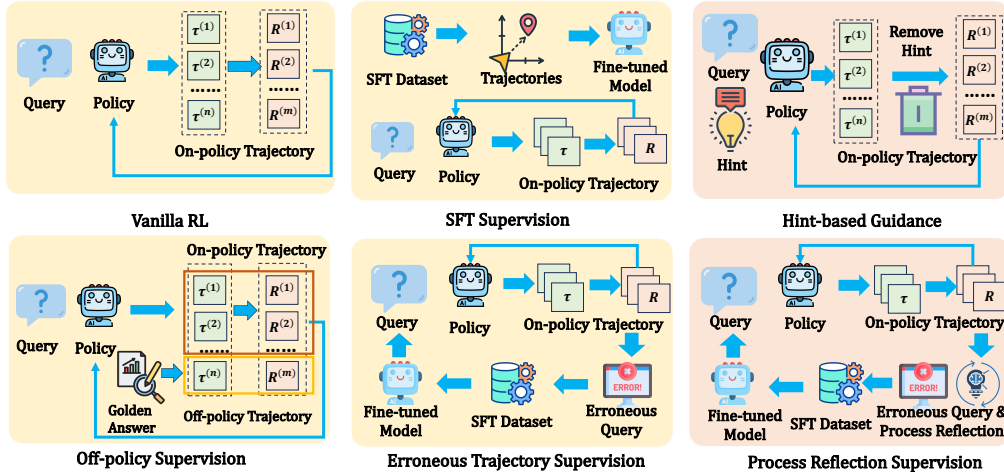


Figure 3 The training frameworks of different methods, where the Hint-based Guidance and Process Reflection Supervision is proposed by our.

degenerate execution patterns. Interestingly, results in Table 2 further show that direct GRPO training remains relatively stable under format changes in OOD evaluation. This indicates that the underlying tool-use capability is not fully lost during collapse; instead, performance is highly sensitive to control-token distributions that can either expose or temporarily mask the learned capability.

Overall, our analysis reveals that tool-use RL is fundamentally more sensitive to structural token dynamics than standard reasoning tasks. Without proper supervision or constraints, these tokens can be easily conflated, leading to structural contamination and, ultimately, training collapse.

4.4 How to improve the collapse problem

Unlike prior works studying isolated supervision strategies, we propose a unified framework to analyze how supervisory signals interact with multi-turn agentic RL. We organize methods into synchronous and interleaved paradigms, enabling a controlled comparison of their effects on stability, structural collapse, and generalization in tool-use RL. Within this framework, we systematically study five supervisory signals: SFT-then-RL, off-policy supervision, hint-based guidance, erroneous trajectory supervision, and process reflection supervision. Structural diagrams are shown in Figure 3, with reward details in Appendix B.

SFT Supervision (SFT then RL) This supervisory signal follows a staged pipeline where SFT is performed before RL. In the SFT phase, the policy π_θ is trained to imitate expert trajectories τ given

queries q from dataset D_{SFT} :

$$\mathcal{L}_{\text{SFT}}(\theta) = -\frac{1}{|D_{\text{SFT}}|} \sum_{(q, \tau) \in D_{\text{SFT}}} \sum_{t=1}^{|\tau|} \log \pi_\theta(a_t | q, \tau_{<t}), \quad (3)$$

this objective encourages instruction-following and basic reasoning abilities. However, its effectiveness is constrained by dataset coverage and lacks the ability to explore behaviors beyond the provided demonstrations.

Off-policy Supervision (OPS) Some methods like LUFFY replace the sampled responses with distilled content or ground truth responses (Yan et al., 2025). Inspired by these, we introduce off-policy supervision by partially replacing model-generated rollouts with ground-truth trajectories, thereby strengthening guidance from correct behaviors. Specifically, we mix high-quality SFT trajectories with on-policy rollouts. For each trajectory τ_i , the normalized advantage is computed as:

$$\hat{A}_i = \frac{R(\tau_i) - \text{mean}(G_{\text{on}} \cup G_{\text{off}})}{\text{std}(G_{\text{on}} \cup G_{\text{off}})}, \quad (4)$$

where G_{on} and G_{off} denote the reward sets from on-policy and off-policy trajectories, respectively. Unlike LUFFY (Yan et al., 2025), which modifies the importance ratio for off-policy data, we apply the same policy ratio to all trajectories $r_{k,t}(\theta) = \pi_\theta(\tau_{k,t} | q, \tau_{k,<t}) / \pi_{\theta_{\text{old}}}(\tau_{k,t} | q, \tau_{k,<t})$ to fairly compare the performance of different supervisory signals without other factors. This yields the off-policy supervision surrogate objective:

$$\mathcal{J}_{\text{OPS}}(\theta) = \frac{1}{Z} \sum_{\tau_i \in \mathcal{D}_{\text{on}} \cup \mathcal{D}_{\text{off}}} \sum_{t=1}^{|\tau_i|} \text{CLIP}(r_{k,t}(\theta), \hat{A}_i, \epsilon), \quad (5)$$

where $Z = \sum_i |\tau_i|$. This formulation treats on-policy and off-policy trajectories uniformly, enabling high-quality demonstrations to guide learning while preserving on-policy exploration.

Hint-based Guidance (HBG) To guide the model towards correct trajectories, we prepend a hint \mathcal{H}_q to the query q during sampling. The hint provides a textual description of how to approach or solve the task, and can be obtained either through manual annotation or generated automatically by a large language model. At each timestep, the trajectory is generated token by token as:

$$\tau_{i,t} \sim \pi_{\theta}(\tau_t \mid q, \tau_{i,<t}, \mathcal{H}_q), \quad (6)$$

for policy optimization, the hint \mathcal{H}_q is removed, and the sampled trajectory τ_i is treated as a standard on-policy rollout conditioned only on q . This design allows hints to influence exploration while ensuring that the learned policy does not depend on auxiliary guidance at inference time.

Erroneous Trajectory Supervision (ETS)

Some methods adopt the interleaved training paradigm of SFT and RL in the training process. Inspired by these (Ma et al., 2025; Lai et al., 2025), we adopt an interleaved RL and SFT paradigm into the tool-learning tasks, constructing SFT data explicitly from RL failure cases. Starting from an initial policy π_{θ_0} , RL collects interaction trajectories, and any query whose sampled trajectories all fail is labeled as erroneous. We then create supervised training data from these failures using the ground-truth solutions to enhance model capability.

Formally, at iteration k , the error-derived supervision set is defined as:

$$\mathcal{E}_k = \bigcup_{q \in Q} \{(q, \tau_{\text{label}}) \mid \forall \tau' \in \mathcal{T}_q, R(\tau') = 0\}, \quad (7)$$

where \mathcal{T}_q denotes the set of trajectories sampled for query q , and $R(\tau)$ is the reward, τ_{label} is the ground truth answer. For each trajectory, we construct supervised training pairs conditioned on the same state or prefix. The resulting supervision loss is:

$$\mathcal{L}_{\text{err}}(\theta) = -\frac{1}{|\mathcal{E}_k|} \sum_{(q,\tau) \in \mathcal{E}_k} \sum_{t=1}^{|\tau|} \log \pi_{\theta}(a_t \mid q, \tau_{<t}), \quad (8)$$

where a_t denotes the ground-truth action at step t . We interleave this corrective SFT step with standard RL updates, and gradually increase the proportion of RL steps over iterations, following the scheduling strategy in ReLIFT (Ma et al., 2025).

Process Reflection Supervision (RPS) In multi-turn agentic RL, sampled trajectories contain rich intermediate information beyond terminal success or failure. Prior work (Tan et al., 2025) used process rewards for guidance, but scalar rewards offer only coarse supervision and often miss nuanced reasoning or tool-use errors. We propose *process reflection supervision*, converting implicit trajectory information into explicit textual guidance.

