TRAJECTORY2GYM: DECISION-GROUNDED COMPOSITIONAL INTERACTION SYNTHESIS FOR TOOL-USE AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Successful tool-use trajectories are not necessarily useful supervision. Even successful interactions may underrepresent multi-step dependencies or contain actions that are unjustified by the information visible to the agent at decision time. We identify two coupled requirements for effective trajectory supervision: compositional coverage, so that important observation–action dependencies are represented, and decision grounding, so that each supervised action is supported by the learner-visible interaction history. We introduce Trajectory2Gym, which constructs executable interactions around under-supported compositional structures and derives supervision only from grounded decisions in their realized executions. Structural-support selection allocates the resulting experience, while parent-normalized supervised fine-tuning preserves the semantic interaction as the unit of training weight. This reframes trajectory generation as the deliberate construction of learnable interaction experience, rather than the collection of successful trajectories. Across five Qwen3 scales, Trajectory2Gym improves Tau1, Tau2, and BFCL at every tested scale, increasing the equal-weight three-benchmark mean by 16.9–29.8 percentage points. The gains are strongest in stateful and multi-turn interaction, where later decisions depend on earlier observations and feedback. Our results suggest that scaling tool-use agents depends not only on collecting more successful trajectories, but on constructing interactions whose dependencies are represented and whose decisions are learnable from the agent's visible history.

1 INTRODUCTION

Ask a support agent to cancel two orders and update a third. A competent agent looks up the account; when the email lookup fails, it asks for a name and ZIP code and retries; it reads each order to check it is pending, confirms with the user, and only then cancels and swaps items using catalog-returned identifiers. Each call is simple; what must be learned is their composition. We call such workflows *compositional interactions*: an observation, user update, or state change determines a later decision. Tool-use agents face this composition constantly (Yao et al., 2023; Schick et al., 2023; Patil et al., 2024; Qin et al., 2024a; Qu et al., 2025), interactive benchmarks make it central to task success (Yao et al., 2025; Barres et al., 2026; Patil et al., 2025), and collected or synthesized trajectories are its standard supervision (Qin et al., 2024b; Zeng et al., 2024).

Recent pipelines show that *what* interactions are collected matters, yet the prevailing strategy still filters trajectories by task completion or per-call execution (Liu et al., 2024c; 2025; Prabhakar et al., 2025; Xu et al., 2025c), and fine-tuning directly on logged trajectories can even fall below the initialization (Wu et al., 2026a). Success certifies *that* an episode ended well, not *what* it teaches or *whether* each imitated action was justified by what the agent could see. Two challenges follow. **Challenge 1 (compositional coverage):** trajectories must exercise cross-step dependencies with repeated support, yet in our pool of 34,772 certified candidates, 40.9% contain at most two supervised decisions and only 10.0% contain a certified decision after an authentic error. **Challenge 2 (decision grounding):** each target must follow from decision-time visible information; successful API execution alone cannot establish this. A collector may use an unseen identifier, and in Figure 1 the API accepts a stale two-item exchange as readily as the lamp-only exchange finally requested by the user.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

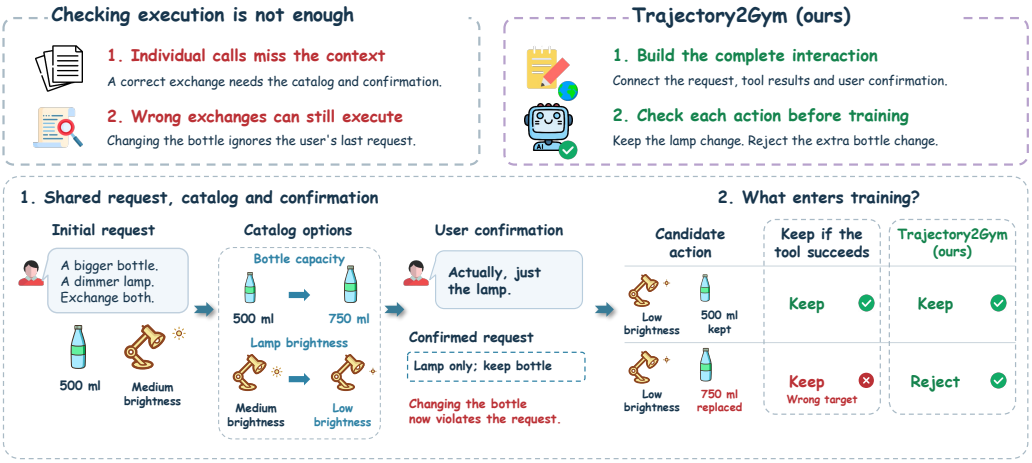


Figure 1: **Compositional interaction requires grounded decisions.** Top: Trajectory2Gym constructs linked interactions (1) and checks each training action against that history (2). Bottom: a Tau2 Retail user requests a larger bottle and a dimmer lamp, then confirms only the lamp. Both rules judge the same two candidate actions. Filtering by execution keeps both because native tool replay accepts both requests. Trajectory2Gym uses catalog evidence and the latest confirmation to keep the lamp exchange and reject the additional bottle exchange. Green checks mark correct selection decisions; the red cross marks erroneous retention. This schematic illustrates supervision selection with paraphrased dialogue and a constructed second candidate.

Our central claim is that **what a trajectory teaches is decided not by its outcome, but by the dependencies it realizes and the decisions its visible history can justify**. Trajectory2Gym operationalizes this claim in three stages (Figure 2). (1) *Compositional interaction synthesis* builds and executes tasks around under-supported interaction paths in native environments, so that information acquisition and recovery are demonstrated (Challenge 1). (2) *Grounded supervision* supervises only decisions certified by visible evidence, execution, and workflow order, keeping authentic errors as recovery context (Challenge 2). (3) *Structural-support learning* selects trajectories to cover capability paths at a per-scale support depth and trains with parent-normalized supervised fine-tuning (SFT). The stages are coupled: synthesis without grounding would teach the collector's shortcuts, grounding without synthesis would certify a pool where recovery and discovery stay rare, and selection without both would only re-weight that pool. With plain supervised fine-tuning from each public initialization, the paired gains measure the complete post-training recipe.

Data built this way lifts every scale: Trajectory2Gym improves Tau1, Tau2, and BFCL on all five Qwen3 models from 1.7B to 30B, raising the three-benchmark mean by 16.9 to 29.8 points. The lift outweighs model size: the post-trained 4B model surpasses the public 30B-A3B initialization on Tau2 (69.17 versus 28.33), and, against public agent-training checkpoints used as references rather than matched controls, the 8B model has the highest three-benchmark mean. Our contributions are:

- We identify compositional coverage and decision grounding as coupled requirements and formalize them as a decision-level supervision mask over agent-visible histories (Section 3).
- We introduce Trajectory2Gym, which synthesizes under-supported interactions in native environments and certifies targets from their executed, agent-visible histories (Sections 4.1 and 4.2).
- We develop structural-support selection with parent-normalized learning and show that the framework improves all three benchmarks at each of five scales, with controlled comparisons of every stage (Sections 4.3 and 5).

2 Related Work

Tool-use benchmarks and training. Single-turn benchmarks test tool and argument selection (Li et al., 2023; Zhuang et al., 2023; Huang et al., 2024; Chen et al., 2024a); later suites add nested, multi-hop, and long-context calls (Basu et al., 2025; Ye et al., 2025; Zhong et al., 2025; Chen et al., 2025a). Interactive benchmarks score whole episodes: multi-turn tool use (Wang et al., 2024b; Liu et al., 2024b; Lu et al., 2025; Guo et al., 2024), τ-bench customer service (Shi et al., 2026), web, computer, and workplace tasks (Zhou et al., 2024; Xie et al., 2024; Mialon et al., 2024; Trivedi et al.,

2024; Xu et al., 2025a; Li et al., 2026), and MCP ecosystems (Luo et al., 2025; Mo et al., 2025; Wang et al., 2026b; Wu et al., 2026b). Our focus is on representing the cross-step dependencies that an executed trajectory actually exercises. Training uses synthesized demonstrations (Parisi et al., 2022; Tang et al., 2023; Yang et al., 2023), cross-environment action traces (Chen et al., 2023; Yin et al., 2024; Chen et al., 2024b; Zhang et al., 2025; Song et al., 2024a; Xi et al., 2025), function-calling specialization (Lin et al., 2024; Wang et al., 2025a), and exploration or reinforcement learning with execution rewards (Song et al., 2024b; Putta et al., 2024; Qian et al., 2025; Feng et al., 2026; Li et al., 2025a; Zhang et al., 2026b; Xi et al., 2026; Zhang et al., 2026e; Team et al., 2025). Outcome-filtered training commonly treats successful trajectories as imitable units, while failure-aware methods add step-level signals (Wang et al., 2024a; Xiong et al., 2024). Our focus is target eligibility under the agent-visible history.

Trajectory synthesis and curation. Trajectory construction is a data-design problem: verified executable synthesis produces function-calling data at scale (Yang et al., 2025), multi-turn extensions add dialogues, tool graphs, and simulated users (Chen et al., 2025b; Yin et al., 2025; Wang et al., 2025b), and web and GUI agents mine tutorials, indirect knowledge, reverse task synthesis, and exploration (Xu et al., 2025b; Ou et al., 2024; Sun et al., 2025; Pahuja et al., 2025). Capability-oriented pipelines target changing or infeasible intents (Wang et al., 2026c), active visual acquisition (Zhang et al., 2026d), or compositional tool graphs (Lu et al., 2023; Shen et al., 2024); others scale environments (Hu et al., 2025; Song et al., 2026a; Tu et al., 2026; Fang et al., 2026; Chen et al., 2026; Wang et al., 2026a; Tian et al., 2026), rebuild executable workspaces from logged trajectories and re-solve them under hidden tests, or standardize formats (Song et al., 2026b). Executable tool-use synthesis often verifies outcomes or calls. For software-engineering agents, Patches-to-Trajectories curates prefix-grounded, efficient steps using privileged reference patches (Ma et al., 2026). Curation is complementary: data selection favors quality over volume (Zhou et al., 2023; Liu et al., 2024a), TopoCurate selects by interaction topology (Yang et al., 2026), and imitation learning separates privileged demonstrator knowledge from student observations (Ross et al., 2011; Chen et al., 2020). Trajectory2Gym combines deficit-guided synthesis in native tool-use environments, policy-visible target eligibility, and structural-support allocation across learner scales.

3 Problem Formulation

Let $\mathcal{E}$ be a tool-use environment and V the projection of an interaction history onto what the agent can see: we take the perspective of a developer who records the complete log during collection but trains a student that acts only through the native interface. Given a request x and tool schema $\mathcal{T}_t$, the policy acts from

$$c_t = \big(x, \mathcal{T}_t, V(H_{<t})\big), \qquad a_t \sim \pi_\theta(\cdot \mid c_t), \tag{1}$$

where $H_{<t}$ is the history preceding decision t; the environment executes a_t, returns an observation, and updates its state, and a completed trajectory τ receives a terminal outcome $R_{\mathcal{E}}(\tau) \in \{0, 1\}$ from the native checker.

For learner scale s we construct a supervised trajectory set

$$\mathcal{D}_s = \{(\tau_i, \mathbf{m}_i)\}_{i=1}^{N_s}, \qquad \mathbf{m}_i \in \{0, 1\}^{T_i}, \tag{2}$$

where T_i is the number of assistant decisions in τ_i and $m_{it} = 1$ marks decision t as a supervised target; the full visible interaction remains context, and the mask only selects targets. Two properties are required. *Compositional coverage* (Challenge 1): the training data provide sufficient support for the observation–action dependencies and multi-step structures the policy should acquire. *Decision grounding* (Challenge 2): each supervised action is supported by the realized interaction and by its decision-time context c_t. Our goal is one procedure that constructs $\mathcal{D}_s$ with both properties for every learner scale (Section 4).

4 Method

Figure 2 connects three stages. (1) Compositional interaction synthesis (Section 4.1, for Challenge 1) turns under-supported paths into executed native interactions; (2) grounded supervision (Section 4.2, for Challenge 2) turns each executed log into a student context and a set of certified targets; and (3) structural-support learning (Section 4.3) decides how often each certified structure enters training and how its rendered examples are weighted (properties proved in Appendix E).

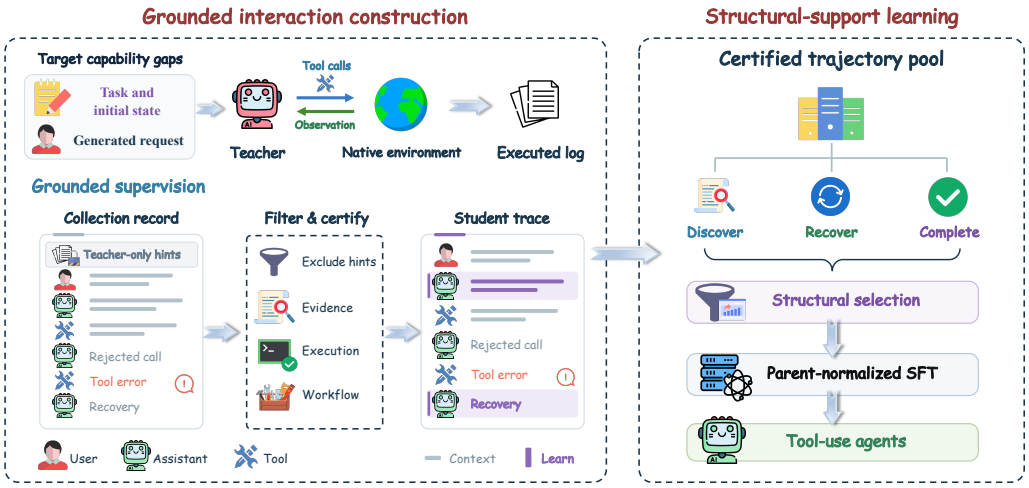


Figure 2: **Overview of Trajectory2Gym.**

4.1 Compositional Interaction Synthesis

A shared capability vocabulary $\mathcal{K}$ covers semantic actions and arguments, observation outcomes, recovery, branching, handoff, termination, and horizon; native tools and observations map to these roles, so one path can be supported by different backends. An ordered path $p = (\kappa_1, \ldots, \kappa_\ell)$ composes these requirements within one interaction. A generator G constructs a request x and compatible initial state s_0 for a target path, and a teacher π_{T} executes the task in the native environment:

$$(x, s_0) \sim G(\cdot \mid p, \mathcal{E}), \qquad \tau \sim P_{\mathcal{E}}(\cdot \mid x, s_0, \pi_{\mathrm{T}}). \tag{3}$$

Each trajectory is assigned one canonical path $p(\tau)$ for support accounting, and the support of p in a selected set $\mathcal{S}$ is

$$n_{\mathcal{S}}(p) = \sum_{\tau \in \mathcal{S}} \mathbf{1}[p(\tau) = p]. \tag{4}$$

The support scenario of Section 1 is the path lookup, error, clarification, retry, order read, confirmation, cancellation, update. We target paths whose support falls below the scale's target (Section 4.3) and condition the generator on them; rare structures need this explicit route, since uniform selection of B out of N candidates misses a path represented once with probability $1 - B/N$ (Proposition 3). Construction establishes the entities, prerequisites, and information sources each path needs (a transaction task may identify the customer but require a lookup for the transaction identifier; a recovery variant creates a discoverable native rejection) and binds required values to visible sources (the request, a user exchange, or a tool result), since only such values can become grounded targets.

