UNDERSTANDING CROSS-ENVIRONMENT AGENT TRANSFER THROUGH TASK SEMANTICS AND INTERACTION REALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Teacher trajectories collected from executable benchmark variants provide a scalable source of supervision for agent post-training. However, given a target benchmark, it remains unclear how source benchmark variants should be selected. Source benchmarks may resemble the target in Task Semantics (TS)—including the task goal, domain, core capabilities, and success conditions—or in Interaction Realization (IR)—including how information is accessed, actions are executed, feedback is provided, and state is managed. We investigate which type of proximity is more predictive of effective transfer. To this end, we propose TIP, an analysis framework that characterizes benchmark relationships along the TS and IR dimensions separately. We generate and filter teacher trajectories from two groups of source benchmarks, fine-tune Qwen3 models via trajectory SFT, and evaluate transfer to a target benchmark. We find that sources more closely aligned with the target in interaction realization yield larger performance gains, whereas sources with greater task-semantic similarity consistently induce negative transfer across the evaluated models. Our study develops a benchmark-relation framework beyond task-semantic similarity, clarifies when trajectories generated by one benchmark can provide useful training signals for another, and offers testable guidance for source-data selection and the analysis of agent transfer across benchmarks.

1 INTRODUCTION

Supervised fine-tuning on interaction trajectories allows agents to learn from experience collected in executable environments. AgentTuning and AgentGym develop this route to agent training across diverse tasks and environments (Zeng et al., 2023; Xi et al., 2025). As more training environments become available, practitioners must decide which experience to collect for a particular target. Collecting successful trajectories alone does not resolve this choice: behaviors demonstrated in a source environment may differ from those needed at deployment. This raises a question for agent post-training: *which source–target relationships are associated with greater transfer gains?*

A natural starting point is to select tasks with similar goals, domains, and capability requirements. Yet these similarities leave open how the agent obtains information and acts on it. Consider two tasks that ask about earlier events. One provides the complete history in the input; the other requires facts to be saved and retrieved after a context reset. Both require recovering an earlier fact, but answering the first provides no demonstration of the memory writes needed by the second. Conversely, tasks in different domains may demonstrate similar patterns of information gathering, state updates, and feedback use. Figure 1 illustrates why task resemblance and interaction resemblance need to be considered separately.

We study these relationships through *Task Semantics* (TS) and *Interaction Realization* (IR). TS describes what a task requires; IR describes how its executable environment exposes information, actions, feedback, and state. Recent work on interface sensitivity further shows that trajectory SFT can reinforce dependence on training interfaces (Gu et al., 2026). This motivates examining the interaction conditions of source experience when interpreting transfer. We develop TIP as an analysis framework that profiles source benchmark variants and the target benchmark along both dimensions and organizes their comparison.

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.

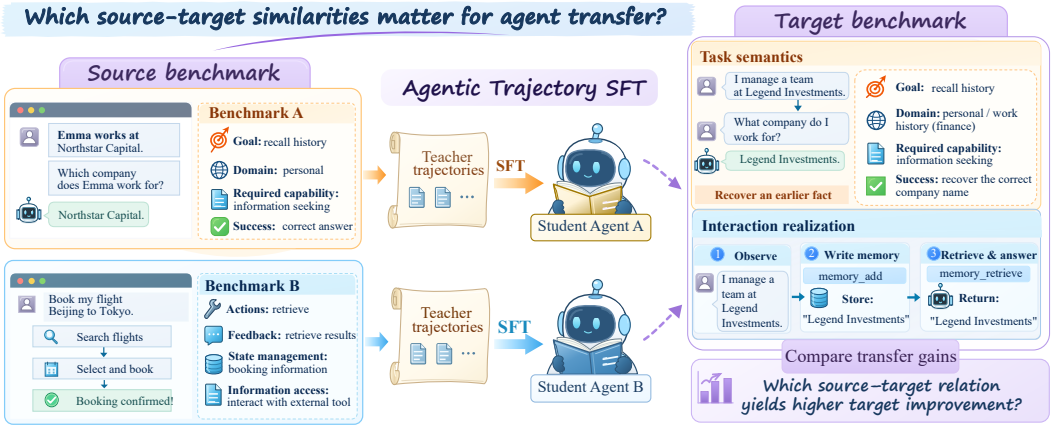


Figure 1: **Task resemblance and interaction resemblance can favor different sources.** Source A and the target both ask for an earlier fact, but A exposes it directly in context. Source B illustrates external actions and feedback in a different domain. The target additionally requires writing and accessing persistent memory. These schematic examples motivate comparing TS and IR proximity; they do not depict the experimental source-group membership.

To make the distinction observable, we compare two source groups whose matching relations favor different dimensions: one is closer to the target in TS and farther in IR, while the other has the reverse relation. A fixed teacher generates trajectories in the selected variants, and three Qwen3 students learn from the filtered trajectories through SFT. BFCL-V4 Memory provides the target, where facts must remain usable across a context reset. We measure transfer as the change from each student's original target performance and inspect the resulting interactions.

IR-SFT outperforms TS-SFT across all three students and memory interfaces. It improves two students over their baselines and produces a smaller decline on the third; TS-SFT produces negative transfer throughout. Cases locate differences in memory writing, retrieval, and answer completion, including the loss of previously successful behavior. The comparison concerns these pooled source datasets under a shared training recipe; their data exposure differs, so the results establish an association rather than an isolated causal effect of IR proximity.