Specifically, trajectories from the current policy π_{θ_k} are fed into an LLM (or auxiliary analyzer) to generate textual reflections analyzing intermediate decisions, structural mistakes, and reasoning gaps, denoted r_{τ}^{ref} . The resulting reflection-augmented dataset is:

$$\mathcal{R}_k = \{(q, \tau, r_{\tau}^{\text{ref}})\}, \quad (9)$$

to further improve the models adherence to correct tool-use formats, we jointly train with erroneous trajectories collected as described in Section 4.4. The reflection loss is defined as:

$$\begin{aligned} \mathcal{L}_{\text{ref}}(\theta) = & -\frac{1}{|\mathcal{R}_k|} \sum_{(q,r_{\tau}^{\text{ref}}) \in \mathcal{R}_k} \log \pi_{\theta}(r_{\tau}^{\text{ref}} \mid q) \\ & -\frac{1}{|\mathcal{E}_k|} \sum_{(q,\tau) \in \mathcal{E}_k} \sum_{t=1}^{|\tau|} \log \pi_{\theta}(a_t \mid q, \tau_{<t}), \end{aligned} \quad (10)$$

the prompt and examples are in Appendix F.

5 Experiments

5.1 Dataset and Models

We train the models on *Qwen2.5-1.5B-Instruct* and *Qwen3-1.7B* and evaluated them on BFCL-V3. To assess generalization, we further tested the models on ACEBench. Detailed training settings are provided in the Appendix B.

5.2 Main Results

Based on Table 1, both Qwen2.5-1.5B-Instruct and Qwen3-1.7B show negligible baseline performance, highlighting the difficulty of multi-turn tool-use without targeted training.

For Qwen2.5-1.5B-Instruct, SFT on BFCL raises the performance floor, with subsequent GRPO providing further refinement, while GRPO alone yields no meaningful gains and its stochasticity often causes catastrophic collapse, indicating that prior structural supervision is indispensable for agentic RL. Among supervisory signals, Off-Policy Supervision and Hint-Based Guidance offer limited benefits. KL divergence during synchronous training is substantially higher than in

Model	Method	Base	Δ	Miss F.	Δ	Miss P.	Δ	Long C.	Δ	Average	Δ
Qwen2.5-1.5B	Vanilla	4.0	-	5.0	-	1.0	-	4.0	-	3.50	-
	SFT _{BFCL}	15.0	(+15.0)	4.0	(+14.0)	6.0	(+16.0)	7.0	(+8.0)	16.75	(+13.25)
	GRPO	0.0	(-4.0)	0.0	(-5.0)	0.0	(-1.0)	0.0	(-4.0)	0.0	(-3.5)
	SFT _{BFCL} + RL	21.0	(+17.0)	22.0	(+17.0)	19.0	(+18.0)	7.0	(+3.0)	17.25	(+13.75)
	SFT _{ToolACE} + RL	23.0	(+19.0)	23.0	(+18.0)	13.0	(+12.0)	10.0	(+6.0)	17.25	(+13.75)
	OPS	1.0	(-3.0)	3.0	(-2.0)	1.0	(0)	1.0	(-3.0)	1.50	(-2.0)
	HBG	0.0	(-4.0)	0.0	(-5.0)	0.0	(-1.0)	0.0	(-4.0)	0.0	(-3.5)
	ETS	26.0	(+22.0)	25.0	(+20.0)	16.0	(+15.0)	13.0	(+9.0)	20.0	(+16.5)
	PRS	31.0	(+27.0)	25.0	(+20.0)	26.0	(+25.0)	21.0	(+17.0)	25.75	(+22.25)
	Qwen3-1.7B	Vanilla	14	-	11	-	14	-	11	-	12.5
SFT _{BFCL}		23	(+9)	20	(+9)	23	(+9)	14	(+3)	20	(+7.5)
SFT _{ToolACE}		12	(-2)	10	(-1)	5	(-9)	8	(-3)	8.75	(-3.75)
GRPO		2	(-13)	0	(-11)	2	(-13)	2	(-10)	1.5	(-11.0)
SFT _{BFCL} + RL		0	(-14)	0	(-11)	0	(-14)	0	(-11)	0	(-12.5)
SFT _{ToolACE} + RL		0	(-14)	0	(-11)	0	(-14)	0	(-11)	0	(-12.5)
OPS		0	(-14)	0	(-11)	0	(-14)	0	(-11)	0	(-12.5)
HBG		1	(-13)	0	(-11)	1	(-13)	1	(-10)	0.75	(-11.75)
ETS		26	(+12)	27	(+14)	19	(+5)	21	(+10)	23.25	(+10.75)
PRS		23	(+9)	22	(+11)	17	(+3)	16	(+5)	19.5	(+7.0)

Table 1 Experimental results on Qwen series models. Best results within each model group are highlighted in **bold**. Detailed training dynamics of individual metrics are provided in the Appendix D.

standard RL (Figure 5), likely due to rapid output distribution shifts when trajectories are not sampled strictly on-policy, destabilizing optimization. This calls for a more granular policy-update and sampling synchronization framework. Process Reflection Supervision achieves the highest average score (25.75), followed by Erroneous Trajectory Supervision; notably, PRS’s reasoning-heavy format, though diverging from standard tool-calling syntax, provides critical logical scaffolding that enhances tool-use competence. These results suggest interleaved training, decoupling SFT-like structural updates from RL exploration is fundamentally more stable than synchronous training.

For Qwen3-1.7B, trends are consistent, but SFT prior to RL does not yield strong improvements. As shown in Figure 15, the model initially outperforms direct RL but eventually suffers catastrophic collapse. Based on Section C, we attribute this instability to the model’s “thinking” mode disrupting structured trajectory learning during training sampling.

5.3 The Analysis of Learning Rate

We evaluate the effect of learning rate on Qwen2.5-1.5B-Instruct across training configurations in Figure 4. A conservative learning rate of 1×10^{-6} yields decreased gains, suggesting that overly cautious updates cannot stabilize multi-turn agentic behaviors. Increasing the learning rate to 1×10^{-5} improves SFT (BFCL), while SFT (ToolACE) ini-

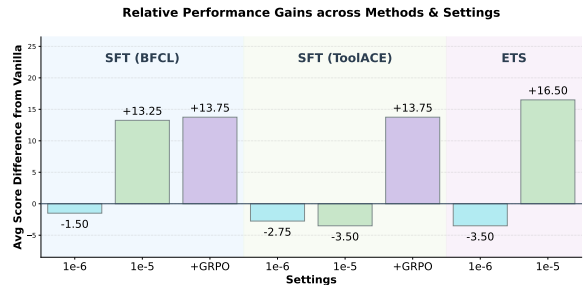


Figure 4 Results on Qwen2.5-1.5B-Instruct under different learning rate. The Y-axis illustrates performance gains and losses relative to the Vanilla baseline (0.0 line), while the X-axis represents various hyperparameter configurations and training stages, with +GRPO indicating RL application on models trained at a 1×10^{-5} lr.

tially suffers from distribution mismatch but later achieves substantial recovery after RL, indicating that RL can reactivate capabilities suppressed by divergent SFT data. When combined with ETS, the larger learning rate consistently improves all metrics, showing that error-driven supervision is highly sensitive to optimization scale. Overall, stabilizing multi-turn training may require larger learning rates than those commonly used in RL, which may explain instability in synchronous training under shared learning rates.