Native execution supplies authentic observations, state changes, and rejections; teacher-only guidance stays out of the student trace. A path is certified only when its action and observation dependencies appear in the executed log, and only certified structures update the support inventory and guide later synthesis rounds; a mistaken generator or teacher proposal therefore surfaces as an uncertified structure, not a training target (templates in Appendix B.3).

4.2 Grounded Supervision

Visible context. Grounded supervision turns the execution log into student contexts and eligible targets, checking visibility, local execution, and workflow completion in that order. The decision context c_t in Eq. (1) contains only events exposed through the native agent interface; the full log remains available for certification, but teacher hints and private events never enter the student context.

Target eligibility. We distinguish native execution validity $e_t \in \{0, 1\}$ from decision-time grounding $v_t \in \{0, 1\}$, with local eligibility

$$g_t = e_t \, v_t. \tag{5}$$

For a tool call, $e_t = 1$ requires authenticated non-error execution or native action-only validation; $v_t = 1$ requires the tool to be available to the agent and identifier-like arguments to be supported by preceding visible information (discoverable tools must first appear in a visible retrieval result).

An identifier known only to the collector cannot support a target. When the demonstrated action remains dependent on collector-private information after conditioning on c_t, a policy restricted to c_t faces the additional uncertainty characterized by Proposition 1. A rejected call and its authentic error remain visible *context* without imitation loss, while the grounded recovery that follows becomes a target (Figure 5 in Appendix G.2).

Workflow completion. Let R_τ be native terminal success, q_τ the task's ordered requirements, and A_τ the canonicalized sequence of relevant user operations and locally eligible assistant actions, with roles, arguments, and order preserved. The final supervision mask is

$$m_t = R_\tau \, \mathbf{1}[q_\tau \preceq A_\tau] \, g_t, \tag{6}$$

where $\preceq$ denotes ordered subsequence inclusion; other actions may occur between required operations, and $q_\tau = \varnothing$ when the native checker fully specifies the task. In the constructed contrast of Figure 1, both calls are executable; v_t checks visible tool and argument provenance, not the later user's task scope. When encoded by the native task, R_τ and q_τ check that scope and the required order. Rejected calls are masked but their authentic errors remain visible context. Trajectories with $\sum_t m_t > 0$ enter the eligible pool; role-specific checks are in Appendix C.

4.3 Structural-Support Learning

Structural selection. We assemble $\mathcal{D}_s$ from the certified pool to preserve task breadth, cover capability paths, and give repeated support to under-represented structures such as discovery, recovery, and completion. For scale s, let d_s be the target number of distinct trajectories per feasible canonical path; it grows with learner scale (1, 2, 3, 4, and 8 from 1.7B to 30B-A3B), so larger learners see more realizations of each path per epoch. The remaining deficit of a selected set $\mathcal{S}$ is

$$\delta_s(p; \mathcal{S}) = \big[d_s - n_{\mathcal{S}}(p)\big]_+. \tag{7}$$

Selection starts from a fixed certified core, adds one completing exemplar per training task, closes uncovered atoms and paths, and then adds trajectories that reduce these deficits. Under a pure cardinality budget, this deficit stage exactly maximizes capped support $\sum_p \min(n_{\mathcal{S}}(p), d_s)$; the complete selector also has task and per-task constraints (Proposition 2; Appendix D). Without the cap, the objective is modular and indifferent to which paths are covered.

Parent-normalized SFT. Selected trajectories are rendered through native prompts, roles, tool schemas, and visible observations. One semantic trajectory may yield several training examples; under per-example averaging a parent's weight is proportional to its number of rendered children (Proposition 4), so long multi-turn interactions would dominate the loss. We therefore define each trajectory's contribution at the parent level. With $(h_u, y_u)_{u=1}^{N_\tau}$ the certified next-token pairs of τ counted across rendered examples,

$$\ell_\theta(\tau) = -\frac{1}{N_\tau} \sum_{u=1}^{N_\tau} \log \pi_\theta(y_u \mid h_u), \qquad \mathcal{L}_{\mathrm{SFT}}(\theta) = \frac{1}{|\mathcal{D}_s|} \sum_{\tau \in \mathcal{D}_s} \ell_\theta(\tau). \tag{8}$$

Each parent thus receives equal weight, non-target visible messages remain context, and regrouping targets into different children leaves the objective unchanged (Appendix D.3).

For illustration, consider a corpus of two interactions that render into three examples and one example, respectively. Row averaging assigns them total weights of $3/4$ and $1/4$, whereas parent normalization assigns $1/2$ to each. Each parent's weight is distributed across its examples in proportion to their certified target-token counts, keeping semantic training mass independent of rendering multiplicity.

5 Experiments

We ask whether data built for compositional coverage and decision grounding improves tool use at every scale with SFT alone (**RQ1**; Section 5.2), how the resulting models compare with public checkpoints and where the gains concentrate (**RQ2**, **RQ3**; Sections 5.2 and 5.3), and which stages contribute under matched ablations (**RQ4**; Section 5.4); Sections 5.5 and 5.6 add a depth sensitivity test and a same-task case study.

Table 1: **Tau1 and Tau2 results at all five scales** (Pass@1, %; higher is better). Blue rows are Trajectory2Gym-trained; Δ is the paired gain in Mean, the equal average of the two benchmark averages. Tau1 Airline is a seen-task, transductive setting (Appendix A). Agent-training checkpoints are re-evaluated on the same complete suites: Simia (Li et al., 2025b), MUA-RL (Zhao et al., 2025), LoopTool (Zhang et al., 2026a), EnvFactory (Xu et al., 2026), and MidTool-RL (Jiang et al., 2026) (all evaluated checkpoints in Appendix F.1); closed-source rows are reported references.

Model	Tau1			Tau2				**Mean**	Δ
	Retail	Airline	Avg.	Retail	Airline	Telecom	Avg.		
Closed-source references									
GPT-5.2 (high)	75.44	63.00	69.22	81.58	83.00	89.69	84.76	76.99	
Claude-4.5-Opus (high)	79.39	67.50	73.45	79.61	84.00	92.32	85.31	79.38	
Re-evaluated agent-training checkpoints									
Simia Tau-SFT (Qwen3-8B)	45.22	28.00	36.61	62.50	50.00	15.00	42.50	39.55	
MUA-RL-8B	51.30	18.00	34.65	55.00	15.00	7.50	25.83	30.24	
MUA-RL-32B	66.09	42.00	54.04	70.00	30.00	35.00	45.00	49.52	
LoopTool-8B	39.13	8.00	23.57	40.00	15.00	17.50	24.17	23.87	
EnvFactory-8B (thinking)	38.26	42.00	40.13	42.50	25.00	12.50	26.67	33.40	
MidTool-RL-8B	41.74	8.00	24.87	57.50	10.00	20.00	29.17	27.02	
Qwen3 initialization and Trajectory2Gym post-training									
Qwen3-1.7B	13.04	4.00	8.52	10.00	20.00	40.00	23.33	15.93	
+ Trajectory2Gym	40.87	28.00	34.43	42.50	20.00	50.00	37.50	**35.97**	+20.04
Qwen3-4B	30.43	18.00	24.22	20.00	15.00	22.50	19.17	21.70	
+ Trajectory2Gym	50.43	38.00	44.22	72.50	50.00	85.00	69.17	**56.70**	+35.00
Qwen3-8B	41.74	14.00	27.87	45.00	10.00	20.00	25.00	26.44	
+ Trajectory2Gym	58.26	44.00	51.13	77.50	50.00	95.00	74.17	**62.65**	+36.21
Qwen3-14B	46.09	18.00	32.04	40.00	20.00	15.00	25.00	28.52	
+ Trajectory2Gym	66.09	50.00	58.04	85.00	75.00	82.50	80.83	**69.44**	+40.92
Qwen3-30B-A3B	54.78	34.00	44.39	57.50	15.00	12.50	28.33	36.36	
+ Trajectory2Gym	69.57	64.00	66.78	87.50	70.00	95.00	84.17	**75.47**	+39.11

5.1 Experimental Setup

We evaluate Trajectory2Gym on Tau1, Tau2, and BFCL with five Qwen3 scales (1.7B, 4B, 8B, 14B, and 30B-A3B). Each model starts from its public initialization and receives one epoch of full-parameter, parent-normalized SFT on its scale-conditioned view (11.6k to 19.9k trajectories; Appendix B.1). Tau1 and Tau2 are scored by native task success with equal-weight domain averaging; BFCL reports its eight categories and the three-group Overall of Gecko. The evidence standard is the paired comparison: each post-trained model is compared with its own initialization under the same task population, native interface, decoding, user simulator, and scorer, with each task run once. We additionally evaluate public agent-training checkpoints from five families, trained through trajectory synthesis, supervised learning, or interactive learning, on the complete Tau1, Tau2, and BFCL suites with the same task populations and scorers and each model's native prompting and tool-call format (Appendix F.1); the tables below show one 8B-scale checkpoint from each of five families, plus MUA-RL-32B. Closed-source systems and ToolACE-2-8B are reported references (Appendix A.3). Neither group is a matched or equal-compute comparison: training data, model families, and context budgets differ, which precludes component-level attribution across methods.

5.2 Stateful Tool Use (RQ1 to RQ3)

Trajectory2Gym raises the Tau1 and Tau2 mean by 20 to 41 points at every scale. All five initializations improve on both benchmarks (Table 1); the paired gain is smallest at 1.7B (20.04) and between 35 and 41 points from 4B upward, and supervision changes what a given capacity delivers: the post-trained 4B model reaches 69.17 on Tau2, against 28.33 for the public 30B-A3B initialization, and the post-trained 30B-A3B model reaches 84.17, within 1.2 points of the reported closed-source references, 84.76 and 85.31 (RQ1). Against re-evaluated agent-training checkpoints, our 8B model reaches 74.17 on Tau2, at least 28 points above every one of them, 14B and 32B ones included, and on Tau1 trails only MUA-RL-32B, which our 14B and 30B-A3B models exceed (RQ2). The gain is also broad: across five domains and five scales, 24 of 25 paired comparisons

Table 2: **BFCL results** (%; higher is better). Overall equally averages the four-category Non-live mean, the three-category Live mean, and Multi-turn Base. Blue rows are Trajectory2Gym-trained; Δ is the paired gain in Overall. Closed-source rows and ToolACE-2-8B follow Gecko Table 3 (Zhang et al., 2026c) without GATS test-time refinement; the other agent-training checkpoints are re-evaluated as in Table 1. Par., Mult., and Irrel. abbreviate the Parallel, Multiple, and Irrelevance categories; a Parallel score of 0.00 is 0/100 correct, not missing data.

Model	Non-live				Live			Multi-turn	**Overall**	Δ
	Simple	Par.	Mult.	Irrel.	Simple	Mult.	Irrel.	Base		
Closed-source references										
GPT-5.5	91.50	90.50	88.50	88.75	73.64	73.79	83.82	60.00	75.63	
Gemini-3.0-Pro	94.50	91.00	94.00	82.50	87.60	80.44	73.19	69.00	79.97	
Agent-training references										
ToolACE-2-8B	88.00	92.50	92.50	95.41	70.93	79.01	84.80	49.00	73.12	
Simia Tau-SFT (Qwen3-8B)	88.50	0.00	92.00	85.83	58.14	38.14	88.69	1.00	43.08	
MUA-RL-8B	94.50	0.00	94.00	81.67	74.42	72.30	84.16	30.00	58.17	
MUA-RL-32B	94.50	66.00	96.00	84.17	84.50	72.87	86.43	51.00	72.48	
LoopTool-8B	97.50	97.00	96.00	81.67	83.72	75.90	74.21	35.00	68.66	
EnvFactory-8B (thinking)	89.50	96.00	96.00	75.00	85.27	76.85	64.48	68.00	77.55	
MidTool-RL-8B	96.00	85.00	93.00	74.17	86.82	73.06	68.55	58.00	73.73	
Qwen3 initialization and Trajectory2Gym post-training										
Qwen3-1.7B	91.00	84.00	93.00	85.83	64.34	61.86	75.34	12.00	55.88	
+ Trajectory2Gym	85.50	79.00	90.00	88.33	53.49	68.12	75.57	48.00	**66.48**	+10.60
Qwen3-4B	93.50	92.00	95.00	86.67	76.74	72.87	78.28	21.00	62.92	
+ Trajectory2Gym	92.50	90.00	97.00	85.83	68.22	74.19	71.27	62.00	**74.85**	+11.93
Qwen3-8B	95.50	94.00	97.00	83.33	73.64	73.81	72.85	36.00	67.30	
+ Trajectory2Gym	95.00	94.00	97.00	89.17	74.42	74.95	72.85	68.00	**78.62**	+11.32
Qwen3-14B	93.00	95.00	93.00	90.00	79.84	76.66	75.34	46.00	72.01	
+ Trajectory2Gym	93.50	94.00	96.00	90.00	73.64	79.32	83.03	67.00	**79.68**	+7.67
Qwen3-30B-A3B	94.00	94.00	97.00	80.83	82.95	74.19	71.72	65.00	77.58	
+ Trajectory2Gym	96.50	94.00	98.00	89.17	69.77	76.09	83.26	72.00	**80.93**	+3.35

improve, with the largest movement in Tau2 Telecom, which rises from 12.50 to 95.00 at 30B-A3B, and the exception, Tau2 Airline at 1.7B, is unchanged at 20.00 (RQ3). Taken together, standard SFT on decision-grounded compositional data, with no reinforcement learning, makes every initialization from 4B upward a stronger stateful agent than any evaluated checkpoint, RL-trained and 32B ones included (Tau mean 56.70 to 75.47 versus at most 49.52); the margin is widest on Tau2, whose multi-step, state-dependent tasks are where compositional coverage and decision grounding are meant to act. These gains reflect the complete recipe, whose training view also changes with scale (Appendix B.1); the matched ablations of Section 5.4 separate its stages.