Our contributions are threefold:

- We characterize source–target relations along task semantics and interaction realization, and use TIP to construct a transfer comparison in which the two dimensions favor different source groups.
- We find that the group closer in interaction realization yields higher transfer gains across three student models and three memory interfaces, while gains relative to the baseline vary across models.
- We analyze target interactions to connect the transfer results to memory construction, evidence use, and the loss of previously successful behavior, informing source-benchmark choice in agent post-training.

2 Related Work

Learning from agent trajectories. ReAct interleaves reasoning and actions in an environment (Yao et al., 2023). AgentTuning, AgentBank, and AgentGym use interaction trajectories to train agents across tasks and environments (Zeng et al., 2023; Song et al., 2024; Xi et al., 2025). More recent work broadens the experience available for training. AgentGym-RL studies multi-turn RL with progressively longer interaction horizons (Xi et al., 2026); CodeGym constructs tool-use environments from coding problems and evaluates transfer to unseen workflows (Du et al., 2026). Agent Learning via Early Experience uses agents' own actions and subsequent states for supervision without requiring rewards (Zhang et al., 2026). These works develop training resources or learning procedures. We keep

trajectory SFT as the learning procedure and examine which source–target relationships accompany its transfer gains.

Transfer relations and training-data selection. Taskonomy and TaskWeb study transfer relations among visual and language tasks (Zamir et al., 2018; Kim et al., 2023). LESS uses gradient features for targeted instruction-data selection, Deita combines complexity, quality, and diversity, and DoReMi learns domain mixtures using a proxy model (Xia et al., 2024; Liu et al., 2024; Xie et al., 2023). For agents, ScaleEnv synthesizes executable tool-use environments and examines how domain diversity affects generalization (Tu et al., 2026). Ability-aware Environment Selection considers ability coverage, redundancy, and gradient conflict when composing environments (Zhu et al., 2026). TIP instead describes task requirements and interaction conditions separately, enabling a comparison where the two dimensions favor different source groups. It supplies a representation for analyzing this choice, rather than an optimized data-mixture policy.

Interaction design and agent behavior. SWE-agent demonstrates the role of the agent–computer interface in task performance (Yang et al., 2024). BFCL, τ-bench, and ToolSandbox provide tool-use and stateful interaction settings (Mao et al., 2025; 2024; Yao et al., 2024; Lu et al., 2025). Closely related to our question, PIPE diagnoses interface reliance through semantics-preserving interface rewrites and finds that trajectory SFT can amplify interface shortcutting (Gu et al., 2026). PIPE varies the interface while preserving execution behavior. Our comparison uses existing source variants with different information access, state management, and action–feedback conditions, then measures transfer to a fixed target protocol. These perspectives are complementary: a favorable transfer gain does not by itself establish robustness to interface changes.

Memory in interactive agents. MemoryAgentBench evaluates incremental memory through multi-turn interactions, including retrieval, learning, understanding, and forgetting (Hu et al., 2026). AMemGym provides interactive conversations with structured state evolution for evaluating memory-driven personalization (Cheng et al., 2026). Both emphasize properties obscured by static long-context question answering. BFCL-V4 Memory similarly exposes memory construction and later querying as agent interactions (Mao et al., 2025). We use this target to study transfer from source trajectories, examining whether earlier facts are preserved, retrieved, and used to complete an answer.

3 Preliminaries and Problem Formulation

We study cross-environment agent transfer through SFT on teacher-generated interaction trajectories. A benchmark family may provide several benchmark variants, each defined by a concrete task subset and execution configuration. When a variant contains materially different task or interaction properties, TIP represents the corresponding portion as a benchmark unit. A source group pools selected variants into one training dataset, while the target benchmark supplies the tasks and evaluation protocol used to measure transfer.

3.1 Agent Interaction and Trajectories

For a task instance q in benchmark unit u, an agent with policy π_θ produces an action $a_j \sim \pi_\theta(\cdot \mid h_j)$ from its visible context h_j. Actions include text responses and tool calls; the environment returns observations that inform subsequent decisions. A *rollout* is the execution process, and its recorded interaction *trajectory* is

$$\tau = (q, o_0, a_0, o_1, a_1, \ldots, o_{L-1}, a_{L-1}, o_L), \tag{1}$$

where o_j is an observation and L is the number of decisions before termination. The visible context depends on the environment's treatment of history, retrieval, and state.

3.2 Supervised Fine-Tuning from Teacher Trajectories

A fixed teacher policy π^{T} generates trajectories through $\tau_i^{(u)} \sim \mathrm{Rollout}(\pi^{\mathrm{T}}, u, q_i)$ for source unit u. Filtering and format conversion produce the unit-level training set $\mathcal{D}_u$. Each example contains

teacher outputs and their visible interaction context. A token mask $m_{z,\ell}$ selects teacher outputs for supervision, giving the SFT objective

$$\mathcal{L}_{\mathrm{SFT}}(\theta; \mathcal{D}_u) = -\frac{1}{Z_u} \sum_{z \in \mathcal{D}_u} \sum_{\ell=1}^{|z|} m_{z,\ell} \log \pi_\theta(z_\ell \mid z_{<\ell}), \tag{2}$$

where $Z_u = \sum_z \sum_\ell m_{z,\ell}$ counts supervised tokens. For source group g, the pooled dataset is $\mathcal{D}_g$ formed from the retained examples of its benchmark units. The student starts from parameters θ_0 and is fine-tuned under configuration λ, yielding $\theta_g = \mathrm{SFT}(\theta_0, \mathcal{D}_g; \lambda)$ and policy π_g. Training settings and resulting data exposure are specified in Section 5.