5.4 The Analysis of Generalization

We evaluate model generalization under out-of-distribution (OOD) settings using the original

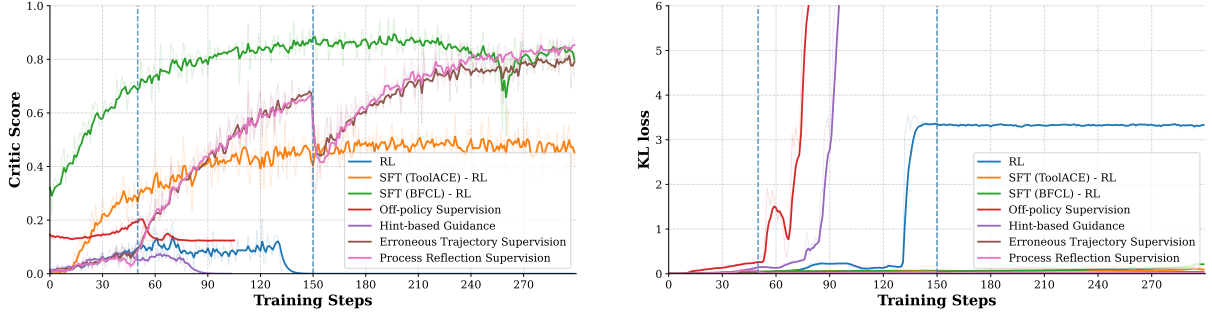


Figure 5 The training dynamics during the training process on the BFCL-V3 dataset of Qwen2.5-1.5B-Instruct.

Method	Format and Content OOD					Content OOD				
	Ad.	Sw.	sf.	Multi Turn	Avg	Ad.	Sw.	sf.	Multi Turn	Avg
Vanilla	38.0	20.0	46.0	0.0	26.00	40.0	32.0	52.0	2.5	31.6
SFT _{BFCL}	0.0	0.0	0.0	0.0	0.00	30.0	20.0	44.0	2.5	24.1
GRPO	32.0	24.0	43.0	0.0	24.75	0.0	0.0	0.0	0.0	0.0
SFT _{BFCL} + RL	0.0	0.0	0.0	0.0	0.00	8.0	8.0	10.0	0.0	6.5
OPS	36.0	24.0	43.0	0.0	25.75	20.0	16.0	50.0	6.7	23.2
HBG	28.0	20.0	42.0	0.0	22.50	0.0	0.0	0.0	0.0	0.0
ETS	10.0	0.0	25.0	0.0	8.75	26.0	22.0	36.0	2.5	21.6
PRS	16.0	8.0	24.0	0.0	12.00	26.0	30.0	43.0	4.2	25.8

Table 2 Generalization results of Qwen2.5-1.5B-Instruct on ACEBench. Left: Format and Content OOD. Right: Content OOD. Best results are in bold, where **Ad.** denotes the Adjust subset, **Sw.** represents the Switch subset, **sf.** indicates single function. Green background means containing isolated SFT stage.

ACEBench framework, which features unseen tool types and prompt structures. Many prior studies prioritize scenario generalization without format generalization. We report both generalization axes to analyze the role of supervisory signals.

Defining ID vs. OOD in our setup. We distinguish content OOD from format OOD. Content OOD evaluates models on tool types and scenarios absent from the in-distribution (ID) training data. Format OOD is instantiated by ACEBenchs fixed invocation template, whereas ID evaluation (including training) follows Qwens native format. Therefore, we observe a general performance drop in OOD scenarios, caused by both factors. Experimental results yield several critical insights:

Format- and content-sensitive overfitting during SFT. In Table 2, SFT methods that excel in-distribution suffer sharp degradation under Format and Content OOD in simple single-step settings but not on Content OOD. This supports our claim that standard SFT induces severe format overfitting, tethering the model to training-time syntax.

RL “collapse” as a format artefact. Methods that appear unstable during training (e.g., GRPO-

only, OPS, HBG) exhibit more stable behavior under Format and Content OOD tests, while Content OOD in same tool format experiments show decline. This confirms that the observed instability is format-specific (token-level probability shifts) rather than a global loss of reasoning or linguistic competence, as hypothesized in Section 4.3. The Multi Turn results further indicate limited generalization during training.

Process-Level Regularization. While interleaved training can degrade single-turn performance, the integration of Process Reflection Supervision (PRS) effectively mitigates this decline. In addition, these results indicate that isolated SFT stages are more prone to inducing format overfitting: while they can substantially improve in-distribution performance, they often harm generalization unless complemented with process-level supervision. Unlike isolated SFT which prioritizes output-format optimization, PRS encourages the model to internalize the underlying logical scaffolding of tool use. This suggests that incorporating reasoning-heavy, process-oriented data during the SFT phase acts as a crucial logical regularizer, enhancing the model’s resilience to distribu-

tion shifts in real-world agentic environments.

6 Conclusion

This work examines reinforcement learning for multi-turn tool-use, identifying two key failure modes: training instability and limited gains. Naive RL can over-amplify control tokens, disrupting structured tool use and causing collapse or plateaus. We systematically study supervisory signal integration, finding interleaved training more stable and effective than synchronous. We also analyze learning rate, SFT data distribution, and generalization, offering a principled framework for stable, effective agentic RL in tool-use settings.

Limitations

This work indicates and analyzes failure modes in multi-turn tool-use tasks, highlighting the critical role of specific tokens and investigating how different intervention signals help mitigate these issues. However, the training data used in this study is relatively limited due to the limited open-source verifiable tool invoking environment, and the impact of data scale on the results is not explored.

Ethics Statement

Our work does not introduce ethical concerns. This paper utilized AI assistance for language polishing of the manuscript, including vocabulary correction and spell checking.

References

- Silin Chen, Shaoxin Lin, Xiaodong Gu, Yuling Shi, Heng Lian, Longfei Yun, Dong Chen, Weiguo Sun, Lin Cao, and Qianxiang Wang. 2025. [Swe-exp: Experience-driven software issue resolution](#). *CoRR*, abs/2507.23361.
- Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. 2025. [Sft memorizes, rl generalizes: A comparative study of foundation model post-training](#). *arXiv preprint arXiv:2501.17161*.
- Runnan Fang, Yuan Liang, Xiaobin Wang, Jialong Wu, Shuofei Qiao, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. 2025. [Memp: Exploring agent procedural memory](#). *CoRR*, abs/2508.06433.
- Yuqian Fu, Tinghong Chen, Jiajun Chai, Xihuai Wang, Songjun Tu, Guojun Yin, Wei Lin, Qichao Zhang, Yuanheng Zhu, and Dongbin Zhao. 2025. [Srft: A single-stage method with supervised and reinforcement fine-tuning for reasoning](#). *arXiv preprint arXiv:2506.19767*.
- Yupu Hao, Pengfei Cao, Zhuoran Jin, Huanxuan Liao, Yubo Chen, Kang Liu, and Jun Zhao. 2025a. [CITI: enhancing tool utilizing ability in large language models without sacrificing general performance](#). In *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 23996–24004. AAAI Press.
- Yupu Hao, Pengfei Cao, Zhuoran Jin, Huanxuan Liao, Yubo Chen, Kang Liu, and Jun Zhao. 2025b. [Evaluating personalized tool-augmented llms from the perspectives of personalization and proactivity](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 21897–21935. Association for Computational Linguistics.
- Jian Hu. 2025. [REINFORCE++: A simple and efficient approach for aligning large language models](#). *CoRR*, abs/2501.03262.
- Qihan Huang, Long Chan, Jinlong Liu, Wangui He, Hao Jiang, Mingli Song, Jingyuan Chen, Chang Yao, and Jie Song. 2025. [Boosting MLLM reasoning with text-debiased hint-grpo](#). *CoRR*, abs/2503.23905.
- Dongfu Jiang, Yi Lu, Zhuofeng Li, Zhiheng Lyu, Ping Nie, Haozhe Wang, Alex Su, Hui Chen, Kai Zou, Chao Du, Tianyu Pang, and Wenhui Chen. 2025. [Verltool: Towards holistic agentic reinforcement learning with tool use](#). *CoRR*, abs/2509.01055.
- Bowen Jin, Hansi Zeng, Zhenrui Yue, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. [Search-r1: Training llms to reason and leverage search engines with reinforcement learning](#). *CoRR*, abs/2503.09516.
- Hanyu Lai, Xiao Liu, Yanxiao Zhao, Han Xu, Hanchen Zhang, Bohao Jing, Yanyu Ren, Shuntian Yao, Yuxiao Dong, and Jie Tang. 2025. [Computerrl: Scaling end-to-end online reinforcement learning for computer use agents](#). *Preprint*, arXiv:2508.14040.
- Jiachun Li, Zhuoran Jin, Tianyi Men, Yupu Hao, Kejian Zhu, Lingshuai Wang, Dongqi Huang, Longxiang Wang, Shengjia Hua, Lu Wang, and 1 others. 2026. [Agentic environment engineering for large language models: A survey of environment modeling, synthesis, evaluation, and application](#). *arXiv preprint arXiv:2606.12191*.
- Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. 2023. [Api-bank: A comprehensive benchmark for tool-augmented llms](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 3102–3116. Association for Computational Linguistics.