5.3 Function Calling and Multi-Turn Behavior (RQ2, RQ3)

BFCL gains concentrate in multi-turn tool use. Overall improves at every scale, driven by Multi-turn Base, which rises by 7 to 41 points, consistent with the cross-step dependencies that synthesis targets (RQ3): every post-trained model from 4B upward exceeds the reported GPT-5.5 (60.00) on Multi-turn Base, and our 30B-A3B model posts 80.93 Overall, numerically above the 79.97 reported for Gemini-3.0-Pro under its source protocol. Single-turn categories start high and mostly hold; the one consistent exception is Live Simple, down 6 to 13 points at four of five scales, which we hypothesize reflects a shift toward information gathering on queries that expect one immediate call. Function-calling skill does not transfer to stateful tasks, which also require later actions to stay consistent with accumulated evidence and state: EnvFactory-8B nearly matches our 8B model on BFCL Overall (77.55 versus 78.62) and ties it on Multi-turn Base (68.00), yet trails it by 47.5 points on Tau2 (Table 1). Across all three benchmarks, our 8B model therefore has the highest equal-weight mean of any evaluated checkpoint, 67.97 against at most 57.17 (RQ2).

Table 3: **Matched component ablations on Qwen3-8B** (%). Each pair compares one Trajectory2Gym design choice with its stage-matched alternative. Mean equally averages Tau1, Tau2, and BFCL; Gain is the Trajectory2Gym choice minus its alternative, so positive values favor the proposed component. Matching conditions are given in Appendix F.3.

Setting	Tau1	Tau2	BFCL	Mean	Gain
Compositional interaction synthesis					
Existing candidate pool	51.00	75.83	80.39	69.08	–
+ Targeted expansion	**58.61**	**81.67**	77.61	**72.63**	**+3.55**
Decision-grounded supervision					
Loose success-derived targets	48.13	71.67	75.86	65.22	–
+ Decision grounding	**51.13**	**74.17**	**78.62**	**67.97**	**+2.75**
Structural-support selection					
Random selection	51.13	74.17	**79.40**	68.23	–
+ Structural selection	**57.00**	**75.83**	78.81	**70.55**	**+2.32**
Parent-normalized training mass					
Rendered-row averaging	49.52	52.50	77.79	59.94	–
+ Parent normalization	**51.13**	**74.17**	**78.62**	**67.97**	**+8.04**

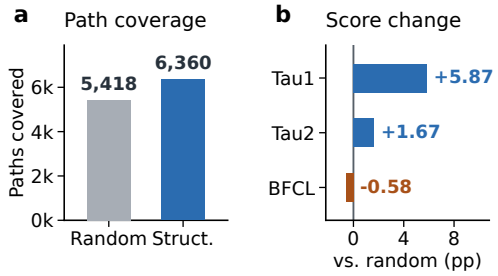


Figure 3: **Structural vs. random selection at a fixed 8B budget** (14,819 parents, equal target tokens). (a) Canonical paths covered. (b) Score change of structural selection (points). Structural selection covers 942 more paths and improves both Tau benchmarks, with BFCL essentially unchanged.

5.4 Ablation Study (RQ4)

All four 8B matched comparisons favor the proposed choices. Each pair in Table 3 changes one design choice at Qwen3-8B and keeps the rest of the procedure matched as closely as possible (Appendix F.3); each arm is a single run, and one Tau2 Airline task moves the three-benchmark mean by 0.56 points, so gains of a few points are directional. All four choices raise the mean in their stage-matched comparisons. Targeted expansion mainly strengthens the stateful Tau tasks, but its pair changes candidate supply without fixing the target-token budget, so it does not isolate targeting at fixed data volume. Grounding improves decision-level supervision, and structural selection covers 942 more canonical paths at the same budget (Figure 3). Parent normalization gives the largest isolated gain (8.04 points): the semantic trajectories, rendered examples, and target masks are unchanged, so this pair isolates how optimization mass is distributed and supports treating a semantic interaction, rather than a rendered row, as the unit of optimization. Because each pair intervenes at a different stage, the gains are not additive.

5.5 Sensitivity Analysis

Table 4: **Sensitivity to structural-support depth at Qwen3-8B** (%). Depth controls the amount of repeated support allocated to certified interaction structures. These control-specific runs use a different training view from the headline 8B checkpoint of Tables 1 and 2.

Depth	Trajectories	Target tokens	Tau1	Tau2	BFCL	Mean
2	13,986	18.02M	55.61	**82.50**	78.74	**72.28**
3	14,819	19.42M	**57.00**	75.83	78.81	70.55
4	15,805	20.61M	54.00	75.00	**79.31**	69.44

More structural support is not automatically better. Performance does not increase monotonically with support depth (Table 4): the three-benchmark mean is highest at the shallowest setting (72.28 versus 70.55 and 69.44) even as trajectories and target tokens continue to grow. The decline

is concentrated in Tau2 (82.50 to 75.00), while BFCL remains comparatively stable. The headline schedule was fixed before these runs, so this is a sensitivity test rather than checkpoint selection; because depth also changes data volume, it does not isolate path support from token count. In short, structural support is an allocation problem rather than a simple data-scaling rule, and we set depth per scale by a predeclared schedule (Appendices F.5 and D).

5.6 Qualitative Analysis

Figure 4 contrasts three 8B agents on a Retail workflow whose two updates must occur in order. Simia has seen the requested address in an order record, yet keeps querying the user profile and transfers to a human: exposed evidence never reaches the decision that needs it. MUA-RL holds the right address but edits the item first, which changes the order status and blocks the address update, an ordering error invisible to per-call checks because the edit itself executes cleanly. Trajectory2Gym confirms both changes, updates the address while the order is pending, then edits the items. These are exactly the behaviors our construction supervises: grounding admits only targets whose arguments trace to visible evidence, and synthesis with workflow-order certification builds and certifies such dependent-state paths. Appendix G gives the traces and six further paired cases.

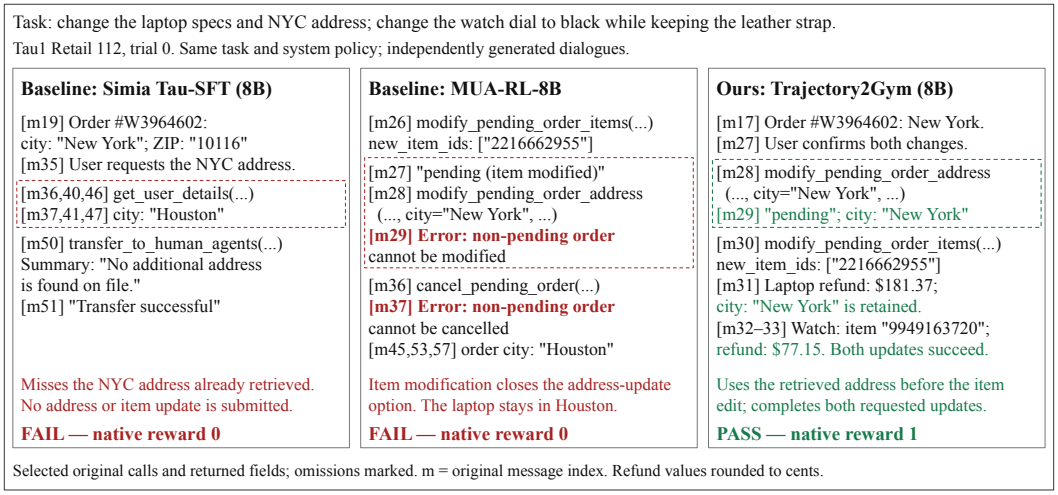


Figure 4: **Same-task comparison of evidence use and action order.** Simia overlooks visible evidence; MUA-RL violates a state-dependent ordering; Trajectory2Gym reuses the evidence and preserves the required dependency (native rewards 0, 0, and 1). Interactions are independently generated and abridged for display; a step-by-step account appears in Appendix G.1.

6 Conclusion

Successful trajectories are the default supervision for tool-use agents, yet success certifies an episode, not what it teaches. This paper has argued that a trajectory's value lies in properties an outcome filter cannot see, compositional coverage and decision grounding, and Trajectory2Gym builds training data around them: it synthesizes the interactions a pool lacks, certifies supervision one decision at a time, and allocates structural support per scale. Across scales from 1.7B to 30B, standard SFT on the constructed interactions improves tool use, especially on multi-turn and stateful tasks. Matched 8B controls give directional evidence for the four design choices, with the clearest isolated gain from parent normalization. The broader lesson is that success should gate supervision, not license it; that authentic errors belong in context, not in targets; and that support is an allocation, not a quantity to maximize. Extending decision-level certification to on-policy rollouts and to web, software, and terminal agents is the natural next step. What a trajectory teaches is decided not by its outcome, but by the dependencies it realizes and the decisions its visible history can justify.

AI USE STATEMENT

We used generative models in two roles. As part of the reported data construction, models proposed training-side task requests and compatible initial states, and a collection teacher produced candidate tool-use trajectories. These proposals were not accepted as supervision on their own: tool calls were executed in native environments, and workflow, policy-visible evidence, and decision-level eligibility checks determined which assistant actions entered supervised fine-tuning. Teacher-only hints and private events were excluded from student inputs. Reported benchmark outcomes came from the pinned native evaluators, including their configured user simulators and natural-language checks, not from manually assigned model judgments.

Generative AI also assisted code drafting and debugging, literature discovery, figure iteration, and manuscript editing. Reported numbers were checked against recorded evaluator outputs, and the authors reviewed AI-assisted prose, citations, and experimental descriptions. The authors take responsibility for the final manuscript and its artifacts.

ETHICS STATEMENT

Interaction synthesis executes real tool calls, so it can mutate backend state, expose private trajectory content, or carry out unsafe actions. The dual-use question is whether the same loop could be pointed at harmful capability paths. We believe the barrier lies with the environment owner rather than the method: synthesis requires an executable, resettable, permissioned environment and its native checkers, and the pipeline assumes rather than replaces that authorization. Training should therefore use resettable, permissioned environments; released traces should remove personal data and credentials; and high-impact actions should require independent authorization. An execution certificate establishes behavioral evidence, not social, legal, or operational acceptability.

REPRODUCIBILITY STATEMENT

The appendix serves as an index to the reported pipeline. Appendix A gives the benchmarks and metrics, the task populations and split hygiene, the comparison protocol with its reference-score sources, and the training configuration. Appendix B gives the training-view sizes per scale, the representation of compositional structure, and the synthesis procedure. Appendix C specifies the policy-visible evidence, the decision and workflow eligibility checks, and the treatment of recovery, context, and repeated actions. Appendix D specifies structural coverage and support depth, experience allocation, and the parent-normalized objective, and Appendix E states and proves the properties of the three stages that the method section relies on. Appendix F gives the unified evaluation of the public agent checkpoints, the matching conditions of the 8B ablations, and the structural-coverage and support-depth diagnostics. Appendix G gives the recorded cases. The repository contains the trajectory schema, synthesis protocols, grounding checks, selector, renderers, training launchers, and split-hygiene audits; release artifacts will include hashes for semantic candidates, selected scale views, rendered training bytes, checkpoints, and task-level evaluation outputs.

REFERENCES

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik R Narasimhan. τ^2-Bench: Evaluating conversational agents in a dual-control environment. In *Forty-third International Conference on Machine Learning*, 2026.

Kinjal Basu, Ibrahim Abdelaziz, Kiran Kate, Mayank Agarwal, Maxwell Crouse, Yara Rizk, Kelsey Bradford, Asim Munawar, Sadhana Kumaravel, Saurabh Goyal, Xin Wang, Luis A. Lastras, and Pavan Kapanipathi. NESTFUL: A benchmark for evaluating LLMs on nested sequences of API calls. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 33538–33547, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1702.

Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Collier, Karthik Narasimhan, and Shunyu Yao. FireAct: Toward language agent fine-tuning. *arXiv preprint arXiv:2310.05915*, 2023.

Chen Chen, Xinlong Hao, Weiwen Liu, Xu Huang, Xingshan Zeng, Shuai Yu, Dexun Li, Shuai Wang, Weinan Gan, Yuefeng Huang, Wulong Liu, Xinzhi Wang, Defu Lian, Baoqun Yin, Yasheng Wang, and Wu Liu. ACEBench: Who wins the match point in tool usage? *arXiv preprint arXiv:2501.12851*, 2025a.

Dian Chen, Brady Zhou, Vladlen Koltun, and Philipp Krähenbühl. Learning by cheating. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura (eds.), *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 66–75. PMLR, 2020.

Mingyang Chen, sunhaoze, Tianpeng Li, Fan Yang, Hao Liang, KeerLu, Bin CUI, Wentao Zhang, Zenan Zhou, and weipeng chen. Facilitating multi-turn function calling for LLMs via compositional instruction tuning. In *The Thirteenth International Conference on Learning Representations*, 2025b.

Zehui Chen, Weihua Du, Wenwei Zhang, Kuikun Liu, Jiangning Liu, Miao Zheng, Jingming Zhuo, Songyang Zhang, Dahua Lin, Kai Chen, and Feng Zhao. T-Eval: Evaluating the tool utilization capability of large language models step by step. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9510–9529, Bangkok, Thailand, August 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.515.

Zehui Chen, Kuikun Liu, Qiuchen Wang, Wenwei Zhang, Jiangning Liu, Dahua Lin, Kai Chen, and Feng Zhao. Agent-FLAN: Designing data and methods of effective agent tuning for large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 9354–9366, Bangkok, Thailand, August 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.557.

Zhaorun Chen, Zhuokai Zhao, Kai Zhang, Bo Liu, Qi Qi, Yifan Wu, Tarun Kalluri, Xuefei Cao, Yuanhao Xiong, Haibo Tong, Huaxiu Yao, Hengduo Li, Jiacheng Zhu, Xian Li, Dawn Song, Bo Li, Jason E Weston, and Dat Huynh. Scaling agent learning via experience synthesis. In *The Fourteenth International Conference on Learning Representations*, 2026.

Runnan Fang, Shihao Cai, Baixuan Li, Jialong Wu, Guangyu Li, Wenbiao Yin, Xinyu Wang, Xiaobin Wang, Liangcai Su, Zhen Zhang, Shibin Wu, Zhengwei Tao, Yong Jiang, Pengjun Xie, Ningyu Zhang, Fei Huang, Wentao Zhang, and Jingren Zhou. Towards general agentic intelligence via environment scaling. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 17610–17621, San Diego, California, United States, 2026. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.872.

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. In *The Fourteenth International Conference on Learning Representations*, 2026.

Zhicheng Guo, Sijie Cheng, Hao Wang, Shihao Liang, Yujia Qin, Peng Li, Zhiyuan Liu, Maosong Sun, and Yang Liu. StableToolBench: Towards stable large-scale benchmarking on tool learning of large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 11143–11156, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.664.