3.3 Transfer Gain and Research Question

For target t, let $\mathcal{Q}_t$ be the evaluation instances and $R_t(q, \tau)$ the score of an execution. Target performance is the mean expected score, $\mathrm{Perf}_t(\pi) = |\mathcal{Q}_t|^{-1} \sum_{q \in \mathcal{Q}_t} \mathbb{E}_{\tau \sim \mathrm{Rollout}(\pi, t, q)}[R_t(q, \tau)]$. We measure transfer gain relative to the same student's initial policy π_0:

$$\Delta_{g \to t} = \mathrm{Perf}_t(\pi_g) - \mathrm{Perf}_t(\pi_0). \tag{3}$$

Positive and negative values indicate improvement and degradation, respectively. Both policies use the same target evaluation protocol. We ask how task-semantic and interaction-realization proximity between source benchmark units and a target benchmark relate to this gain, comparing the two matching relations through target performance and observed student behavior.

4 TIP: Task–Interaction Profiling Framework

TIP describes source benchmark variants and the target benchmark along Task Semantics (TS) and Interaction Realization (IR). The profiles establish the comparison dimensions and guide source-group construction; transfer gains are measured after trajectory collection and student training. Figure 2 presents the workflow.

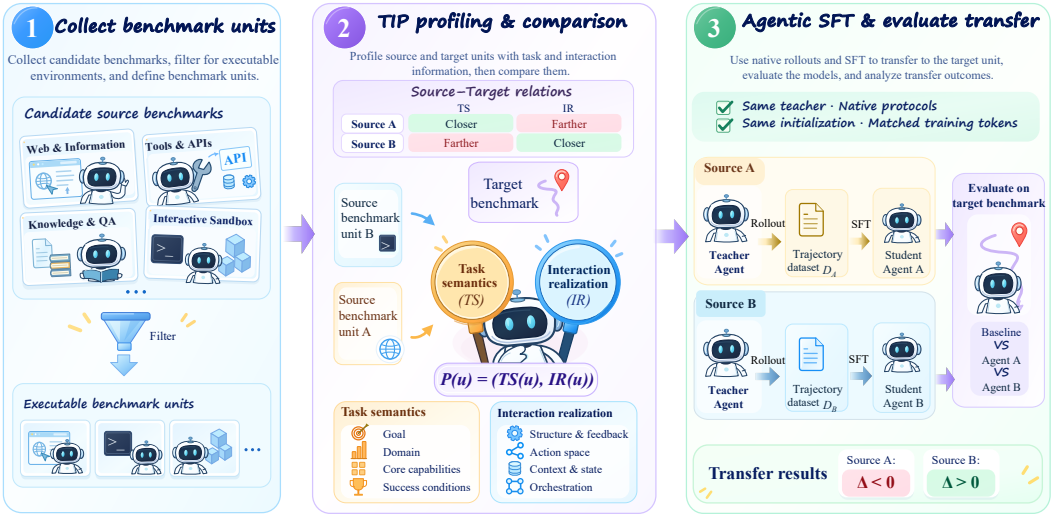


Figure 2: **TIP and the transfer-study workflow.** Left: collect executable benchmark variants and define units with consistent task and interaction properties. Middle: profile TS and IR separately to organize the source groups. Right: collect native teacher trajectories, fine-tune from each student's original checkpoint, and measure target gains. The groups share a training recipe but differ in data exposure; higher IR-SFT gains do not imply positive transfer for every student.

4.1 Candidate Benchmark Collection and Screening

We collect candidate benchmarks from arXiv, Hugging Face, and GitHub, linking papers, official repositories, task data, and runtime documentation. Candidates provide clear task definitions,

executable instances, an action–feedback loop, and a scorer or explicit success criterion. An investigator reviews the documentation and implementation to establish task scope, interaction mode, and execution requirements. This process produces a collection of benchmark profiles.

4.2 Evidence-Grounded Task–Interaction Profiles

A benchmark variant may contain multiple task subsets or interaction modes. We describe each portion with consistent task and interaction properties as a *benchmark unit*, tied to its data, code revision, and execution configuration. For unit u, the profile is

$$P(u) = \big(P_{\mathrm{TS}}(u), P_{\mathrm{IR}}(u)\big). \tag{4}$$

TS captures task goals, domains, core capability requirements, and success conditions. IR captures information access, available actions, feedback, state and context management, and execution control. Table 1 summarizes the fields. For example, repairing a code defect requires reasoning about the program and satisfying its tests; available editing tools, test feedback, and retry control describe how that task is realized.