- Xuefeng Li, Haoyang Zou, and Pengfei Liu. 2025. [Torl: Scaling tool-integrated RL](#). *CoRR*, abs/2503.23383.
- Mingyang Liu, Gabriele Farina, and Asuman E. Ozdaglar. 2025a. [UFT: unifying supervised and reinforcement fine-tuning](#). *CoRR*, abs/2505.16984.
- Weiwen Liu, Xu Huang, Xingshan Zeng, Xinlong Hao, Shuai Yu, Dexun Li, Shuai Wang, Weinan Gan, Zhengying Liu, Yuanqing Yu, Zezhong Wang, Yuxian Wang, Wu Ning, Yutai Hou, Bin Wang, Chuhan Wu, Xinzhi Wang, Yong Liu, Yasheng Wang, and 8 others. 2025b. [Toolace: Winning the points of LLM function calling](#). In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.
- Siyuan Lu, Zechuan Wang, Hongxuan Zhang, Qintong Wu, Leilei Gan, Chenyi Zhuang, Jinjie Gu, and Tao Lin. 2025. [Don't just fine-tune the agent, tune the environment](#). *CoRR*, abs/2510.10197.
- Lu Ma, Hao Liang, Meiyi Qiang, Lexiang Tang, Xiaochen Ma, Zhen Hao Wong, Junbo Niu, Chengyu Shen, Runming He, Yanhao Li, and 1 others. 2025. Learning what reinforcement learning can't: Interleaved online fine-tuning for hardest questions. *arXiv preprint arXiv:2506.07527*.
- Xinji Mai, Haotian Xu, Zhong-Zhi Li, Weinong Wang, Jian Hu, Yingying Zhang, Wenqiang Zhang, and 1 others. 2025. Agent rl scaling law: Agent rl with spontaneous code execution for mathematical problem solving. *arXiv preprint arXiv:2505.07773*.
- Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ramakanth Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, Edouard Grave, Yann LeCun, and Thomas Scialom. 2023. [Augmented language models: a survey](#). *Trans. Mach. Learn. Res.*, 2023.
- Shishir G. Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. 2025. [The berkeley function calling leaderboard \(BFCL\): from tool use to agentic evaluation of large language models](#). In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*. OpenReview.net.
- Akshara Prabhakar, Zuxin Liu, Ming Zhu, Jianguo Zhang, Tulika Awalganekar, Shiyu Wang, Zhiwei Liu, Haolin Chen, Thai Hoang, Juan Carlos Niebles, Shelby Heinecke, Weiran Yao, Huan Wang, Silvio Savarese, and Caiming Xiong. 2025a. [Apigen-mt: Agentic pipeline for multi-turn data generation via simulated agent-human interplay](#). *CoRR*, abs/2504.03601.
- Akshara Prabhakar, Zuxin Liu, Ming Zhu, Jianguo Zhang, Tulika Awalganekar, Shiyu Wang, Zhiwei Liu, Haolin Chen, Thai Hoang, Juan Carlos Niebles, and 1 others. 2025b. [Apigen-mt: Agentic pipeline for multi-turn data generation via simulated agent-human interplay](#). *arXiv preprint arXiv:2504.03601*.
- Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiushi Chen, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. 2025. [Toolrl: Reward is all tool learning needs](#). *CoRR*, abs/2504.13958.
- Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024. [Toolllm: Facilitating large language models to master 16000+ real-world apis](#). In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.
- Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-Rong Wen. 2025. [Tool learning with large language models: a survey](#). *Frontiers Comput. Sci.*, 19(8):198343.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. [Deepseekmath: Pushing the limits of mathematical reasoning in open language models](#). *CoRR*, abs/2402.03300.
- Saksham Sahai Srivastava and Vaneet Aggarwal. 2025. [A technical survey of reinforcement learning techniques for large language models](#). *CoRR*, abs/2507.04136.
- Weiting Tan, Xinghua Qu, Ming Tu, Meng Ge, Andy T. Liu, Philipp Koehn, and Lu Lu. 2025. [Process-supervised reinforcement learning for interactive multimodal tool-use agents](#). *CoRR*, abs/2509.14480.
- Qwen Team and 1 others. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2(3).
- Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. 2024. [Appworld: A controllable world of apps and people for benchmarking interactive coding agents](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 16022–16076. Association for Computational Linguistics.
- Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, and 1 others. 2025. [Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning](#). *arXiv preprint arXiv:2504.20073*.
- Junde Wu, Jiayuan Zhu, Yuyuan Liu, Min Xu, and Yueming Jin. 2025. [Agentic reasoning: A streamlined framework for enhancing llm reasoning with agentic tools](#). *arXiv preprint arXiv:2502.04644*.

Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and Yu Su. 2024. Travelplanner: A benchmark for real-world planning with language agents. *arXiv preprint arXiv:2402.01622*.

Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru Li, Xiaosen Zheng, Zejun Ma, and Bo An. 2025. [Simpletir: End-to-end reinforcement learning for multi-turn tool-integrated reasoning](#). *CoRR*, abs/2509.02479.

Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. 2025. [Learning to reason under off-policy guidance](#). *CoRR*, abs/2504.14945.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 40 others. 2025. [Qwen3 technical report](#). *CoRR*, abs/2505.09388.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. 2024. [\$\tau\$ -bench: A benchmark for tool-agent-user interaction in real-world domains](#). *CoRR*, abs/2406.12045.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan Cao. 2023. [React: Synergizing reasoning and acting in language models](#). In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Runzhe Zhan, Yafu Li, Zhi Wang, Xiaoye Qu, Dongrui Liu, Jing Shao, Derek F. Wong, and Yu Cheng. 2025. [Exgrpo: Learning to reason from experience](#). *CoRR*, abs/2510.02245.

Feng Zhang, Zezhong Tan, Xinhong Ma, Ziqiang Dong, Xi Leng, Jianfei Zhao, Xin Sun, and Yang Yang. 2025a. [Adhint: Adaptive hints with difficulty priors for reinforcement learning](#). *arXiv preprint arXiv:2512.13095*.

Guibin Zhang, Hejia Geng, Xiaohang Yu, Zhenfei Yin, Zaibin Zhang, Zelin Tan, Heng Zhou, Zhongzhi Li, Xiangyuan Xue, Yijiang Li, and 1 others. 2025b. [The landscape of agentic reinforcement learning for llms: A survey](#). *arXiv preprint arXiv:2509.02547*.