Mengkang Hu, Pu Zhao, Can Xu, Qingfeng Sun, Jian-Guang Lou, Qingwei Lin, Ping Luo, and Saravan Rajmohan. AgentGen: Enhancing planning abilities for large language model based agent via environment and task generation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1*, pp. 496–507. ACM, 2025. doi: 10.1145/3690624.3709321.

Yue Huang, Jiawen Shi, Yuan Li, Chenrui Fan, Siyuan Wu, Qihui Zhang, Yixin Liu, Pan Zhou, Yao Wan, Neil Zhenqiang Gong, and Lichao Sun. MetaTool benchmark for large language models: Deciding whether to use tools and which to use. In *The Twelfth International Conference on Learning Representations*, 2024.

Fengqing Jiang, Yite Wang, Boyi Liu, Zhaoyang Wang, Canwen Xu, Zhewei Yao, Radha Poovendran, and Yuxiong He. MidTool: Mid-training data synthesis for agentic tool use. In *Third Conference on Language Modeling*, 2026.

Junlong Li, Wenshuo Zhao, Jian Zhao, Weihao Zeng, Haoze Wu, Xiaochen Wang, Rui Ge, Yuxuan Cao, Yuzhen Huang, Wei Liu, Junteng Liu, Zhaochen Su, Yiyang Guo, Fan Zhou, Lueyang Zhang, Juan Michelini, Xingyao Wang, Xiang Yue, Shuyan Zhou, Graham Neubig, and Junxian He. The tool decathlon: Benchmarking language agents for diverse, realistic, and long-horizon task execution. In *The Fourteenth International Conference on Learning Representations*, 2026.

Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. API-Bank: A comprehensive benchmark for tool-augmented LLMs. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 3102–3116, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.187.

Xuefeng Li, Haoyang Zou, and Pengfei Liu. ToRL: Scaling tool-integrated RL. *arXiv preprint arXiv:2503.23383*, 2025a.

Yuetai Li, Huseyin A. Inan, Xiang Yue, Wei-Ning Chen, Lukas Wutschitz, Janardhan Kulkarni, Radha Poovendran, Robert Sim, and Saravan Rajmohan. Simulating environments with reasoning models for agent training. *arXiv preprint arXiv:2511.01824*, 2025b.

Qiqiang Lin, Muning Wen, Qiuying Peng, Guanyu Nie, Junwei Liao, Jun Wang, Xiaoyun Mo, Jiamu Zhou, Cheng Cheng, Yin Zhao, Jun Wang, and Weinan Zhang. Hammer: Robust function-calling for on-device language models via function masking. *arXiv preprint arXiv:2410.04587*, 2024.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. In *The Twelfth International Conference on Learning Representations*, 2024a.

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, Duyu Tang, Dandan Tu, Lifeng Shang, Xin Jiang, Ruiming Tang, Defu Lian, Qun Liu, and Enhong Chen. ToolACE: Winning the points of LLM function calling. In *The Thirteenth International Conference on Learning Representations*, 2025.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. AgentBench: Evaluating LLMs as agents. In *The Twelfth International Conference on Learning Representations*, 2024b.

Zuxin Liu, Thai Hoang, Jianguo Zhang, Ming Zhu, Tian Lan, Shirley Kokane, Juntao Tan, Weiran Yao, Zhiwei Liu, Yihao Feng, Rithesh Murthy, Liangwei Yang, Silvio Savarese, Juan Carlos Niebles, Huan Wang, Shelby Heinecke, and Caiming Xiong. APIGen: Automated pipeline for generating verifiable and diverse function-calling datasets. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 54463–54482. Curran Associates, Inc., 2024c. doi: 10.52202/079017-1725.

Jiarui Lu, Thomas Holleis, Yizhe Zhang, Bernhard Aumayer, Feng Nan, Haoping Bai, Shuang Ma, Shen Ma, Mengyu Li, Guoli Yin, Zirui Wang, and Ruoming Pang. ToolSandbox: A stateful, conversational, interactive evaluation benchmark for LLM tool use capabilities. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1160–1183, Albuquerque, New Mexico, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-naacl.65.

Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. Chameleon: Plug-and-play compositional reasoning with large language models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 43447–43478. Curran Associates, Inc., 2023. doi: 10.52202/075280-1882.

Ziyang Luo, Zhiqi Shen, Wenzhuo Yang, Zirui Zhao, Prathyusha Jwalapuram, Amrita Saha, Doyen Sahoo, Silvio Savarese, Caiming Xiong, and Junnan Li. MCP-Universe: Benchmarking large language models with real-world model context protocol servers. *arXiv preprint arXiv:2508.14704*, 2025.

Murong Ma, Tianyu Chen, Yun Lin, Shuai Lu, Qinglin Zhu, Yeyun Gong, Zhiyong Huang, Peng Cheng, Yan Lu, and Jin Song Dong. From patches to trajectories: Privileged process supervision for software-engineering agents. *arXiv preprint arXiv:2605.21996*, 2026.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. GAIA: A benchmark for general AI assistants. In *The Twelfth International Conference on Learning Representations*, 2024.

Guozhao Mo, Wenliang Zhong, Jiawei Chen, Qianhao Yuan, Xuanang Chen, Yaojie Lu, Hongyu Lin, Ben He, Xianpei Han, and Le Sun. LiveMCPBench: Can agents navigate an ocean of MCP tools? *arXiv preprint arXiv:2508.01780*, 2025.

Tianyue Ou, Frank F. Xu, Aman Madaan, Jiarui Liu, Robert Lo, Abishek Sridhar, Sudipta Sengupta, Dan Roth, Graham Neubig, and Shuyan Zhou. Synatra: Turning indirect knowledge into direct demonstrations for digital agents at scale. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 91618–91652. Curran Associates, Inc., 2024. doi: 10.52202/079017-2908.

Vardaan Pahuja, Yadong Lu, Corby Rosset, Boyu Gou, Arindam Mitra, Spencer Whitehead, Yu Su, and Ahmed Hassan Awadallah. Explorer: Scaling exploration-driven web trajectory synthesis for multimodal web agents. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 6300–6323, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.326.

Aaron Parisi, Yao Zhao, and Noah Fiedel. TALM: Tool augmented language models. *arXiv preprint arXiv:2205.12255*, 2022.

Shishir G. Patil, Tianjun Zhang, Xin Wang, and Joseph E. Gonzalez. Gorilla: Large language model connected with massive APIs. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 126544–126565. Curran Associates, Inc., 2024. doi: 10.52202/079017-4020.

Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E. Gonzalez. The berkeley function calling leaderboard (BFCL): From tool use to agentic

evaluation of large language models. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 48371–48392. PMLR, 2025.

Akshara Prabhakar, Zuxin Liu, Ming Zhu, Jianguo Zhang, Tulika Manoj Awalgaonkar, Shiyu Wang, Zhiwei Liu, Haolin Chen, Thai Hoang, Juan Carlos Niebles, Shelby Heinecke, Weiran Yao, Huan Wang, Silvio Savarese, and Caiming Xiong. APIGen-MT: Agentic pipeline for multi-turn data generation via simulated agent-human interplay. In *Advances in Neural Information Processing Systems 38*, pp. 72647–72669. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2025. doi: 10.52202/085713-2184.

Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. Agent q: Advanced reasoning and learning for autonomous AI agents. *arXiv preprint arXiv:2408.07199*, 2024.

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. In *Advances in Neural Information Processing Systems 38*, pp. 116896–116926. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2025. doi: 10.52202/085713-3524.

Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Xuanhe Zhou, Yufei Huang, Chaojun Xiao, Chi Han, Yi Ren Fung, Yusheng Su, Huadong Wang, Cheng Qian, Runchu Tian, Kunlun Zhu, Shihao Liang, Xingyu Shen, Bokai Xu, Zhen Zhang, Yining Ye, Bowen Li, Ziwei Tang, Jing Yi, Yuzhang Zhu, Zhenning Dai, Lan Yan, Xin Cong, Yaxi Lu, Weilin Zhao, Yuxiang Huang, Junxi Yan, Xu Han, Xian Sun, Dahai Li, Jason Phang, Cheng Yang, Tongshuang Wu, Heng Ji, Guoliang Li, Zhiyuan Liu, and Maosong Sun. Tool learning with foundation models. *ACM Computing Surveys*, 57(4):1–40, 2024a. doi: 10.1145/3704435.

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. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *The Twelfth International Conference on Learning Representations*, 2024b.

Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-rong Wen. Tool learning with large language models: a survey. *Frontiers of Computer Science*, 19(8), January 2025. doi: 10.1007/s11704-024-40678-2.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík (eds.), *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635, Fort Lauderdale, FL, USA, 2011. PMLR.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 68539–68551. Curran Associates, Inc., 2023. doi: 10.52202/075280-2997.

Yongliang Shen, Kaitao Song, Xu Tan, Wenqi Zhang, Kan Ren, Siyu Yuan, Weiming Lu, Dongsheng Li, and Yueting Zhuang. TaskBench: Benchmarking large language models for task automation. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 4540–4574. Curran Associates, Inc., 2024. doi: 10.52202/079017-0148.

Quan Shi, Alexandra Zytek, Pedram Razavi, Karthik R Narasimhan, and Victor Barres. τ-Knowledge: Evaluating conversational agents over unstructured knowledge. In *Forty-third International Conference on Machine Learning*, 2026.

Xiaoshuai Song, Haofei Chang, Guanting Dong, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. EnvScaler: Scaling tool-interactive environments for LLM agent via programmatic synthesis. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 8326–8357, San Diego, California, United States, 2026a. Association for Computational Linguistics. ISBN 979-8-89176-395-1. doi: 10.18653/v1/2026.findings-acl.407.

Yifan Song, Weimin Xiong, Xiutian Zhao, Dawei Zhu, Wenhao Wu, Ke Wang, Cheng Li, Wei Peng, and Sujian Li. AgentBank: Towards generalized LLM agents via fine-tuning on 50000+ interaction trajectories. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 2124–2141, Miami, Florida, USA, November 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.116.

Yifan Song, Da Yin, Xiang Yue, Jie Huang, Sujian Li, and Bill Yuchen Lin. Trial and error: Exploration-based trajectory optimization of LLM agents. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7584–7600, Bangkok, Thailand, 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.409.

Yueqi Song, Ketan Ramaneti, Zaid Sheikh, Ziru Chen, Boyu Gou, Tianbao Xie, Yiheng Xu, Danyang Zhang, Apurva Gandhi, Fan Yang, Joseph Liu, Tianyue Ou, Zhihao Yuan, Frank F. Xu, Shuyan Zhou, Xingyao Wang, Xiang Yue, Tao Yu, Huan Sun, Yu Su, and Graham Neubig. Agent data protocol: Unifying datasets for diverse, effective fine-tuning of LLM agents. In *The Fourteenth International Conference on Learning Representations*, 2026b.

Qiushi Sun, Kanzhi Cheng, Zichen Ding, Chuanyang Jin, Yian Wang, Fangzhi Xu, Zhenyu Wu, Chengyou Jia, Liheng Chen, Zhoumianze Liu, Ben Kao, Guohao Li, Junxian He, Yu Qiao, and Zhiyong Wu. OS-Genesis: Automating GUI agent trajectory construction via reverse task synthesis. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5555–5579, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.277.

Qiaoyu Tang, Ziliang Deng, Hongyu Lin, Xianpei Han, Qiao Liang, Boxi Cao, and Le Sun. ToolAlpaca: Generalized tool learning for language models with 3000 simulated cases. *arXiv preprint arXiv:2306.05301*, 2023.

Kimi Team et al. Kimi K2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*, 2025.

Xiaoyu Tian, Haotian Wang, Shuaiting Chen, Hao Zhou, Kaichi Yu, Yudian Zhang, Jade Ouyang, Junxi Yin, Jiong Chen, Baoyan Guo, Lei Zhang, Junjie Tao, Yuansheng Song, Ming Cui, and Chengwei Liu. ASTRA: Automated synthesis of agentic trajectories and reinforcement arenas. *arXiv preprint arXiv:2601.21558*, 2026.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. 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)*, pp. 16022–16076. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.850.

Dunwei Tu, Hongyan Hao, Hansi Yang, Yihao Chen, Yu Yang, Yueqing Sun, Xingchen Liu, Furao Shen, Qi GU, Hui Su, and Xunliang Cai. ScaleEnv: Scaling environment synthesis from scratch for generalist interactive tool-use agent training. In *Forty-third International Conference on Machine Learning*, 2026.

Renxi Wang, Haonan Li, Xudong Han, Yixuan Zhang, and Timothy Baldwin. Learning from failure: Integrating negative examples when fine-tuning large language models as agents. *arXiv preprint arXiv:2402.11651*, 2024a.

Renxi Wang, Xudong Han, Lei Ji, Shu Wang, Timothy Baldwin, and Haonan Li. ToolGen: Unified tool retrieval and calling via generation. In *The Thirteenth International Conference on Learning Representations*, 2025a.

Xingyao Wang, Zihan Wang, Jiateng Liu, Yangyi Chen, Lifan Yuan, Hao Peng, and Heng Ji. MINT: Evaluating LLMs in multi-turn interaction with tools and language feedback. In *The Twelfth International Conference on Learning Representations*, 2024b.

Zezhong Wang, Xingshan Zeng, Weiwen Liu, Liangyou Li, Yasheng Wang, Lifeng Shang, Xin Jiang, Qun Liu, and Kam-Fai Wong. ToolFlow: Boosting LLM tool-calling through natural and coherent dialogue synthesis. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 4246–4263, Albuquerque, New Mexico, April 2025b. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.214.

Zhaoyang Wang, Canwen Xu, Boyi Liu, Yite Wang, Siwei Han, Zhewei Yao, Huaxiu Yao, and Yuxiong He. Agent world model: Infinity synthetic environments for agentic reinforcement learning. In *Forty-third International Conference on Machine Learning*, 2026a.

Zhenting Wang, Qi Chang, Hemani Patel, Shashank Biju, Cheng-En Wu, Quan Liu, Aolin Ding, Alireza Rezazadeh, Ankit Shah, Yujia Bao, and Eugene Siow. MCP-bench: Benchmarking tool-using LLM agents with complex real-world tasks via MCP servers. In *The Fourteenth International Conference on Learning Representations*, 2026b.

Ziyi Wang, Yuxuan Lu, Yimeng Zhang, Pei Chen, Ziwei Dong, Jing Huang, Jiri Gesi, Xianfeng Tang, Chen Luo, Qun Liu, Yisi Sang, Hanqing Lu, Manling Li, Jin Lai, and Dakuo Wang. Trajectory2Task: Training robust tool-calling agents with synthesized yet verifiable data for complex user intents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 44021–44044. Association for Computational Linguistics, 2026c. doi: 10.18653/v1/2026.acl-long.2037.