Profiles are grounded in task specifications and executable implementations. We inspect the runtime from task loading through action execution, feedback, and termination, and use an interaction graph to organize these relationships. Field values follow shared definitions and controlled vocabularies, with documentation or code evidence supporting each assignment. Core capability requirements describe what the task demands. A tool's availability does not make its use a core capability: an action belongs to TS only when it is an indispensable task requirement. The mechanism exposing the action belongs to IR. Similarly, a successful outcome belongs to TS, while feedback timing and termination control belong to IR. These boundaries prevent interface details from being counted in both dimensions.

4.3 Comparing Task and Interaction Relations

For a source unit u and target unit t, TIP separately computes task-semantics distance $D_{\mathrm{TS}}(u,t)$ and interaction-realization distance $D_{\mathrm{IR}}(u,t)$. Categorical attributes use an indicator distance, while set-valued attributes use Jaccard distance:

$$d_k(u,t) = \begin{cases} \mathbf{1}[v_k(u) \neq v_k(t)], & k \text{ categorical}, \\ 1 - \dfrac{|v_k(u) \cap v_k(t)|}{|v_k(u) \cup v_k(t)|}, & k \text{ set-valued}. \end{cases} \tag{5}$$

We average comparable attributes within fields, then average fields within each dimension:

$$D_X(u,t) = \frac{1}{|F_X|} \sum_{f \in F_X} \frac{1}{|A_f|} \sum_{k \in A_f} d_k(u,t), \qquad X \in \{\mathrm{TS}, \mathrm{IR}\}, \tag{6}$$

where A_f and F_X denote the comparable attributes and fields for the pair (u,t). Participating fields receive equal weight, and lower distances indicate closer profiles. The distances describe profile proximity, not a calibrated prediction of transfer gain.

The two distances provide separate views of the source–target relationship. We use them to organize two source groups: one closer in task semantics and farther in interaction realization, and the other with the reverse relation. Their benchmark-family composition is given in Section 5; Appendix A.4 lists the source variants and the shared profile fields.

4.4 Teacher Trajectories and Training-Data Construction

Teacher rollouts follow each selected variant's native tools, feedback, and execution protocol. Successful, valid trajectories are converted into SFT examples while preserving the context visible at each decision. Teacher outputs supply supervision; task inputs and tool observations supply context. Section 5 specifies the teacher, filtering, and shared training recipe.

5 Experiments

We compare the transfer gains from two source groups on the BFCL-V4 Memory target benchmark, using three student models and a shared SFT recipe.

Table 1: Task–interaction profile fields used to describe source benchmark variants and the target benchmark.

Field	Description
Task semantics	
Task goal	The result the agent must achieve
Task domain	The knowledge domain and principal objects
Core capability requirements	Capabilities needed to complete the task
Success conditions	The requirements for a successful outcome
Interaction realization	
Interaction structure and feedback	Information flow; feedback timing and visibility
Action space	Available operations and their effects
Context and state management	Access to history and persistence of state
System orchestration	Control of actions, phases, and termination

5.1 Target Benchmark and Group Comparison

BFCL-V4 Memory evaluates questions about facts from earlier interactions through three memory interfaces: key–value (KV), vector (Vector), and recursive summarization (RecSum). The agent constructs memory during prior conversations and accesses it after a context reset to answer later queries. The target benchmark therefore combines historical-information questions with requirements for memory construction, state persistence, and retrieval.

Using TIP, we form a *TS group* closer to the target in task semantics and farther in interaction realization, and an *IR group* with the reverse relation. Table 2 lists the eight benchmark families; Appendix A.1 gives their twelve variants and retained example counts.

5.2 Teacher Trajectories and Student Fine-Tuning

We use DeepSeek-V4-Flash (DeepSeek-AI, 2026) as the teacher with the same generation settings across source variants. Successful, valid trajectories are converted into SFT examples and filtered for context length and template compatibility, yielding 3,145 TS and 4,217 IR examples.

We fine-tune Qwen3-30B-A3B-Instruct-2507, Qwen3-8B, and Qwen3-4B-Instruct-2507. Each condition starts independently from the corresponding original checkpoint, which also serves as its baseline. Training on the TS and IR groups yields TS-SFT and IR-SFT, respectively.

Within each student model, both conditions use one epoch, global batch size 8, peak learning rate 10^{-5} with cosine decay, and a 131,072-token training window. Each condition uses one training seed, and we evaluate its final checkpoint.

Evaluation protocol. Every checkpoint is evaluated three times under the same BFCL revision, with temperature 0.001 and a 131,072-token inference-service window. Each repeat reruns memory construction and querying, with 155 scored instances per interface (465 in total). We report mean accuracy across repeats and interfaces, and transfer gain over the corresponding baseline in percentage points (pp). Sample standard deviations describe evaluation variability for each fixed checkpoint, not variability across training seeds. Since each checkpoint constructs its own memory, scores reflect both memory construction and subsequent querying. Appendix A.2 provides the protocol and repeat scores.

6 Results and Analysis

We compare transfer gains across the two source groups, then examine how the outcomes vary across models and appear in target interactions.

Table 2: Source groups and their relative TS/IR matching relations to BFCL-V4 Memory. These labels describe the group comparison, not identical properties of every constituent variant. Appendix A.1 lists the twelve variants and retained example counts.