Kaiyi Zhang, Ang Lv, Jinpeng Li, Yongbo Wang, Feng Wang, Haoyuan Hu, and Rui Yan. 2025c. [Stephint: Multi-level stepwise hints enhance reinforcement learning to reason](#). *arXiv preprint arXiv:2507.02841*.

Shaokun Zhang, Yi Dong, Jieyu Zhang, Jan Kautz, Bryan Catanzaro, Andrew Tao, Qingyun Wu, Zhiding Yu, and Guilin Liu. 2025d. [Nemotron-research-tool-n1: Exploring tool-using language models with reinforced reasoning](#). *CoRR*, abs/2505.00024.

A Analysis

The detailed classification of model output structures:

- **Healthy Tool Call:** Model output contains a valid tool invocation: both opening `<tool_call>` and closing `</tool_call>` tags are present and correctly matched. The internal action is valid, the structure conforms to the expected JSON schema, and no stray `<|im_end|>` tokens or partial tags appear within the invocation.
- **Healthy Response:** Model output does **not** include any tool-call tags (`<tool_call>`, `</tool_call>`) but ends with a valid end-of-turn token `<|im_end|>`. The content is a coherent natural language response without extraneous or misused tool tags.
- **Text Polluted:** Outputs with partially misused tool-call tags, such as missing closing tags, adding `<think>`, embedded `<|im_end|>` inside a tool invocation, or extra tags appended to normal text. Indicates structural corruption where the model mixes dialogue text with tool invocation tags incorrectly, potentially causing protocol violations or parsing errors.
- **Collapsed:** Outputs consisting solely of the end-of-turn token `<|im_end|>`, or cases where all tool invocations collapse to just `<|im_end|>` without any semantic content. Reflects severe RL failure, where the model short-circuits the interaction to maximize reward or avoid penalties, producing no meaningful action or response.

The more analysis results are in Figure 6 and Figure 7.

B Training Details

During RL training for Qwen3, we prepend `<think>\n\n</think>\n\n` before each action to disable the thinking mode. During gradient updates, this prefix is removed, and only the generated content is used for optimization.

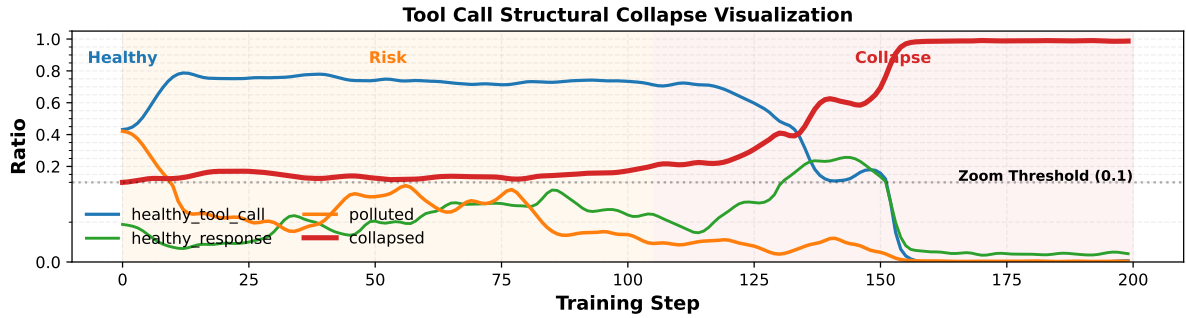


Figure 6 The changes of token appearing frequency in the SFT then RL training process of model Qwen3-1.7B.

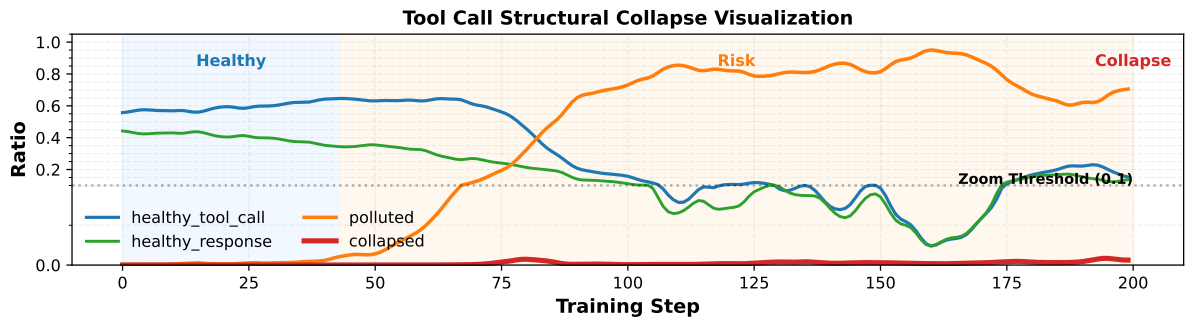


Figure 7 The changes of token appearing frequency in the RL training process of model Qwen3-1.7B.

Type	Example
Healthy Tool Call	<code><tool_call>{"name": "search", "arguments": {...}}</tool_call>< im_end ></code>
Healthy Response	The answer can be derived by first querying the database...< im_end >
Polluted Output	The result is ...<tool_call>< im_end >
Collapsed Output	<tool_call>< im_end >

Table 3 Examples of different output types observed during multi-turn tool-use training.

Parameter	SFT
Epochs	3
Train batch size	128
Learning rate	1e-5
Max sequence length	8192

Table 4 Supervised Fine-Tuning (SFT) configuration.

The SFT and RL parameter configurations are summarized in the Table 4 and Table 5. The experimental setups for the different supervisory interventions are as follows: for off-policy supervision, 7 trajectories are sampled from the model and 1 trajectory uses the ground-truth label; for hint-based guidance, 6 trajectories are generated normally and 2 are conditioned on hints. For interleaved training, the RL rounds for Qwen2.5-1.5B-

Parameter	RL
Total training steps	300
Batch size	60
PPO mini-batch size	20
Learning rate	1e-6
Max turns per trajectory	30
KL loss coefficient	0.0001

Table 5 Reinforcement Learning (RL) configuration.

Instruct are 50, 100, and 150, while for Qwen3-1.7B, they are 50, 50, and 100.

We follow the BFCL reward design in all the reinforcement learning with two evaluation types: **state-based** and **response-based** evaluation. The reward is 1 if both state-based evaluation and

response-based evaluation is 1, and it will be 0 if not. State-based evaluation compares the backend state after all function calls with the ground-truth final state, focusing on tasks that modify internal states. Response-based evaluation compares the model’s execution path with the minimal viable ground-truth function-call path, enabling evaluation of read-only tasks such as querying stock or weather information.

In RPS, we use gpt-5-mini as the auxiliary analyzer to generate textual reflections. The prompt and output example are in Appendix F.

C The Analysis of Qwen3 Training

We believe that the performance collapse observed in Qwen3 after the SFT-then-RL pipeline is primarily caused by inconsistencies in prompt formatting. Specifically, Qwen3 is designed to operate in a thinking mode, where the model is expected to generate an explicit reasoning segment. When reasoning output is disabled, the prompt must explicitly include a placeholder such as `<think>\n\n</think>`. However, during the SFT stage, the training data contains only direct tool-call outputs without this reasoning wrapper. If, during the subsequent RL stage, the prompt is modified to include the thinking-related tokens while the supervised data does not reflect this structure, a mismatch is introduced between the SFT distribution and the RL sampling format. This inconsistency can destabilize training, leading to structural drift and eventual performance collapse. The results in Figure 8 demonstrate this point.