Jie Wu, Zhenru Zhang, Beichen Zhang, Xuwu Wang, Yuhui Su, Mouxiang Chen, Peng Wang, Zhihai Wang, Que Shen, Hao Zhou, An Yang, Fei Huang, Yujiu Yang, and Dayiheng Liu. Terminal-Universe: Turning agent trajectories into scalable terminal environments. *arXiv preprint arXiv:2609.04148*, 2026a.

Zijian Wu, Xiangyan Liu, xinyuan zhang, Lingjun Chen, Fanqing Meng, Lingxiao Du, Yiran Zhao, Fanshi Zhang, Yaoqi Ye, Jiawei Wang, Zirui Wang, Jinjie Ni, Yufan Yang, Arvin Xu, and Michael Qizhe Shieh. MCPMark: A benchmark for stress-testing realistic and comprehensive MCP use. In *The Fourteenth International Conference on Learning Representations*, 2026b.

Zhiheng Xi, Yiwen Ding, Wenxiang Chen, Boyang Hong, Honglin Guo, Junzhe Wang, Xin Guo, Dingwen Yang, Chenyang Liao, Wei He, Songyang Gao, Lu Chen, Rui Zheng, Yicheng Zou, Tao Gui, Qi Zhang, Xipeng Qiu, Xuanjing Huang, Zuxuan Wu, and Yu-Gang Jiang. AgentGym: Evaluating and training large language model-based agents across diverse environments. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27914–27961, Vienna, Austria, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.acl-long.1355.

Zhiheng Xi, Jixuan Huang, Chenyang Liao, Baodai Huang, Jiaqi Liu, Honglin Guo, Yajie Yang, Rui Zheng, Junjie Ye, Jiazheng Zhang, Wenxiang Chen, Wei He, Yiwen Ding, Guanyu Li, Zehui Chen, Zhengyin Du, Xuesong Yao, Yufei Xu, Jiecao Chen, Tao Gui, Zuxuan Wu, Qi Zhang, Xuanjing Huang, and Yu-Gang Jiang. AgentGym-RL: An open-source framework to train LLM agents for long-horizon decision making via multi-turn RL. In *The Fourteenth International Conference on Learning Representations*, 2026.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 52040–52094. Curran Associates, Inc., 2024. doi: 10.52202/079017-1650.

Weimin Xiong, Yifan Song, Xiutian Zhao, Wenhao Wu, Xun Wang, Ke Wang, Cheng Li, Wei Peng, and Sujian Li. Watch every step! LLM agent learning via iterative step-level process refinement. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 1556–1572, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.93.

Frank F. Xu, Yufan Song, Boxuan Li, Yuxuan Tang, Kritanjali Jain, Mengxue Bao, Zora Zhiruo Wang, Xuhui Zhou, Zhitong Guo, Murong Cao, Mingyang Yang, Hao Yang Lu, Amaad Martin, Zhe Su, Leander Melroy Maben, Raj Mehta, Wayne Chi, Lawrence Keunho Jang, Yiqing Xie, Shuyan Zhou, and Graham Neubig. TheAgentCompany: Benchmarking LLM agents on consequential real world tasks. In *Advances in Neural Information Processing Systems 38*, pp. 10291–10321. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2025a. doi: 10.52202/085713-0315.

Minrui Xu, Zilin Wang, Mengyi Deng, Zhiwei Li, Zhicheng Yang, Xiao Zhu, Yinhong Liu, Boyu Zhu, Baiyu Huang, Chao Chen, Heyuan Deng, Fei Mi, Lifeng Shang, Xingshan Zeng, and Zhijiang Guo. EnvFactory: Scaling tool-use agents via executable environments synthesis and robust RL. *arXiv preprint arXiv:2605.18703*, 2026.

Yiheng Xu, Dunjie Lu, Zhennan Shen, Junli Wang, Zekun Wang, Yuchen Mao, Caiming Xiong, and Tao Yu. AgentTrek: Agent trajectory synthesis via guiding replay with web tutorials. In *The Thirteenth International Conference on Learning Representations*, 2025b.

Zhangchen Xu, Adriana Meza Soria, Shawn Tan, Anurag Roy, Ashish Sunil Agrawal, Radha Poovendran, and Rameswar Panda. TOUCAN: Synthesizing 1.5m tool-agentic data from real-world MCP environments. *arXiv preprint arXiv:2510.01179*, 2025c.

Chen Yang, Ran Le, Yun Xing, Zhenwei An, Zongchao Chen, Wayne Xin Zhao, Yang Song, and Tao Zhang. ToolMind technical report: A large-scale, reasoning-enhanced tool-use dataset. *arXiv preprint arXiv:2511.15718*, 2025.

Jinluan Yang, Yuxin Liu, Zhengyu Chen, Chengcheng Han, Yueqing Sun, Qi Gu, Hui Su, Xunliang Cai, Fei Wu, and Kun Kuang. Topocurate: Modeling interaction topology for tool-use agent training. *arXiv preprint arXiv:2603.01714*, 2026.

Rui Yang, Lin Song, Yanwei Li, Sijie Zhao, Yixiao Ge, Xiu Li, and Ying Shan. GPT4tools: Teaching large language model to use tools via self-instruction. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 71995–72007. Curran Associates, Inc., 2023. doi: 10.52202/075280-3149.

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*, 2023.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik R Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. In *The Thirteenth International Conference on Learning Representations*, 2025.

Junjie Ye, Zhengyin Du, Xuesong Yao, Weijian Lin, Yufei Xu, Zehui Chen, Zaiyuan Wang, Sining Zhu, Zhiheng Xi, Siyu Yuan, Tao Gui, Qi Zhang, Xuanjing Huang, and Jiecao Chen. ToolHop: A query-driven benchmark for evaluating large language models in multi-hop tool use. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2995–3021, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.150.

Da Yin, Faeze Brahman, Abhilasha Ravichander, Khyathi Chandu, Kai-Wei Chang, Yejin Choi, and Bill Yuchen Lin. Agent Lumos: Unified and modular training for open-source language agents. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12380–12403, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.670.

Fan Yin, Zifeng Wang, I-Hung Hsu, Jun Yan, Ke Jiang, Yanfei Chen, Jindong Gu, Long Le, Kai-Wei Chang, Chen-Yu Lee, Hamid Palangi, and Tomas Pfister. Magnet: Multi-turn tool-use data synthesis and distillation via graph translation. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 32600–32616, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1566.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. AgentTuning: Enabling generalized agent abilities for LLMs. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3053–3077, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.181.

Jianguo Zhang, Tian Lan, Ming Zhu, Zuxin Liu, Thai Hoang, Shirley Kokane, Weiran Yao, Juntao Tan, Zhiwei Liu, Yihao Feng, Juan Carlos Niebles, Shelby Heinecke, Huan Wang, Silvio Savarese, and Caiming Xiong. xLAM: A family of large action models to empower AI agent systems. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 11583–11597, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.578.

Kangning Zhang, Weiwen Liu, Wenxiang Jiao, Kounianhua Du, Yuan Lu, Weinan Zhang, and Yong Yu. LoopTool: Closing the data–training loop for robust LLM tool calls. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 21144–21164, San Diego, California, United States, 2026a. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.968.

Shaokun Zhang, Yi Dong, Jieyu Zhang, Jan Kautz, Bryan Catanzaro, Andrew Tao, Qingyun Wu, Zhiding Yu, and Guilin Liu. Nemotron-Research-Tool-N1: Exploring tool-using language models with reinforced reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026b.

Zeyu Zhang, Guohao Li, Zhenchang Xing, Alexandros Apostolopoulos, Yu Lin Lee, and Liang Zheng. Gecko: A simulation environment with stateful feedback for refining agent tool calls. *arXiv preprint arXiv:2602.19218*, 2026c.

Zhengbo Zhang, Changtao Miao, Jinbo Su, Zhaowen Zhou, Chunxia Zhang, Xukai Wang, Ruiqi Liu, Kaiyuan Zheng, Jiansheng Cai, Bo Zhang, Zhe Li, Shiming Xiang, and Ying Yan. Visual-seeker: Towards visual-native multimodal agentic search via active visual reasoning. *arXiv preprint arXiv:2606.15231*, 2026d.

Zhiwei Zhang, Fei Zhao, Rui Wang, Zezhong Wang, Bin Liang, Jiakang Wang, Yao Hu, Shaosheng Cao, and Kam-Fai Wong. Robust tool use via Fission-GRPO: Learning to recover from execution errors. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 40477–40491, San Diego, California, United States, 2026e. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.1880.

Weikang Zhao, Xili Wang, Chengdi Ma, Lingbin Kong, Zhaohua Yang, Mingxiang Tuo, Xiaowei Shi, Yitao Zhai, and Xunliang Cai. MUA-RL: Multi-turn user-interacting agent reinforcement learning for agentic tool use. *arXiv preprint arXiv:2508.18669*, 2025.

Lucen Zhong, Zhengxiao Du, Xiaohan Zhang, Haiyi Hu, and Jie Tang. ComplexFuncBench: Exploring multi-step and constrained function calling under long-context scenario. *arXiv preprint arXiv:2501.10132*, 2025.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. LIMA: Less is more for alignment. In A. Oh, T. Naumann, A. Globerson, K. Saenko,

M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 55006–55021. Curran Associates, Inc., 2023. doi: 10.52202/075280-2400.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations*, 2024.

Yuchen Zhuang, Yue Yu, Kuan Wang, Haotian Sun, and Chao Zhang. ToolQA: A dataset for LLM question answering with external tools. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 50117–50143. Curran Associates, Inc., 2023. doi: 10.52202/075280-2180.

A Experimental Protocol

A.1 Benchmarks and Metrics

We evaluate Trajectory2Gym on Tau1, Tau2, and BFCL. Tau1 contains 115 Retail and 50 Airline tasks, while Tau2 contains 40 Retail, 20 Airline, and 40 Telecom tasks. For both Tau benchmarks, we report native task success within each domain and macro-average equally across domains.

BFCL contains 1,718 examples spanning four Non-live categories, three Live categories, and Multi-turn Base. We use the same three-group Overall aggregation as in the main text, equally averaging the Non-live mean, Live mean, and Multi-turn score.

A.2 Training and Evaluation Populations

Training-side interaction construction follows the released data partitions of each environment. Tau1 Retail uses separate training and evaluation populations. Tau1 Airline uses the public 50-task identity set for training-side interaction construction and fresh-dialogue evaluation; we therefore treat it as a seen-task, transductive setting rather than task-disjoint generalization.

For Tau2 Retail, Airline, and Telecom, training-side construction uses the official train split and evaluation uses the official test split. BFCL follows its released training-side and evaluation memberships. This provides example-level separation, although semantically related tool behaviors may occur across the two populations.

A.3 Comparison Protocol

Our primary comparison pairs each Trajectory2Gym model with its corresponding public Qwen3 initialization. Within each scale, the two models are evaluated on the same benchmark instances with the same agent interface, tool access, decoding configuration, user simulator where applicable, and native scorer. These paired evaluations isolate the effect of the post-training procedure from differences in evaluation conditions.

We additionally report public agent-training checkpoints as external references. These models are evaluated on the same benchmark populations and scoring rules, while retaining model-native prompting and tool-call interfaces where necessary. They therefore provide cross-method context rather than matched interventions; full results and model-specific settings are given in Appendix F.1.

Closed-source systems in Tables 1 and 2 are included only as contextual references under their reported protocols. The BFCL results for GPT-5.5, Gemini-3.0-Pro, and ToolACE-2-8B follow Table 3 of Gecko; we use the unrefined scores rather than the additional GATS test-time computation.

A.4 Training Configuration

All Trajectory2Gym models are initialized from their corresponding public Qwen3 checkpoints and trained for one epoch with full-parameter, assistant-token causal cross-entropy. We use a global batch size of eight rendered examples, bfloat16 precision, AdamW with weight decay 0.01, a cosine learning-rate schedule with ten warmup steps, and a maximum gradient norm of 1.0. Training is non-thinking throughout, with no preference-learning, process-reward, or reinforcement-learning objective.

Among optimization hyperparameters, only the learning rate varies with model scale: 1.1×10^{-5} for 1.7B, 7×10^{-6} for 4B and 8B, and 3.7×10^{-6} for 14B and 30B-A3B. Each learner receives the scale-conditioned training experience defined by the structural-support procedure in Appendix D.

B Interaction Synthesis and Data Composition

B.1 Training Interaction Corpus

We treat each complete realized interaction as a semantic unit of training experience. It preserves the user request, the observations exposed through native execution, and the grounded assistant decisions that can serve as learning targets. Keeping these elements within one interaction preserves the dependency between earlier evidence and later decisions.

The 1.7B–14B learners draw from a common corpus of 34,772 certified interactions spanning 2,515 training-side tasks. The 30B-A3B learner draws from an expanded corpus of 35,819 interactions spanning 3,602 tasks. Both corpora use the same compositional representation and selection principle; the expanded corpus broadens the set of available tasks and interaction structures.

The selected views contain 11,583, 13,986, 14,819, 15,805, and 19,937 interactions for the 1.7B, 4B, 8B, 14B, and 30B-A3B learners, respectively, corresponding to approximately 13.0M, 18.0M, 19.4M, 20.6M, and 23.9M supervised target tokens. Task coverage remains fixed at 2,515 tasks for the 1.7B–14B views and expands to 3,602 tasks for 30B-A3B.

The available experience is strongly skewed toward short, direct interactions. Within the common corpus, 40.9% of interactions contain at most two supervised assistant decisions, while only 10.0% contain a grounded decision after an authentic error observation. Thus corpus size alone does not imply adequate exposure to recovery, feedback-conditioned continuation, or other cross-step dependencies. This imbalance motivates explicit control over compositional support.

B.2 Compositional Structure Representation

We represent an interaction by the dependencies that connect decisions across time rather than by its sequence of native tool names. A later action may depend on an earlier observation, user update, environment response, or state transition. Examples include reusing a retrieved identifier, revising a decision after a rejection, or satisfying a prerequisite before a later state change.

To compare such dependencies across tasks and environments, we use a shared capability vocabulary $\mathcal{K}$ that abstracts semantic operations, observations, control-flow events, and interaction horizon. Temporal order over these elements defines a compositional path. The representation therefore captures how decisions depend on one another, rather than the surface API through which they are expressed.

For example, retrieving an identifier and later using it in an update instantiates the same information-dependent structure across different tool APIs. Likewise, a rejected action followed by information acquisition and a successful continuation instantiates a recovery structure independently of the backend that produced the rejection.

A crucial distinction is between *intended* and *realized* structure. A target composition can guide task construction, but it contributes support only when the dependency is actually realized during native execution. A task intended to require recovery, for example, provides no recovery support if the collected interaction never encounters and resolves the relevant obstacle. Structural support is therefore defined by the learner's realized experience, not by the generator's specification.