Source group	Benchmark families	Relation to target
TS	MemGround, GateMem, StructMemEval, AMemGym	Closer in task semantics; farther in interaction realization
IR	VehicleMemBench, ToolMisuseBench, ToolMaze, TravelBench	Farther in task semantics; closer in interaction realization

6.1 IR-SFT Yields Higher Transfer Gains

IR-SFT outperforms TS-SFT on every student model. Table 3 reports target accuracy and gains relative to each model's baseline. The mean-accuracy advantage of IR-SFT over TS-SFT is 15.70–16.56 pp across the three models. The ordering holds separately for KV, Vector, and RecSum.

The group closer in task semantics produces negative transfer on all three models. The group closer in interaction realization improves two models and produces a smaller decline on the third. Thus, similarity in how tasks are carried out corresponds to greater transfer value in this conflicting-match comparison, even when the source tasks are farther from the target in their semantics. Within each student, every IR-SFT evaluation repeat also exceeds every TS-SFT repeat in mean accuracy (Appendix A.2). The aggregate ordering is therefore not driven by a single favorable evaluation run.

Table 3: **Target transfer on BFCL-V4 Memory.** Mean accuracy over three evaluation repeats of each checkpoint. Gains are relative to the same model's original checkpoint.

	Accuracy by memory interface (%) ↑			**Target performance**	
Training condition	Key–value interface	Vector interface	Recursive summarization interface	Mean accuracy (%) ↑	Gain over baseline (pp)
Qwen3-30B-A3B-Instruct-2507					
Baseline	8.17	13.55	28.82	16.85	–
TS-SFT	5.81	3.87	18.49	9.39	−7.46
IR-SFT	**20.22**	**26.88**	**29.46**	**25.52**	**+8.67**
Qwen3-8B					
Baseline	7.74	18.06	28.39	18.06	–
TS-SFT	7.53	10.97	6.67	8.39	−9.68
IR-SFT	**14.62**	**28.60**	**31.61**	**24.95**	**+6.88**
Qwen3-4B-Instruct-2507					
Baseline	**16.13**	13.98	**33.55**	**21.22**	–
TS-SFT	1.94	1.94	3.01	2.29	−18.92
IR-SFT	15.91	**15.48**	22.58	17.99	−3.23

Each interface contains 155 questions. Mean accuracy equally averages the interfaces; gains use unrounded scores in percentage points (pp). TS-SFT and IR-SFT use the groups in Table 2, with 3,145 and 4,217 retained examples, respectively, and one training epoch. Bold marks the highest accuracy in each column within a model.

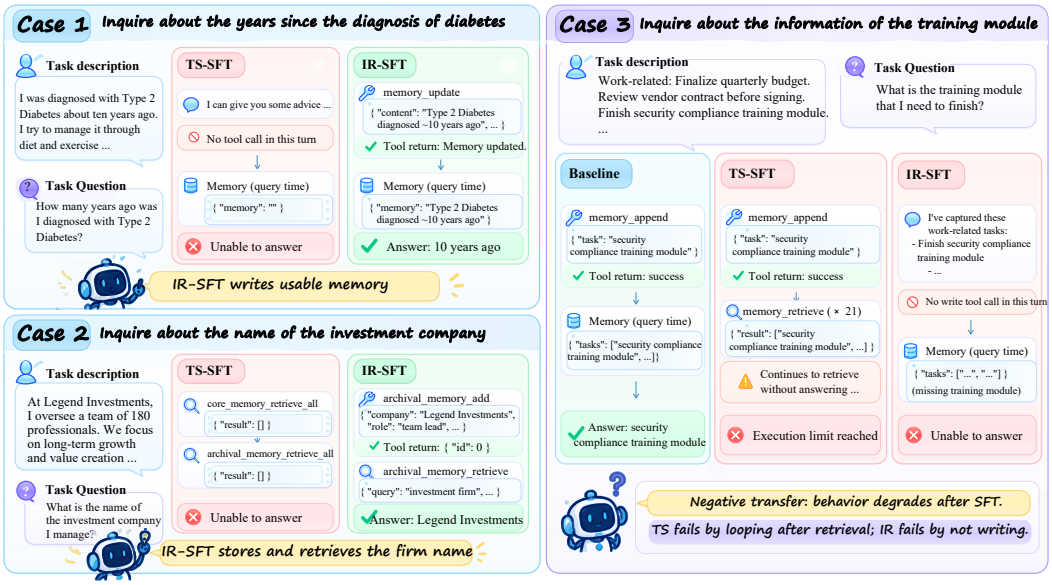


Figure 3: **Memory construction and query-time behavior on three target questions.** Cases 1 (RecSum) and 2 (Vector) show Qwen3-30B-A3B-Instruct-2507 in repeat 2: IR-SFT preserves the needed facts, whereas TS-SFT lacks usable memory. Case 3 shows Qwen3-4B-Instruct-2507 in repeat 1: the baseline succeeds, TS-SFT retrieves repeatedly without answering, and IR-SFT omits the write. Messages and tool payloads are abbreviated; the panels combine construction and querying across the context reset.

6.2 Transfer Gains Vary Across Student Models

IR-SFT produces gains of +8.67 pp, +6.88 pp, and −3.23 pp on Qwen3-30B-A3B-Instruct-2507, Qwen3-8B, and Qwen3-4B-Instruct-2507, respectively. On the 4B model, it reduces mean accuracy from 21.22% to 17.99%, while TS-SFT reduces it to 2.29%. IR-SFT remains above TS-SFT for every student, although its net transfer changes sign.