D Detailed Evaluation

Since evaluation is conducted using the final checkpoint, some models may demonstrate strong performance during intermediate training stages but fail to retain these gains at convergence. We report the evaluation results in the training process as follows. We can find the best performance during the training process. The results of Qwen2.5 is shown in Figure 9, Figure 10 and Figure 11. The results of Qwen3 is shown in Figure 12, Figure 13 and Figure 14.

E Training Dynamic

The Training dynamics of Qwen2.5 are shown in Figure 5 and Qwen3 are shown in Figure 15.



Figure 8 The training reward with or without thinking token

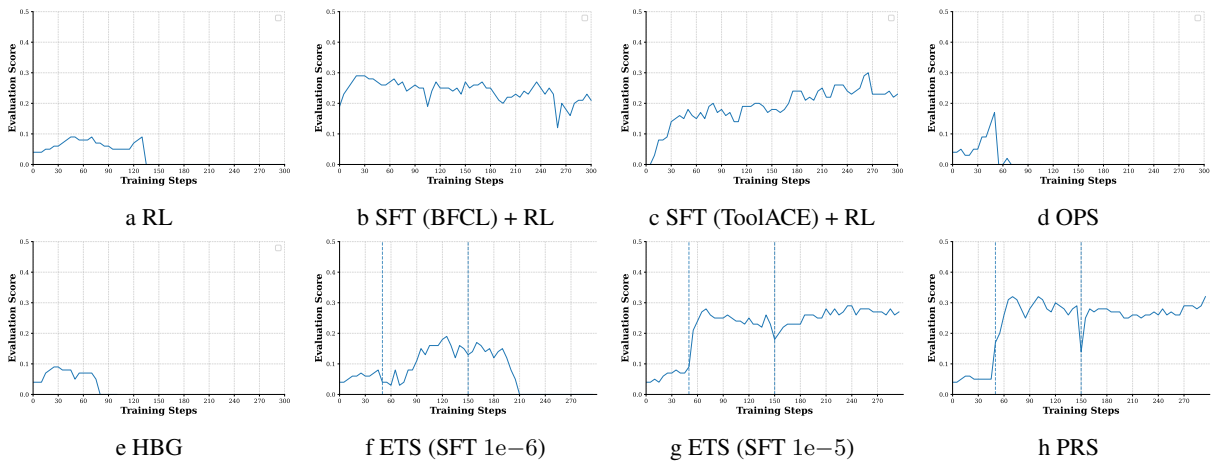


Figure 9 Evaluation results of scenario *Base* under different supervisory signals of Qwen2.5-1.5B-Instruct.

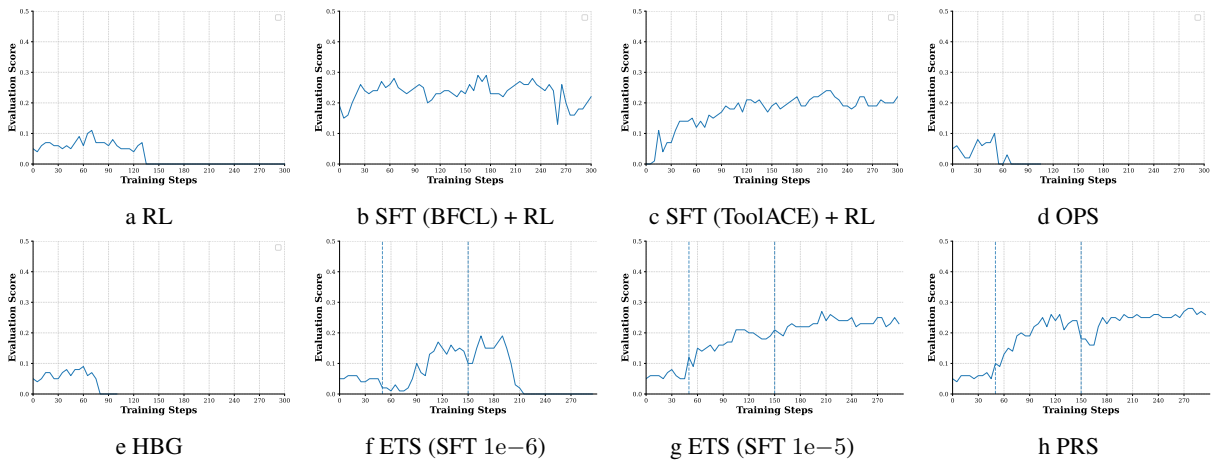


Figure 10 Evaluation results of scenario *Miss Func* under different supervisory signals of Qwen2.5-1.5B-Instruct.

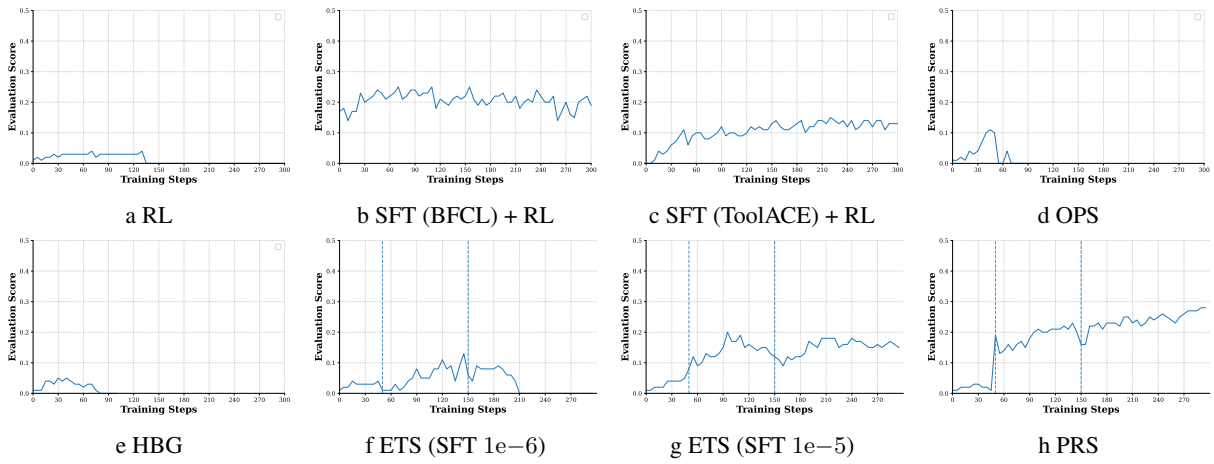


Figure 11 Evaluation results of scenario *Miss Param* under different supervisory signals of Qwen2.5-1.5B-Instruct.

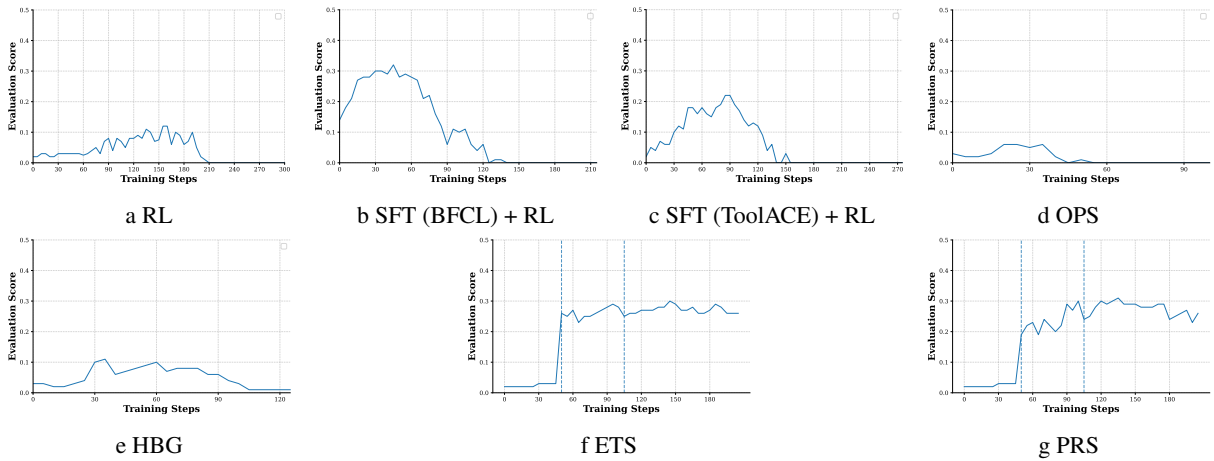


Figure 12 Evaluation results of scenario *Base* under different supervisory signals of Qwen3-1.7B.