B.3 Synthesis from Structural Deficits

Synthesis is directed toward compositional structures that remain weakly represented in the current interaction corpus. Given a target path, task construction produces a user request and an initial environment state for which successful completion requires the corresponding dependency.

Different dependencies induce different forms of construction. Information-dependent tasks make a value needed later in the workflow obtainable only through earlier visible evidence. Recovery tasks introduce a resolvable obstacle whose cause and resolution are exposed through native feedback. State-dependent tasks require later operations to respect information or state established earlier in the interaction.

The resulting task is then executed in the native environment by a collection policy. Observations, state transitions, user responses, errors, and terminal outcomes are therefore consequences of execution rather than annotations inserted by the generator. Collection-side guidance may shape the attempted interaction, but it does not become learner-visible evidence unless it is exposed through the native interface.

After execution, the realized interaction is mapped back to the compositional representation. Only structures that actually occurred contribute support, and decision-grounded supervision determines which realized assistant decisions can become learning targets. Subsequent synthesis is then directed toward structures that remain under-supported.

Thus intended structure determines *where to generate*, whereas realized structure determines *what experience was actually obtained*.

C Decision-Grounded Supervision

Decision grounding asks whether a realized assistant decision can be reconstructed from the information available to the policy when that decision is made. This differs from execution validity: an action may be accepted by the environment while still relying on information unavailable to the learner. The grounding procedure therefore separates what happened in the collected interaction from which decisions constitute reproducible supervision.

C.1 Policy-Visible Evidence

For a decision at step t, grounding is defined relative to the policy-visible history $V(H_{<t})$. Only information exposed through the native agent interface can justify the decision. User messages and assistant-visible tool observations may therefore provide evidence, whereas collector-private state and private external operations cannot directly support a target.

The relevant notion is information availability rather than textual overlap. A decision is grounded when the information needed to choose it is recoverable from the preceding visible interaction. For example, an identifier known only to the collection policy cannot justify an assistant action even if the resulting call succeeds. If the same identifier is subsequently returned through a visible tool observation, later decisions may legitimately depend on it.

The same principle applies to dynamically available tools. When tool discovery is part of the native interface, a tool can support a supervised action only after its identity and usable interface have become visible to the assistant. More generally, arguments referring to users, orders, items, accounts, or other environment entities must be supported by preceding visible evidence.

Private events may change the underlying environment without entering the student history. Their consequences become usable only when they are subsequently exposed through an observation available to the assistant. Grounding therefore preserves the distinction between latent environment state and evidence available to the acting policy. These checks implement specific requirements for visible provenance; they do not prove that every admitted action is uniquely determined by its visible history or verify every possible reason for choosing it.

C.2 Decision and Workflow Eligibility

A supervised decision must satisfy both execution and information requirements. Execution validity asks whether the action is admissible under the native environment semantics; decision grounding asks whether its choice is supported by the policy-visible history. Neither condition subsumes the other.

This distinction is especially important for successful trajectories. Trajectory-level success certifies an episode outcome, but does not imply that every intermediate action is an appropriate imitation target. A successful action that depends on collector-private information remains ungrounded, while a grounded decision may occur only after several observations or clarifications have accumulated.

For tasks with ordered state changes, local eligibility is supplemented by workflow consistency. Required operations must appear in the realized interaction in an order compatible with the task.

Information gathering, user clarification, and recovery may occur between those operations, but a rejected or masked action cannot satisfy a required workflow step simply because the trajectory eventually succeeds.

The resulting supervision therefore distinguishes three levels: *trajectory success*, which concerns completion of the interaction; *execution validity*, which concerns whether an individual action is accepted by the environment; and *decision grounding*, which concerns whether that action is justified by information available when it is taken. Only decisions satisfying the relevant levels contribute imitation loss.

Native environments occasionally provide action-level validation without a corresponding observation. Such validation can authorize the action as a target, but does not create an unobserved action–observation transition.

C.3 Failure as Evidence for Recovery

An action need not be a valid imitation target to contribute useful information to the interaction. This is particularly important for recovery.

When an assistant action fails, the failed action itself receives no imitation loss. If the action and its authentic error response are visible through the native interface, however, they remain part of the policy-visible history. A later recovery decision can then be supervised from the richer context created by that feedback. The learner therefore observes the consequence of the failed attempt without being trained to reproduce the failure.

Repeated actions are treated according to the same principle. A repeated read after a state transition may reveal new information and support a new decision, even when its serialized action matches an earlier call. Likewise, a repeated operation may be required by the task after the environment state has changed. Surface repetition alone is therefore insufficient to identify redundancy.

By contrast, a repeated action that contributes no new evidence, state change, or required operation provides no additional compositional support. The distinction is semantic: what matters is whether the repeated action changes the information or state on which subsequent decisions depend.

This treatment allows one realized interaction to contain both masked failures and valid supervision for their recovery. Figure 5 provides a concrete example in Appendix G.2.

D Structural-Support Learning

Grounding determines which realized decisions can provide valid supervision, but it does not determine how certified experience should be distributed across interaction structures. Structural-support learning addresses two remaining questions: which interactions should be retained for a learner, and how much aggregate optimization weight each retained interaction should receive.

D.1 Structural Coverage and Support Depth

We distinguish whether a compositional structure is represented from how much repeated support it receives. Let $\mathcal{P}(\tau)$ denote the compositional structures realized by interaction τ. For repeated-support accounting, each interaction is associated with one canonical path $p(\tau) \in \mathcal{P}(\tau)$. This assignment does not imply that the interaction contains only one useful dependency; its remaining capability content continues to contribute to structural coverage.

For a selected set $\mathcal{S}$, the support of path p is

$$n_{\mathcal{S}}(p) = \sum_{\tau \in \mathcal{S}} \mathbf{1}[p(\tau) = p]. \tag{9}$$

For learner scale s, let d_s denote the desired support depth. The remaining deficit of path p is

$$\delta_s(p; \mathcal{S}) = \big[d_s - n_{\mathcal{S}}(p)\big]_+. \tag{10}$$

For an unselected interaction τ, its contribution to repeated structural support is

$$\begin{aligned}\Delta_s(\tau \mid \mathcal{S}) &= \delta_s(p(\tau); \mathcal{S}) - \delta_s(p(\tau); \mathcal{S} \cup \{\tau\}) \\ &= \mathbf{1}\big[n_{\mathcal{S}}(p(\tau)) < d_s\big].\end{aligned} \tag{11}$$

Thus $d_s = 1$ corresponds to path coverage, while larger values allocate repeated experience to structures that are already represented. For the scaling study, we use $d_s = 1, 2, 3, 4$, and 8 for the 1.7B, 4B, 8B, 14B, and 30B-A3B learners, respectively.

The depth schedule specifies how training experience is allocated; it is not a monotonic claim about the amount of support that maximizes downstream performance. The fixed-8B sensitivity study in Table 4 confirms this distinction: increasing depth adds interactions and target tokens, but the three-benchmark mean does not improve monotonically.

D.2 Experience Allocation

Selection balances two complementary objectives: preserving breadth over tasks and compositional structures, and providing repeated support for structures that remain under-represented.

We begin from a fixed certified core shared across training views. Selection then proceeds hierarchically. Task coverage preserves completing demonstrations across the available training tasks; structural coverage adds interactions that expose previously uncovered capability atoms and canonical paths; finally, support expansion allocates additional interactions toward paths with positive deficit δ_s. This ordering prevents repeated exposure to frequent structures from displacing broad task and capability coverage.

Within each stage, interactions that provide comparable structural support are distinguished by secondary criteria. Task-level selection favors concise, completing demonstrations, while structural selection favors interactions that add visible capability content with limited redundant supervision. Recovery and branching are retained when they contribute additional interaction structure rather than merely increasing trajectory length.

The resulting procedure is deliberately independent of downstream evaluation outcomes. Benchmark identity, evaluation score, teacher identity, and test-task identity are not used as selection features. Likewise, negative log-likelihood under the public initialization does not determine which interactions are retained. Training-set composition therefore emerges from task breadth, compositional coverage, and structural-support deficits rather than from benchmark-specific quotas or difficulty scores.

The exact deterministic ordering, per-task expansion cap, and tie-breaking rules are specified in the released configuration. Equation (11) characterizes one component of the allocation rule; it does not imply global optimality of the complete multi-stage selector. Appendix E.2 analyzes the idealized deficit stage under a pure cardinality budget.

D.3 Parent-Normalized Learning Objective

Structural selection operates on complete semantic interactions, whereas a single interaction may render into multiple native training examples. A conventional row-averaged objective would therefore assign greater optimization mass to interactions that happen to produce more rendered examples. We instead normalize the objective at the semantic-interaction level.

Suppose selected interaction i renders into examples $j \in \mathcal{J}_i$. Let $m_{ijk} \in \{0, 1\}$ denote the causal supervision mask for token k in rendered example j, let $n_{ij} = \sum_k m_{ijk} > 0$ be its number of supervised tokens, and let $N_i = \sum_{j \in \mathcal{J}_i} n_{ij}$ be the total number of supervised tokens associated with interaction i. The mean certified-token loss of rendered example j is

$$\ell_{ij}(\theta) = -\frac{1}{n_{ij}} \sum_k m_{ijk} \log \pi_\theta\big(y_{ijk} \mid \widetilde{h}_{ij,<k}\big). \tag{12}$$

If P_s semantic interactions are selected for learner scale s, the parent-normalized empirical objective is

$$\mathcal{L}_{\text{parent}}(\theta) = \frac{1}{P_s} \sum_i \sum_{j \in \mathcal{J}_i} \frac{n_{ij}}{N_i} \ell_{ij}(\theta). \tag{13}$$

The inner weights sum to one for every semantic interaction. Consequently, each selected interaction receives the same aggregate coefficient $1/P_s$, while its certified target occurrences share that mass internally. This separates the intended importance of an interaction from the number of rendered examples used to represent it.

If $M_s = \sum_i |\mathcal{J}_i|$ is the number of rendered examples, the same objective can be implemented with per-example coefficient

$$\omega_{ij} = \frac{M_s}{P_s} \frac{n_{ij}}{N_i}. \tag{14}$$

Indeed,

$$\frac{1}{M_s} \sum_i \sum_{j \in \mathcal{J}_i} \omega_{ij} \ell_{ij}(\theta) = \mathcal{L}_{\text{parent}}(\theta). \tag{15}$$

Hence, for a fixed set of certified target occurrences and unchanged conditioning histories, repartitioning one semantic interaction into a different number of rendered examples does not change its intended aggregate weight. This is the rendering-invariance property analyzed formally in Appendix E.4. Finite optimization trajectories may still depend on sample ordering and batch construction.

E Theoretical Notes on the Construction

This appendix records elementary properties of the three stages of Section 4, in the notation of Section 3. They are not generalization guarantees; they make precise what each stage can and cannot do, and they are the basis for the justifications given in the main text.

E.1 Grounding and Hidden Information

A collected decision is a triple (c, z, a): the student-visible context $c = c_t$ of Eq. (1), the collector-private information z (teacher hints, private events, identifiers known only to the collector), and the assistant action a. Let (C, Z, A) denote a draw from the collection distribution restricted to any set of decisions, and let H and I denote conditional entropy and mutual information. A student is any policy $\pi_\theta(a \mid c)$; it never observes z.

Proposition 1 (Grounding floor)**.** *For every student policy,*

$$\mathbb{E}\big[-\log \pi_\theta(A \mid C)\big] \;\geq\; H(A \mid C) \;=\; H(A \mid C, Z) + I(A; Z \mid C), \tag{16}$$

with equality in the inequality if and only if $\pi_\theta(\cdot \mid c) = p(\cdot \mid c)$ for almost every c. If the collector acts deterministically given (c, z), the floor equals $I(A; Z \mid C)$, the information about the action that is carried by private state and unavailable to the student.

Proof. $\mathbb{E}[-\log \pi_\theta(A \mid C)] - H(A \mid C) = \mathbb{E}_C\big[\mathrm{KL}\big(p(\cdot \mid C) \,\|\, \pi_\theta(\cdot \mid C)\big)\big] \geq 0$ by Gibbs' inequality, with equality if and only if the two conditionals agree almost surely. The decomposition is the chain rule $H(A \mid C) = H(A \mid C, Z) + I(A; Z \mid C)$, and $H(A \mid C, Z) = 0$ when A is a function of (C, Z). □

Remark 1. The grounding check excludes targets whose identifier-like arguments lack preceding visible support. Exclusion by this operational rule does not by itself imply $I(A; Z \mid C) > 0$ in the collection distribution. When an action retains dependence on collector-private information after conditioning on visible context, Proposition 1 identifies the additional optimal log loss faced by a student restricted to C relative to one observing (C, Z). This motivates excluding unsupported targets; it does not assert that every excluded target has a strictly positive information term or that every admitted action is uniquely determined by C.

Remark 2 (Certification is a nested filter). Since $m_t = R_\tau \, \mathbf{1}[q_\tau \preceq A_\tau] \, e_t \, v_t$ is a product of indicators, the certified decisions are exactly those passing all four checks, a decision failing any one check is excluded regardless of the others, and the eligible pool $\{\tau : \sum_t m_t > 0\}$ is a subset of the success-filtered pool $\{\tau : R_\tau = 1\}$. Certification therefore only ever removes targets relative to outcome filtering; Proposition 1 identifies what is removed.

E.2 Deficit-Driven Selection Maximizes Capped Support

For a candidate pool $\mathcal{C}$, a selected set $\mathcal{S} \subseteq \mathcal{C}$, and a cap $d \geq 1$, define the capped support

$$F_d(\mathcal{S}) = \sum_{p \in \mathcal{P}} \min\big(n_{\mathcal{S}}(p),\, d\big), \qquad \sum_{p \in \mathcal{P}} \delta_d(p; \mathcal{S}) = d\,|\mathcal{P}| - F_d(\mathcal{S}), \tag{17}$$

where $\mathcal{P}$ is the set of feasible canonical paths and $\delta_d(p; \mathcal{S}) = [d - n_{\mathcal{S}}(p)]_+$ as in Eq. (7). Reducing the total deficit is therefore the same as increasing F_d. Let N_p be the number of pool candidates whose canonical path is p.