The 4B decline is concentrated in RecSum: IR-SFT falls 10.97 pp relative to the baseline, while KV falls 0.22 pp and Vector increases 1.51 pp. These differences motivate examining which interaction behaviors improve and which previously successful behaviors are lost after fine-tuning.

6.3 Analysis of Transfer Behavior

Figure 3 follows three questions from the prior conversation through memory construction and later querying. Cases 1 and 2 compare TS-SFT and IR-SFT on Qwen3-30B-A3B-Instruct-2507; Case 3 examines a previously successful behavior lost after fine-tuning Qwen3-4B-Instruct-2507.

Together, the cases move from a clear gain in persistent memory to a more specific failure mode: an agent may retrieve useful evidence yet still fail to complete the task, or may never write the needed state at all. This progression helps separate the interaction behaviors associated with the aggregate transfer results.

Turning Earlier Facts into Usable Memory. In Cases 1 and 2, IR-SFT saves the facts needed after the context reset for Qwen3-30B-A3B-Instruct-2507. TS-SFT answers both questions incorrectly in all three repeats, whereas IR-SFT answers both correctly throughout.

In Case 1 (RecSum), the user reports a Type 2 Diabetes diagnosis about ten years earlier. TS-SFT responds about diet and lifestyle without writing memory, leaving an empty snapshot before the reset. IR-SFT calls `memory_update` to save the diagnosis and interval; this information survives subsequent updates and supports the later answer.

In Case 2 (Vector), the user mentions managing 180 people at Legend Investments. TS-SFT does not save the company name, and its later `core_memory_retrieve_all` and `archival_memory_retrieve_all` calls return empty lists. IR-SFT writes the fact through `archival_memory_add`, retrieves it through `archival_memory_retrieve`, and answers correctly. Both cases locate the difference in memory construction: a relevant conversational response does not ensure that information remains available after the reset.

Loss of Previously Successful Behavior on the 4B Model. In Case 3, the user requests recording "Finish security compliance training module" and later asks which training remains. The baseline writes the module through `memory_append` and answers correctly in all three repeats; both SFT conditions fail throughout, at different stages.

TS-SFT saves the training content and retrieves evidence containing "security compliance," but continues calling `memory_retrieve` until the execution-step limit. The displayed repeat contains 21 retrieval calls and no final answer. IR-SFT only retrieves during the prior conversation and performs no write, despite claiming to have recorded the tasks. The module is absent from persistent memory, and its later answer is incorrect.

These cases distinguish missing state updates from failure to complete an answer after obtaining evidence. The broader 8B RecSum diagnostic reinforces that tool activity alone is insufficient: IR-SFT uses query tools in 461 of 465 executions, yet 105 end without a final answer (Appendix A.3). The traces illustrate behaviors accompanying transfer; they do not establish their prevalence across all target questions.

7 Limitations

The comparison uses two pooled source groups and a single Memory target with three interfaces. Source-group identity changes benchmark composition and data exposure together with TS/IR proximity, so the results do not isolate the causal contribution of either dimension. A comparison with matched supervised-token budgets and additional source groups would help separate these factors.

Each condition uses one training seed. Repeated target evaluations characterize fixed-checkpoint variability, but do not establish robustness to training randomness. The three selected cases illustrate where behaviors differ; they do not estimate how frequently each failure mechanism occurs. TIP also depends on evidence-grounded field assignments and equal-weight distance aggregation, whose sensitivity has not been measured here. Broader targets, profile audits, and retention analysis would test the scope of the observed ordering.

Pooling also prevents attributing gains to individual source variants: the study does not determine whether all constituents help or a subset dominates. Per-variant ablations would be needed before using these group-level findings as a source-ranking rule.

8 Conclusion

We study how task-semantic and interaction-realization proximity relate to cross-environment agent transfer through trajectory SFT. TIP makes these relations explicit and organizes the source-group comparison. On BFCL-V4 Memory, the group closer in interaction realization yields higher gains than the group closer in task semantics across all three student models, with positive transfer on two and reduced negative transfer on the third. Target traces locate the observed differences in memory writing, retrieval, and answer completion, including regressions on previously successful questions. These results support comparing source benchmark variants by their interaction conditions alongside their task semantics. For source selection, they motivate checking whether successful trajectories demonstrate the state updates and evidence use required by the target, and measuring retention alongside aggregate gains.

Reproducibility Statement

Section 4 describes the TS/IR profiles, source-group comparison procedure, and trajectory-to-training workflow. Appendix A provides benchmark membership, retained examples, training and evaluation settings, repeat scores, and the profiling prompt. Figure 3 identifies the displayed cases by their questions, student models, and evaluation repeats. We will release the code for profiling, trajectory processing, training, and evaluation, together with the profile assignments and supporting evidence needed to reproduce the source–target comparisons.

AI Use Statement

LLM agents assist benchmark profiling and generate training trajectories. Generative AI tools also assisted research design, literature lookup, drafting, analysis of experiment records, case studies, plots, and LaTeX preparation. Reported scores and case facts were checked against evaluation and trace records. The authors are responsible for the final manuscript.