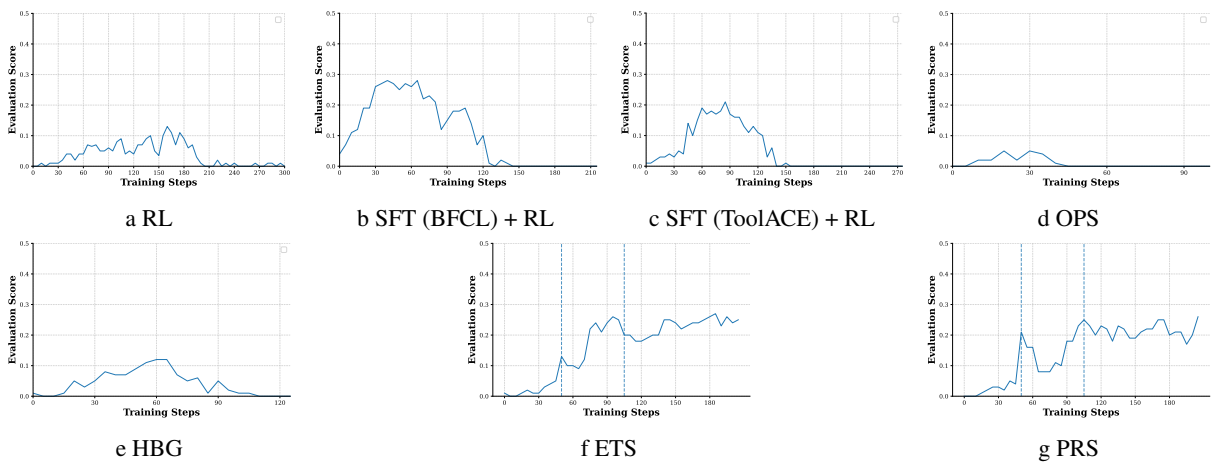


Figure 13 Evaluation results of scenario *Miss Func* under different supervisory signals of Qwen3-1.7B.

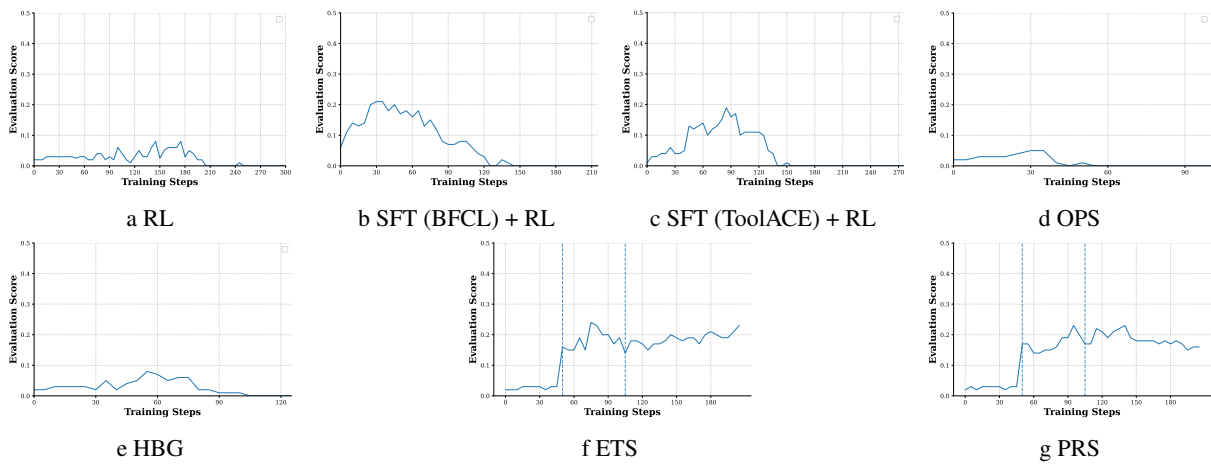


Figure 14 Evaluation results of scenario *Miss Param* under different supervisory signals of Qwen3-1.7B.

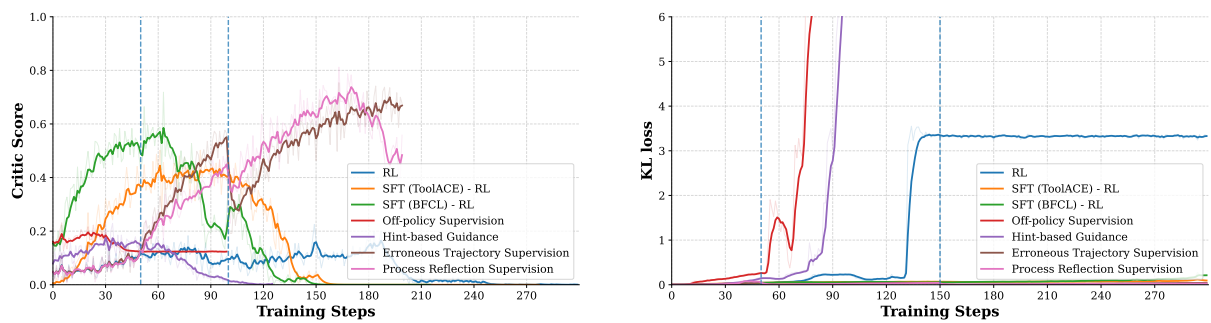


Figure 15 The training dynamics during the training process on the BFCL-V3 dataset of Qwen3-1.7B.

F Prompt and Example

The prompt of RPS is in Prompt 1 and the example is in Example 2.

You are an expert in intelligent agent tool-use analysis.

Below is the erroneous interaction between a model and tools (including the ground-truth tool-use sequence for reference). Your task is to provide a **comprehensive, layered, and actionable analysis**, and **generalize to similar scenarios** to create augmentation data. **Note:** You should write in natural language (not structured JSON), but every correct tool call must strictly follow the raw `<tool_call> ... </tool_call>` format.

Inputs to be filled by caller:

- Available Tools: {available_tools}
- Model Interaction Log (Error Turn): {formatted_summary}

Your required outputs:

- Core Error Analysis (must be first)**
 - Briefly state the root cause of the error (1-2 lines)
 - Provide 2-4 pieces of evidence from the interaction log to support your conclusion
 - Provide one immediate and actionable fix recommendation
- Generalization: Generate 3-5 similar scenarios**

For each scenario, describe in natural paragraphs:

 - One-sentence scenario description (what & why it occurs)
 - The user request and necessary context
 - A typical model mistake that might happen
 - **The correct tool call sequence**
(each step must use the raw `<tool_call>\n{...}\n</tool_call>` block, containing exactly one JSON object with only `name` and `arguments`)
 - What reasoning procedures can be borrowed from the current failed case
 - What aspects differ and require different handling
 - A brief validation strategy for confirming the fix (auto or manual)

The scenarios must cover different error categories:

 - Missing parameters
 - Wrong tool selection
 - Output mismatch
 - Timeout/failure
 - Ambiguous request or multi-intent
- Provide one scenario that is similar but requires a different solution**
 - Explain the similarity (shared reasoning or decision features)
 - Explain clearly why different tool(s) are required
 - Provide correct tool calls in the same `<tool_call>` format
 - Provide rationale
- Output style and quality requirements**
 - Natural language, clear paragraph structure
 - No JSON schemas, no markdown code fences, no meta-explanations
 - Tool calls must be valid JSON and executable
 - Provide 1-2 acceptance criteria per scenario for filtering low-quality synthetic data

5. **Example (format reference only, do not repeat unless needed)**:

```
<tool_call>
{"name": "search_database", "arguments": {"query": "latest model evaluation", "top_k": 5}}
</tool_call>
```

Please begin by analyzing the root cause of the provided error turn, then generalize to 35 similar scenarios (including at least 1 scenario that requires different handling).