Proposition 2 (Exactness of the deficit stage)**.** *1. F_d is monotone and submodular on subsets of $\mathcal{C}$, with marginal gain $\Delta_d(\tau \mid \mathcal{S}) = \mathbf{1}[n_{\mathcal{S}}(p(\tau)) < d]$, which is Eq.* (11)*.*

2. Starting from any fixed set $\mathcal{S}_0$ and adding at most B candidates, the greedy rule that repeatedly adds a candidate with positive marginal gain attains

$$\max_{\mathcal{S} \supseteq \mathcal{S}_0,\ |\mathcal{S} \setminus \mathcal{S}_0| \leq B} F_d(\mathcal{S}) = \min\Big(F_d(\mathcal{S}_0) + B,\ \sum_{p \in \mathcal{P}} \min(N_p, d)\Big). \tag{18}$$

3. Without a cap ($d = \infty$), $F_\infty(\mathcal{S}) = |\mathcal{S}|$ is modular: every budget-feasible selection is optimal and the objective expresses no preference among paths. The cap is what encodes breadth, and $d = 1$ counts covered paths.

Proof. (1) Each candidate contributes to exactly one path, so $n_{\mathcal{S}}(p)$ is modular in $\mathcal{S}$, and $n \mapsto \min(n, d)$ is nondecreasing and concave; a sum of nondecreasing concave functions of modular functions is monotone submodular. Adding τ raises $\min(n_{\mathcal{S}}(p(\tau)), d)$ by one exactly when $n_{\mathcal{S}}(p(\tau)) < d$. (2) Upper bound: each added candidate raises F_d by at most one, and $F_d(\mathcal{S}) \leq \sum_p \min(N_p, d)$ for every $\mathcal{S} \subseteq \mathcal{C}$. Achievability: whenever $F_d(\mathcal{S}) < \sum_p \min(N_p, d)$ there is a path with $n_{\mathcal{S}}(p) < \min(N_p, d)$, hence an unselected candidate of that path with marginal gain one; greedy adds one per step, so F_d grows by exactly one per step until the budget or the bound is reached. (3) Immediate from $\min(n, \infty) = n$. □

Remark 3. The implemented selection has several stages (a fixed core, one completing exemplar per task, atom and path coverage, then deficit reduction) and breaks ties by grounding risk, recovery, branching, redundancy, and cost (Appendix D). Proposition 2 concerns the deficit stage with $\mathcal{S}_0$ the output of the earlier stages: its greedy order is exact for the capped objective, and tie-breaking only chooses among candidates with equal marginal gain. The coverage stage is the $d = 1$ case.

E.3 Random Surplus Selection Misses Rare Paths

Proposition 3 (Miss probability under uniform selection)**.** *Suppose B surplus candidates are drawn uniformly without replacement from a pool of N candidates, and path p has N_p candidates in that pool. Then*

$$\Pr\big[n_{\mathcal{S}}(p) = 0\big] = \frac{\binom{N - N_p}{B}}{\binom{N}{B}} = \prod_{j=0}^{N_p - 1} \frac{N - B - j}{N - j} \;\leq\; \Big(1 - \frac{B}{N}\Big)^{N_p}, \tag{19}$$

with equality for $N_p = 1$: a path represented once is missed with probability exactly $1 - B/N$. The expected number of covered paths is $\sum_p \big(1 - \Pr[n_{\mathcal{S}}(p) = 0]\big)$.

Proof. $n_{\mathcal{S}}(p)$ is hypergeometric; the product form counts the B-subsets that avoid all N_p candidates of p, and each factor is at most $(N - B)/N$ because $(N - B - j)/(N - j)$ is nonincreasing in j. □

Remark 4. Coverage-first selection covers every feasible path whenever the budget allows (Proposition 2 with $d = 1$), whereas uniform selection misses rare paths with a probability that decays only geometrically in N_p. Taking the overall selection ratio of the fixed-8B comparison as an illustration ($B/N \approx 14{,}819/34{,}772 \approx 0.43$), a path represented once is missed with probability about 0.57 and one represented three times with probability about 0.19. This is the mechanism behind the 942 canonical paths that random surplus selection left uncovered at the same budget (Figure 3); the actual control draws the surplus after fixed anchors, so the numbers are illustrative.

E.4 Parent Normalization

Let parent i render into children $j \in \mathcal{J}_i$ with certified target counts n_{ij} and $N_i = \sum_j n_{ij}$, and let P_s and $M_s = \sum_i |\mathcal{J}_i|$ be the numbers of parents and children. Write ℓ_{ij} for the child-mean loss of Eq. (13), and let $\mathcal{L}_{\mathrm{row}} = \frac{1}{M_s} \sum_{i,j} \ell_{ij}$ denote per-example (row) averaging.

Proposition 4 (Training mass under the two objectives)**.** *1. Under $\mathcal{L}_{\mathrm{row}}$, parent i receives total coefficient $|\mathcal{J}_i|/M_s$, so the coefficients of two parents are in the ratio of their child counts; under $\mathcal{L}_{\mathrm{parent}}$ every parent receives $1/P_s$.*

2. Under $\mathcal{L}_{\mathrm{parent}}$ every certified target occurrence of parent i has weight $1/(P_s N_i)$; under $\mathcal{L}_{\mathrm{row}}$ an occurrence in child j has weight $1/(M_s n_{ij})$, so occurrences in short children are up-weighted relative to those in long children.

3. For a fixed multiset of target occurrences with fixed conditioning prefixes, $\mathcal{L}_{\mathrm{parent}}$ is invariant to how the occurrences are partitioned into children, whereas $\mathcal{L}_{\mathrm{row}}$ is not.

4. The implementation's child coefficients $\omega_{ij} = \alpha\, n_{ij}/N_i$ with $\alpha = M_s/P_s$ satisfy $\frac{1}{M_s} \sum_{i,j} \omega_{ij} \ell_{ij} = \mathcal{L}_{\mathrm{parent}}$, Eq. (15).

Proof. (1) Sum the coefficients of the children of parent i in each objective. (2) Expand $\ell_{ij} = \frac{1}{n_{ij}} \sum_k m_{ijk} \left(-\log \pi_\theta(y_{ijk} \mid \widetilde{h}_{ij,<k})\right)$ and multiply by the parent coefficients from (1). (3) By (2), $\mathcal{L}_{\mathrm{parent}} = \frac{1}{P_s} \sum_i \frac{1}{N_i} \sum_{u \in i} \left(-\log \pi_\theta(y_u \mid h_u)\right)$ depends only on the occurrence multiset and on N_i, neither of which changes under regrouping; the row weight $1/(M_s n_{ij})$ depends on the partition through M_s and n_{ij}. (4) Substitute ω_{ij}: $\frac{1}{M_s} \sum_{i,j} \frac{M_s}{P_s} \frac{n_{ij}}{N_i} \ell_{ij} = \frac{1}{P_s} \sum_i \sum_j \frac{n_{ij}}{N_i} \ell_{ij}$. □

Remark 5. Item (1) is why long multi-turn interactions, which render into more children, dominate $\mathcal{L}_{\mathrm{row}}$; the parent-normalization row of Table 3 measures the effect of removing this reweighting on identical renderings and masks. Items (3) and (4) are what make the objective independent of native rendering details. Finite-run updates can still depend on sample order and batching, as noted in Appendix D.3.

F Additional Experimental Analysis

The main text reports the five-scale paired study, representative public agent-training baselines, and the matched 8B component controls. This appendix complements those results in two ways. First, it reports the complete public-checkpoint reference set evaluated under our unified benchmark protocol. Second, it provides additional analysis of the matched controls and structural-support allocation.

F.1 Complete Public-Checkpoint Reference Set

The main tables report representative 8B checkpoints from the public agent families, together with MUA-RL-32B. Table 5 expands this comparison to all sixteen public checkpoints in the current quantitative reference set. All models use the benchmark populations and scoring rules described in Appendix A, while model-native prompting and tool-call interfaces are preserved where necessary. These results therefore provide a common evaluation reference rather than a matched training comparison.

Table 5: **Complete public-checkpoint evaluation** (%). Tau scores are domain macro averages; Mean equally averages Tau1, Tau2, and BFCL. Trajectory2Gym-8B is shown for reference.

Model	Tau1	Tau2	BFCL	Mean
MUA-RL				
8B	34.65	25.83	58.17	39.55
14B	47.57	45.00	64.05	52.21
32B	54.04	45.00	72.48	57.17
Simia (Qwen3-8B)				
Tau-SFT	36.61	42.50	43.08	40.73
Tau-SFT-RL	39.17	41.67	42.50	41.11
AgentBench-SFT	18.35	16.67	64.37	33.13
OfficeBench-SFT*	19.04	9.17	65.82	31.34
Simia (Qwen2.5)†				
Tau-SFT (7B)	39.74	35.00	39.73	38.16
Tau-SFT-RL (7B)	32.87	33.33	41.03	35.75
Tau-SFT (32B)	48.39	45.83	43.43	45.88
AgentBench-SFT (7B)	12.04	14.17	42.25	22.82
OfficeBench-SFT (7B)	19.48	15.00	41.23	25.24
OfficeBench-SFT-RL (7B)	15.30	17.50	42.16	24.99
Other public agents				
LoopTool-8B	23.57	24.17	68.66	38.80
EnvFactory-8B (thinking)	40.13	26.67	77.55	48.12
MidTool-RL-8B	24.87	29.17	73.73	42.59
Trajectory2Gym (Qwen3-8B)	51.13	74.17	78.62	**67.97**

* The OfficeBench Qwen3-8B BFCL diagnostic uses modified reasoning-token handling; its BFCL and derived Mean are excluded from the main comparison. † Qwen2.5 uses its native assistant prefix.

F.2 Cross-Benchmark Behavior

The complete checkpoint set reinforces the distinction between function-calling accuracy and stateful task execution. The Qwen3 Simia checkpoints trained on Tau trajectories transfer more strongly to Tau than their AgentBench- or OfficeBench-trained counterparts. LoopTool, EnvFactory, and MidTool obtain comparatively strong BFCL scores without corresponding Tau2 performance. These differences motivate reporting all three benchmarks rather than a single function-calling score.

The largest separation in this reference set occurs on Tau2, where Trajectory2Gym-8B exceeds every re-evaluated public checkpoint, including the larger MUA-RL variants. Across Tau1, Tau2, and BFCL, it also attains the highest equal-weight mean among the public checkpoints evaluated here.

F.3 Matched Component Controls

The main-text ablation in Table 3 reports four matched Qwen3-8B comparisons. Because the interventions operate on different stages of the construction procedure, each comparison has its own matched reference; the resulting gains are therefore not an additive decomposition of the full-method improvement.

Targeted expansion. The reference uses the existing certified interaction corpus, while the proposed condition adds experience synthesized around under-supported structures before applying the same downstream selection and training procedure. This comparison changes candidate supply and therefore does not hold the training-token budget fixed.

Decision grounding. The semantic interactions and policy-visible histories are held fixed, while the set of supervised assistant decisions changes. The comparison therefore isolates whether success-derived targets are filtered by decision-time evidence.

Structural selection. Both conditions draw from the same eligible interaction corpus and retain the same task and capability anchors, 14,819 semantic interactions, and target-token budget. Structural allocation covers 6,360 canonical paths, compared with 5,418 under random surplus selection.

Parent normalization. The selected semantic interactions, rendered examples, target masks, and sample order are held fixed. Only the aggregate optimization mass assigned to each semantic interaction changes.

F.4 Structural Coverage under a Matched Budget

The structural-selection control connects experience allocation with structural coverage. At the same budget of 14,819 semantic interactions and matched target-token mass, structural selection covers 6,360 canonical paths, compared with 5,418 under random surplus selection: an increase of 942 paths.

The corresponding performance change is concentrated on the stateful Tau benchmarks. As reported in Table 3, structural selection increases Tau1 and Tau2 while slightly decreasing BFCL, yielding a 2.32-point gain in the equal-weight three-benchmark mean. Together, these results suggest that the allocation rule primarily changes which stateful interaction structures are represented, rather than simply increasing the amount of supervision.

F.5 Support-Depth Sensitivity

Table 4 varies the structural-support depth at fixed Qwen3-8B. Changing depth alters both the number of retained interactions and the amount of supervised target mass, so this experiment should be interpreted as sensitivity to the resulting training allocation rather than as an isolated causal effect of repeated path exposure.

The three-benchmark mean is highest at depth two and decreases as additional support is allocated. Tau2 accounts for most of this decline, while BFCL remains comparatively stable. The result reinforces the role of support depth as an allocation parameter: more repeated support is not automatically better. The support schedule used in the scaling study was specified independently of these sensitivity outcomes.

G Qualitative Evidence

The examples illustrate behavior rather than estimate its frequency or causally attribute improvements to individual components. They retain both successful interactions and a remaining failure.

Appendix G.1 walks through the same-task comparison of Figure 4, Appendix G.2 shows how grounded supervision treats a recorded training interaction (Figure 5), and Appendix G.3 shows six further paired cases against re-evaluated checkpoints, including a remaining failure.

G.1 Full Same-Task Comparison

Figure 4 compares three agents on a Retail workflow requiring updates to a laptop order and a watch order. The requested shipping address appears in an earlier order observation, while the user profile contains a different address. Completing the request requires both reusing that evidence and updating the shipping address before an item modification changes the order status.

Simia Tau-SFT. After observing the requested address, Simia repeatedly queries the user profile, which returns the other address. It transfers the request to a human without completing either update.

MUA-RL-8B. The model first modifies the laptop item in order `#W3730488`. The returned status is `pending (item modified)`. Its subsequent New York address update is rejected with `non-pending order cannot be modified`; a later cancellation attempt is rejected as well. Changing the user profile to New York does not change this order: subsequent order reads still return Houston. The native task reward is 0.

Trajectory2Gym. The agent obtains confirmation, applies the retrieved address while the order remains pending, and then updates the laptop and watch items. The native task succeeds, whereas both external-baseline trajectories fail.

These independently generated dialogues share a task and system policy, but not identical observation histories. The selected case illustrates evidence use and prerequisite ordering, rather than estimating failure frequency or proving a causal component effect.

G.2 Decision-Level Grounding Example

Table 6 shows the successful Retail training interaction of Figure 5, the scenario that opens Section 1, which contains one failed lookup followed by grounded recovery and several dependent state-changing operations. The failed call remains visible but receives no imitation loss; subsequent decisions are supervised once their required evidence becomes visible.

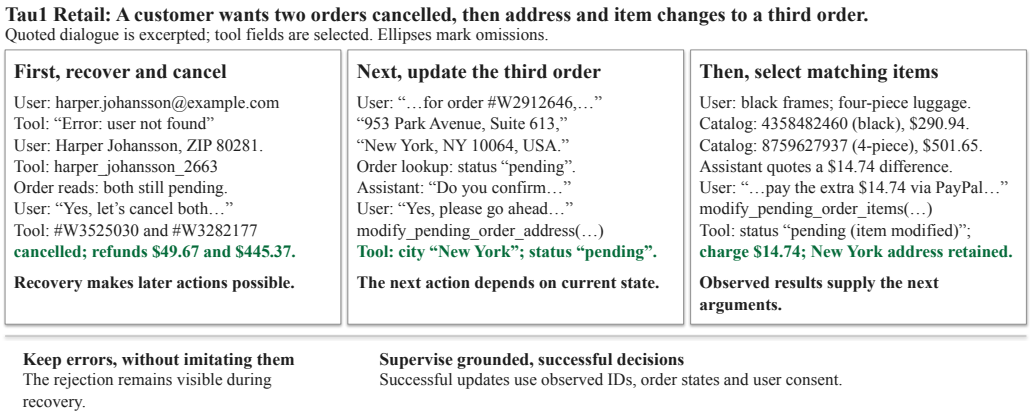


Figure 5: **Grounded supervision within a compositional interaction.** A failed email lookup remains visible context without receiving imitation loss. Subsequent visible evidence supports identity recovery, cancellation, address modification, and catalog-grounded item updates. Dialogue and tool fields are abridged for display.