Ethics Statement

The study uses benchmark environments and teacher-generated interaction trajectories. The reported conversations are benchmark cases. Relevant release considerations include benchmark licenses, redistribution permissions, and privacy in conversation and memory records.

References

Jiayang Cheng, Dongyu Ru, Lin Qiu, Yiyang Li, Xuezhi Cao, Yangqiu Song, and Xunliang Cai. AMemGym: Interactive Memory Benchmarking for Assistants in Long-Horizon Conversations. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=sfrVLzsmlf.

DeepSeek-AI. DeepSeek-V4: Towards Highly Efficient Million-Token Context Intelligence, 2026. URL https://arxiv.org/abs/2606.19348.

Weihua Du, Hailei Gong, Zhan Ling, Kang Liu, Lingfeng Shen, Xuesong Yao, Yufei Xu, Dingyuan Shi, Yiming Yang, and Jiecao Chen. Generalizable End-to-End Tool-Use RL with Synthetic CodeGym. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=QRSeFZfu8E.

Weizheng Gu, Chengze Li, Zhuohao Yu, Mengyuan Sun, Zhibang Yang, Wei Wang, Hongrui Jia, Shikun Zhang, and Wei Ye. What Do Agents Learn from Trajectory-SFT: Semantics or Interfaces? In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=u1i3XoSXQG.

Yuanzhe Hu, Yu Wang, and Julian McAuley. Evaluating Memory in LLM Agents via Incremental Multi-Turn Interactions. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=DT7JyQC3MR.

Joongwon Kim, Akari Asai, Gabriel Ilharco, and Hannaneh Hajishirzi. TaskWeb: Selecting Better Source Tasks for Multi-task NLP. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 11032–11052. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.680. URL https://aclanthology.org/2023.emnlp-main.680/.

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 *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2312.15685.

Jiarui Lu, Thomas Holleis, Yizhe Zhang, Bernhard Aumayer, Feng Nan, Felix 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. arXiv preprint arXiv:2408.04682, 2025. URL https://arxiv.org/abs/2408.04682.

Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Jason Huang, Vishnu Suresh, Yixin Huang, Xiaowen Yu, Joseph E. Gonzalez, and Shishir G. Patil. BFCL V3: Multi-Turn and Multi-Step Function Calling Evaluation. Official Berkeley Function-Calling Leaderboard documentation, 2024. URL https://gorilla.cs.berkeley.edu/blogs/13_bfcl_v3_multi_turn.html. Released September 19, 2024; updated December 10, 2024. Accessed September 10, 2026.

Huanzhi Mao, Amitoj Singh, Pracheeti Shikarkhane, Kai Wen, Shishir G. Patil, and Joseph E. Gonzalez. BFCL V4 Agentic, Part 2: Evaluating Tool Calling for Memory. Official Berkeley Function-Calling Leaderboard documentation, 2025. URL https://gorilla.cs.berkeley.edu/blogs/16_bfcl_v4_memory.html. Released July 17, 2025. Accessed September 10, 2026.

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 *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 2124–2141. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-emnlp.116. URL https://aclanthology.org/2024.findings-emnlp.116/.

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 *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=4GX4r1CYtD.

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, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1355. URL https://aclanthology.org/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 *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=ZgCCDwcGwn.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: Selecting Influential Data for Targeted Instruction Tuning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54104–54132. PMLR, 2024. URL https://proceedings.mlr.press/v235/xia24c.html.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V. Le, Tengyu Ma, and Adams Wei Yu. DoReMi: Optimizing Data Mixtures Speeds Up Language Model Pretraining. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://papers.neurips.cc/paper_files/paper/2023/hash/dcba6be91359358c2355cd920da3fcbd-Abstract-Conference.html.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://arxiv.org/abs/2405.15793.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing Reasoning and Acting in Language Models. arXiv preprint arXiv:2210.03629, 2023. URL https://arxiv.org/abs/2210.03629.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains. arXiv preprint arXiv:2406.12045, 2024. URL https://arxiv.org/abs/2406.12045.

Amir Zamir, Alexander Sax, William Shen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling Task Transfer Learning. arXiv preprint arXiv:1804.08328, 2018. URL https://arxiv.org/abs/1804.08328.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. AgentTuning: Enabling Generalized Agent Abilities for LLMs. arXiv preprint arXiv:2310.12823, 2023. URL https://arxiv.org/abs/2310.12823.

Kai Zhang, Xiangchao Chen, Bo Liu, Tianci Xue, Zeyi Liao, Zhihan Liu, Xiyao Wang, Yuting Ning, Zhaorun Chen, Xiaohan Fu, Jian Xie, Yuxuan Sun, Boyu Gou, Qi Qi, Zihang Meng, Jianwei Yang, Ning Zhang, Xian Li, Ashish Shah, Dat Huynh, Hengduo Li, Zi Yang, Xuefei Cao, Lawrence Jang, Shuyan Zhou, Jiacheng Zhu, Huan Sun, Jason Weston, Yu Su, and Yifan Wu. Agent Learning via Early Experience. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=N3dXUHY5dD.

Kejian Zhu, Zhuoran Jin, Dongqi Huang, Hongbang Yuan, Yupu Hao, Kang Liu, and Jun Zhao. Beyond Simply Environment Scaling: Designing Effective Environment Distributions for Multimodal Agent Learning. arXiv preprint arXiv:2608.03571, 2026. URL https://arxiv.org/abs/2608.03571.