Prompt 1 The RPS prompt

Turn 1

1) User intent understanding, correct behavior, and success points

- Intent: Price-check then book a business-class SFLA flight for Robert Trenton on 2024-11-25 using travel card card_3487 (access code 1293), and obtain an invoice to verify charges.
- Correct behavior: (a) get flight cost to confirm price, (b) book the flight with provided card and access token, (c) retrieve the invoice for verification.
- Success points: All three tools were called in the correct order. Responses show cost \$400, booking_id 3426812 and transaction_id 45451592, and retrieve_invoice returned a consistent invoice with cost and booking details. This aligns with the users intent and ground truth.

2) Mistake classification and root cause analysis

- Mistakes: None meaningful. Actions, arguments, and sequence match the ground-truth sequence exactly.
- Root cause: N/A.

3) Improvement plan (corrected reasoning and ideal tool call sequence)

- Reasoning: Confirm price before booking to avoid unexpected charges; then book using provided card and access token; finally retrieve the invoice to verify charges and record transaction details.

- Ideal tool call sequence (matches ground truth and log):

```
<tool_call>
```

```
{"name": "get_flight_cost", "arguments": {"travel_from": "SFO", "travel_to": "LAX", "travel_date": "2024-11-25", "travel_class": "business"}}
```

```
</tool_call>
```

```
<tool_call>
```

```
{"name": "book_flight", "arguments": {"access_token": "1293", "card_id": "card_3487", "travel_date": "2024-11-25", "travel_from": "SFO", "travel_to": "LAX", "travel_class": "business"}}
```

```
</tool_call>
```

```
<tool_call>
```

```
{"name": "retrieve_invoice", "arguments": {"access_token": "1293", "booking_id": "3426812"}}
```

```
</tool_call>
```

- Alignment: This is identical to the ground truth sequence and satisfies the users requirement to book and verify the invoice.

Turn 2

1) User intent understanding, correct behavior, and success points

- Intent: Post a tweet on Robert Trentons travel account: "Loved my flight journey!" with hashtag #TravelDiaries, then retweet it from his travel account to maximize visibility.
- Correct behavior: Before calling post_tweet/retweet, ensure the Twitter account is authenticated (credentials or an authenticated session) and that the correct account is targeted for posting/retweeting. If credentials or an authenticated account identifier are missing, request them rather than calling tools.

2) Mistake classification (fine-grained) and root cause analysis

- Mistake classification:

- Missing Parameter / Precondition Check Failure: The model attempted post_tweet without confirming authentication or asking for credentials.
- Incorrect remediation flow: After failed authentication attempt using wrong credentials, the model tried retweeting (still unauthenticated), producing further errors.

- Root cause:

- The agent did not treat authentication as a required precondition; it attempted actions without validating credentials or asking for clarifying information (username, which account to use).
- It also attempted to remediate by guessing credentials ("user"/"pass") rather than requesting valid credentials from the user.

3) Improvement plan (corrected reasoning and ideal tool call sequence)

- Corrected reasoning: When a user requests an action that requires authentication but does not provide credentials or indicate an authenticated session, the agent must pause and request the missing authentication information (or ask the user to confirm which authenticated account to use). No tool calls should be made until valid credentials or an authenticated session are provided.

- Action to take now (at this turn): Ask for the necessary authentication details and account identifier. Do not call any tools until the user supplies credentials or instructs a currently-authenticated account to be used.

- No tool calls should be made at this turn. If the user supplies credentials, the correct sequence (ground truth) would be:

```
<tool_call>
{"name": "authenticate_twitter", "arguments": {"username": "john", "password": "john1234"}}
</tool_call>
<tool_call>
{"name": "post_tweet", "arguments": {"content": "Loved my flight journey!"}}
</tool_call>
<tool_call>
{"name": "retweet", "arguments": {}}
```

- How this aligns with ground truth and logic: Ground truth specifies that because of missing parameters, no tool calls should be made; instead, the agent should request credentials. The three-step sequence above is the correct follow-up once credentials are provided (authenticate post retweet).

Turn 3

1) User intent understanding, correct behavior, and success points

- Intent: The user supplied Twitter credentials; authenticate with those credentials, post the tweet, and retweet it.

- Correct behavior: Authenticate using username/password, post the requested tweet, then retweet to amplify it.

- Success points: The agent authenticated successfully (authentication_status: true), posted the tweet ("Loved my flight journey!"), and retweeted successfully. The returned post id (10) and retweet success indicate the actions worked and satisfy the users request.

2) Mistake classification and root cause analysis

- Mistakes: Minor inconsistency only:

- The ground-truth post_tweet call used just content (no explicit tags). In the log the tweet included tags in earlier attempts; the final posted tweet in the log did include tags in earlier step but ground truth shows no tags in the tool call. This is a minor divergence in arguments, not a functional error user asked for hashtag #TravelDiaries, so including the tag is acceptable.

- The ground truth retweet call had empty arguments; the agent used tweet_id in the log. Either form is acceptable depending on API design; using tweet_id is explicit and clear.

- Root cause: Small differences in how the API was used (presence/absence of tags field, different retweet parameter shapes). These do not change outcome.

3) Improvement plan (corrected reasoning and ideal tool call sequence)

- Corrected reasoning: Authenticate with supplied credentials; once authenticated, post the tweet exactly as requested (include hashtag #TravelDiaries if the user specified it), capture the returned tweet id, then retweet that tweet to increase visibility. Confirm success and return post and retweet statuses to user.

- Ideal tool call sequence (matches ground truth intent; any of the two retweet argument styles is acceptable depending on API here we show the ground-truth calls and the explicit tweet-id approach):

```
<tool_call>
{"name": "authenticate_twitter", "arguments": {"username": "john", "password": "john1234"}}
</tool_call>
<tool_call>
{"name": "post_tweet", "arguments": {"content": "Loved my flight journey!"}}
</tool_call>
<tool_call>
{"name": "retweet", "arguments": {}}
```

- If using the explicit tweet id returned by post_tweet (recommended for clarity):

```
<tool_call>
{"name": "retweet", "arguments": {"tweet_id": 10}}
```

- Alignment: This aligns with the ground-truth successful sequence: authenticate post_tweet retweet. The explicit tweet_id variant ensures the retweet targets the correct post.

Summary Recommendations (cross-turn)

- Always validate preconditions: check for required credentials or tokens before calling authentication-protected tools. If missing, request them.

- Follow a predictable pattern for transactional flows: estimate cost confirm booking retrieve invoice.

- When a user supplies credentials, authenticate then perform actions; do not guess credentials.
- When API shapes differ (e.g., retweet accepting {} vs tweet_id), prefer explicit parameters (tweet_id) if available to avoid ambiguity.

Prompt 2 The example of RPS