Table 6: Decision-level supervision in the multi-order Retail case. The failed lookup uses an email supplied by the user: its argument is visible, but its execution fails. The call and error remain context; user messages and tool responses receive no imitation loss.

Decision	Assistant decision	Preceding visible evidence	Supervision
0	Ask for identity	User requests cancellations and updates	Supervised
1	Lookup by email	User-supplied email; tool returns user not found	Context only
2–4	Recover identity	Error, then user-supplied name and ZIP 80281	Supervised
5–7	Read two orders; confirm cancellations	User gives both order IDs; reads show pending status	Supervised
8–9	Cancel both orders	User explicitly confirms both cancellations	Supervised
10–14	Report results; confirm and change address	Third order is pending; user supplies and confirms the address	Supervised
15–17	Read two catalogs; confirm item changes	Requested variants and PayPal ID from user; IDs and prices from catalogs	Supervised
18	Modify both items	Catalog-derived replacements; user confirms the $14.74 difference	Supervised
19	Report updated order	Returned item IDs, address, status, and payment entry	Supervised

The agent recovers identity, confirms and cancels two pending orders, then updates another order's address before changing its items. Catalog observations supply the replacement identifiers, and user confirmation authorizes the consequential changes. The case illustrates how one visible interaction can contain both masked failure and valid supervision for evidence-dependent recovery.

G.3 Additional Behavior Cases

The six cases in Figures 6–11 illustrate changing intent, feedback use, dependent operations, state prerequisites, observation binding, and a remaining failure. Each compares a named external agent with Trajectory2Gym on the same task, using their respective trained checkpoints. Dialogues evolve independently, so these cases illustrate behavior rather than isolate the cause of an aggregate performance difference. The figures excerpt dialogue and selected tool fields, with omissions marked; m is the original message index and "log" is the BFCL source-log index. Case-specific source paths, full records, and native outcomes are retained in the accompanying JSON.

Tau2 Retail task 5: the user initially requests a larger bottle and a dimmer desk lamp. Both agents face the same task and episode seed, but dialogues develop independently.

Simia Tau-SFT (Qwen3-8B)

[m26] Proposes lamp 7453605304 and bottle 4579334072.
[m27 User, excerpt] "...desk lamp exchange for now."

[m28 Assistant] exchange_delivered_order_items(
order_id="#W6390527", new_item_ids=["7453605304"], ...)
[m29 Tool] status: "exchange requested"
[m31 User, excerpt] "Instead, I'd like to return the water bottle."
[m33 User] Confirms the bottle return.
[m34 Assistant] return_delivered_order_items(
order_id="#W6390527", item_ids=["8538875209"], ...)
[m35 Tool] "Error: Non-delivered order cannot be returned"

...
[m44 Assistant] transfer_to_human_agents(...)
[m45 Tool] "Transfer successful"

The earlier exchange changes the order state. A later, explicitly confirmed bottle return is rejected.

Native task result: FAIL (0)

Trajectory2Gym (Qwen3-8B)

[m12–15] Product catalogs for bottle and lamp are retrieved.
[m20] Assistant requests confirmation of the proposed exchange.
[m21 User, excerpt] "I'd like to keep the water bottle as is."
[m22–23] Lamp-only price difference is recalculated: $12.07.
[m24] Assistant requests lamp-only confirmation.

[m25 User, excerpt]
"Instead, I'd like to return the water bottle."
[m26 Assistant: return_delivered_order_items]
order_id: "#W6390527"
item_ids: ["8538875209"]
payment_method_id: "paypal_7644869"
[m27 Tool: selected fields]
status: "return requested"; return_items: ["8538875209"]

exchange_items: null; exchange_new_items: null

The trained rollout follows the revised bottle-return request. The two user dialogues are not identical.

Native task result: PASS (1)

Figure 6: **Changing intent during a stateful interaction.** Simia submits an exchange before a later return request, which the changed order state prevents. In the independently generated Trajectory2Gym dialogue, the user changes the request before submission and the return succeeds. The different histories make this a behavior illustration, not an identical-context comparison.

Tau2 Telecom: achieve excellent mobile-data speed in France. Phone roaming is off and Data Saver is on; backend roaming is already enabled.

MUA-RL-8B

[m16 Assistant] check_network_status()
[m17 Tool] "Error: Tool 'check_network_status' not found."
[m18 Assistant] check_status_bar()
[m19 Tool] "Error: Tool 'check_status_bar' not found."
[m20 Assistant] check_apn_settings()
[m21 Tool] "Error: Tool 'check_apn_settings' not found."
[m22 Assistant] check_data_restriction_status()
[m23 Tool] "Error: Tool 'check_data_restriction_status' not found."
[m24–35] Six more phone-side calls also fail.

No phone-side action is delegated successfully.
Termination reason: too_many_errors.

MUA-RL makes ten assistant-side calls to phone tools. The user must execute those tools; the episode ends at the error limit.

Native task result: FAIL (0)

Trajectory2Gym (Qwen3-8B)

[m13 User] Phone roaming is reported as disabled.
[m14–16] Assistant requests phone roaming; user turns it on.

[m20 Tool] "Speed Test Result: 55.00 Mbps (Good). ..."
[m21 User, excerpt] "...I really need excellent internet speed..."
[m22 Assistant, excerpt]
"Can you please check_data_restriction_status on your phone?"
[m24 Tool] "Data Saver mode is ON (limits data usage)."
[m26] Assistant asks the user to turn Data Saver off.
[m28 Tool, excerpt] "Data Saver Mode is now OFF."

[m30] Assistant asks the user to repeat the speed test.
[m32 Tool]
"Speed Test Result: 275.00 Mbps (Excellent). Connection is very fast."
Native mobile-data and excellent-speed assertions both pass.

The assistant delegates phone actions and uses the reported speed to choose the next diagnostic step.

Native task result: PASS (1)

Figure 7: **Delegating phone-side actions and using feedback.** MUA-RL repeatedly attempts phone-side operations unavailable to the assistant. Trajectory2Gym delegates those operations to the user and adapts its continuation to the reported device state and connection speed, completing the task.

BFCL multi_turn_base_166: obtain a business-class fare for 2026-07-15, set a $2,000 budget, book, retrieve the invoice and contact support.

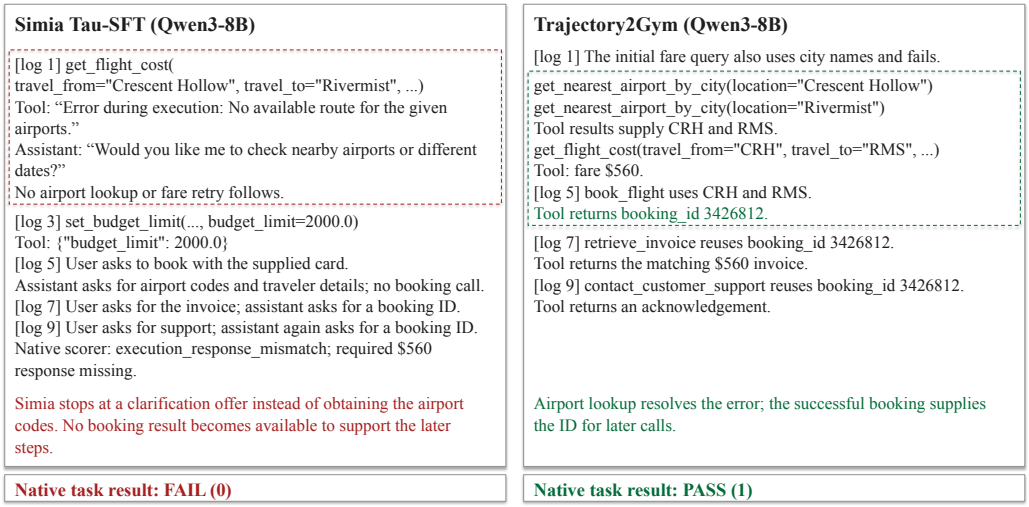


Figure 8: **Recovery enables dependent operations.** After a route error, Simia proposes a lookup but does not complete the booking. Trajectory2Gym retrieves the needed airport codes, books the flight, and reuses the returned booking identifier for invoice and support calls. Only the Trajectory2Gym interaction satisfies the native task checker.

Tau2 Retail task 18: the user initially asks to return an office chair, then changes the request to an exchange. The exchange requires a delivered order.

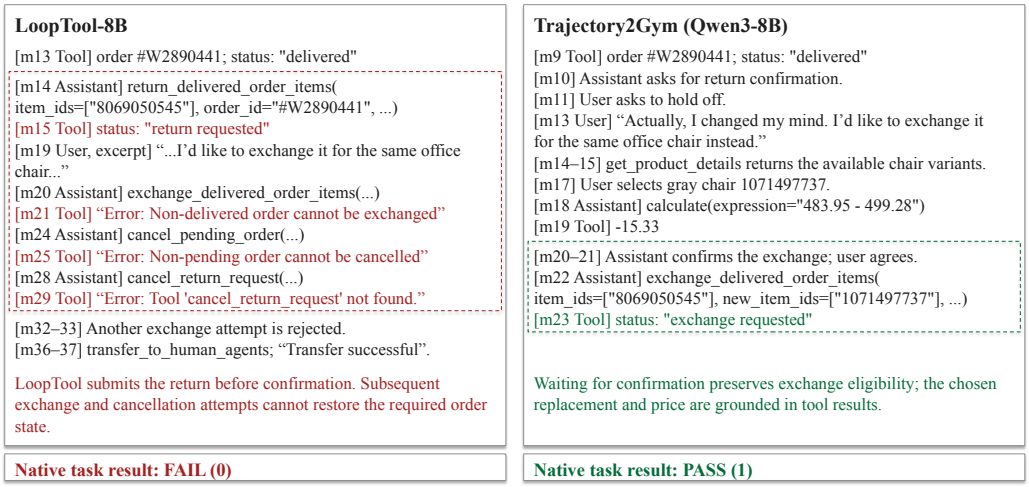


Figure 9: **An early return blocks later exchange attempts.** LoopTool submits a return before confirmation, leaving later exchange and cancellation attempts invalid. Trajectory2Gym preserves the delivered state while obtaining confirmation, then submits the user-selected exchange.

BFCL multi_turn_base_159: quote the economy fare on 2026-11-15 between the first two returned airports. A later request separately asks for an SFO–LAX booking and insurance.

Simia Tau-SFT (Qwen3-8B)

[log 1] list_all_airports(); get_flight_cost(RMS, SBK, ...)
Tool: {"travel_cost_list": [440.0]}

[log 3] User requests SFO–LAX booking and supplies the access token.
Assistant: "Could you please confirm the user's first and last name?"
No book_flight call is made in this turn.
[log 5, step 0] authenticate_travel(...)
Tool: expires_in 2; access_token "251675"
[step 1] authenticate_travel(...)
Tool: expires_in 2; access_token "966117"
[steps 2–20] The authentication call repeats.
No booking or insurance purchase is completed.

Native scorer: empty_turn_model_response for the booking turn.

The initial fare is correct. The failure occurs later: Simia does not execute the requested booking and repeats authentication instead.

Native task result: FAIL (0)

Trajectory2Gym (Qwen3-8B)

[log 1] list_all_airports()
Tool: ["RMS", "SBK", "MPC", "SVP", ...]
get_flight_cost(travel_from="RMS", travel_to="SBK", travel_date="2026-11-15", travel_class="economy")
Tool returns $440.

...
[log 3] After the new SFO–LAX request, book_flight(...) is called.
Tool returns booking_id 3426812.
[log 5] The insurance operation reuses booking_id 3426812.
The complete native task is marked valid.

Trajectory2Gym answers the fare query, executes the later booking request, and uses the returned booking ID for insurance.

Native task result: PASS (1)

Figure 10: **A correct fare does not complete the workflow.** Both agents report the correct fare, but Simia does not execute the requested booking. Trajectory2Gym books the flight and uses its returned identifier for the subsequent insurance operation, completing the dependent workflow.

Tau2 Telecom: restore service with an overdue bill, airplane mode and broken APN settings. Success requires connectivity and no overdue bill B1234321.

Simia Tau-SFT (Qwen3-8B)

[m18 Assistant] send_payment_request(bill_id="B1234321", ...)
[m23 User] make_payment()
[m24 Tool] Records the $65 payment.
[m26 Assistant] resume_line(line_id="L1002", ...)
[m27 Tool, excerpt] "Line resumed successfully"
[m33 User] toggle_airplane_mode()
... [additional phone checks omitted]

[m53 User] check_apn_settings()
[m54 Tool, excerpt] "Current APN Name: broken"
[m57 User] reset_apn_settings()
[m58 Tool, excerpt] "APN settings will reset at reboot."
[m59 User] reboot_device()
[m60 Tool] Status bar reports 5G signal and data enabled.

Native checker: both environment assertions pass.

Simia completes payment, line resumption and device repair. Both connectivity and overdue-bill assertions pass (64 messages).

Native task result: PASS (1)

Trajectory2Gym (Qwen3-8B)

[m14 Assistant] send_payment_request(bill_id="B1234321", ...)
[m15 Tool] "Payment request sent to the customer for bill B1234321"
[m17 User] check_payment_request()
[m18 Tool] "You have a payment request for bill B1234321 of 65.0 USD."

[m19 User, excerpt] "I'll accept it. What should I do next?"
[m20 Assistant] transfer_to_human_agents(...)
Summary excerpt: "Agent lacks make_payment tool to process the payment."
[m21 Tool] "Transfer successful"
No make_payment or resume_line action follows.
Native service-status and overdue-bill checks both fail.

The trained policy stops after handoff (24 messages). Payment and repair remain incomplete; this is retained as a negative case.

Native task result: FAIL (0)

Figure 11: **Remaining failure against a successful external baseline.** Simia completes payment, line resumption, and device repair. Trajectory2Gym transfers after sending a payment request, leaving the connectivity and overdue-bill checks unsatisfied. This case retains an external-agent success and a Trajectory2Gym failure.