A Experimental Details and Additional Results

This appendix provides training data, configurations, repeated evaluations, and supporting behavioral diagnostics. The abbreviations 4B, 8B, and 30B refer to Qwen3-4B-Instruct-2507, Qwen3-8B, and Qwen3-30B-A3B-Instruct-2507, respectively.

A.1 Training Data and Configuration

Table 4 lists the twelve benchmark variants in the two source groups. Successful trajectories from each source variant are converted into examples and filtered for context length and template compatibility. The retained datasets contain 3,145 TS examples and 4,217 IR examples.

Table 4: Retained training examples by benchmark variant. Retained counts follow context-length and template filtering.

Benchmark family / variant	Source group	Successful examples	Retained examples
memground/trpg_success	TS	146	144
gatemem/long_context	TS	1164	1164
structmemeval/mem_agent	TS	254	254
amemgym/native	TS	766	726
amemgym/awi	TS	857	857
vehiclemembench/none	IR	84	84
vehiclemembench/gold	IR	390	388
vehiclemembench/key_value	IR	246	246
vehiclemembench/summary	IR	264	264
toolmisusebench/api_model_train	IR	1519	1519
toolmaze/legacy_full	IR	937	926
travelbench/legacy_full	IR	803	790

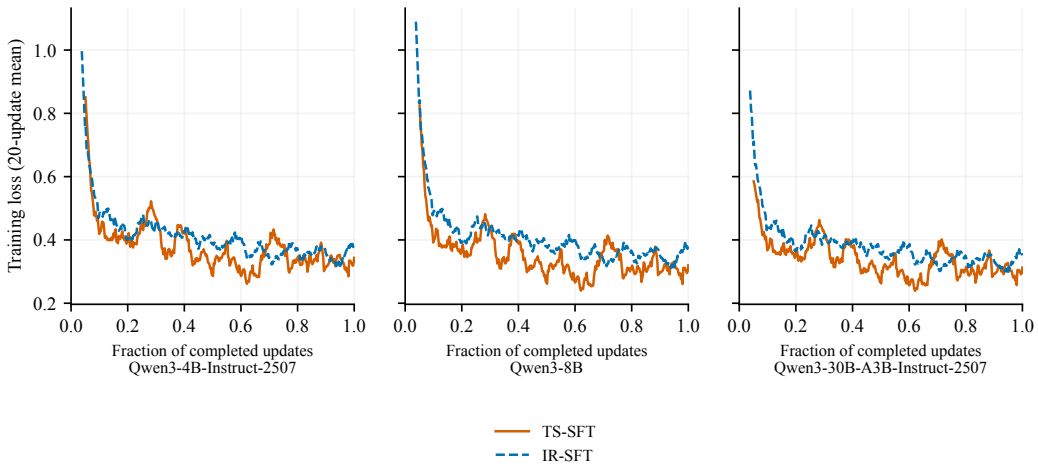


Figure 4: Teacher-output training loss for TS-SFT and IR-SFT on each student, smoothed over 20 updates. The horizontal axis normalizes each run by its own completed updates; it does not equate update counts or supervised-token exposure across source groups.

Figure 4 shows the training-loss curves. All six fine-tuning runs use one epoch, global batch size 8, Adam with $(\beta_1, \beta_2) = (0.9, 0.95)$, weight decay 0.1, gradient clipping 0.1, and seed 1234. The learning rate follows cosine decay from 10^{-5} to 10^{-6} after 5% warmup. The training window is 131,072 tokens. Qwen3-4B-Instruct-2507 and Qwen3-8B use tensor parallelism 2 and context parallelism 4 on eight H200 GPUs. Qwen3-30B-A3B-Instruct-2507 uses tensor parallelism 4, context parallelism 4, and expert parallelism 8 on sixteen H200 GPUs. The 8B model uses YaRN with factor 4; the two Instruct-2507 models use native RoPE.

A.2 Evaluation Protocol and Complete Results

Evaluation uses BFCL revision `6ea57973c7a6097fd7c5915698c54c17c5b1b6c8`, temperature 0.001, 64 workers, and a 131,072-token inference-service window. Each repeat runs 111 prerequisite conversation entries to construct memory, followed by 465 questions: 155 each for key–value, vector, and recursive-summarization memory. Each query uses the memory constructed by the evaluated checkpoint during the same repeat. The official scorer determines correctness, with unsuccessful executions counted as incorrect.

For each checkpoint, we equally average interface accuracies. Tables 5 and 6 give means with sample standard deviations and individual repeats. Figure 5 plots mean accuracy with evaluation standard deviations. The three 30B baseline repeats follow the same protocol as the fine-tuned checkpoints. Gains are computed from unrounded scores.

Table 5: Memory interface accuracy (%) over three evaluation repeats. Each interface has 155 scored instances per repeat; SD describes evaluation variability for fixed checkpoints.

Model	Condition	KV	Vector	RecSum
4B	Baseline	16.13 ± 1.94	13.98 ± 0.37	33.55 ± 1.71
4B	TS-SFT	1.94 ± 0.00	1.94 ± 0.65	3.01 ± 1.34
4B	IR-SFT	15.91 ± 4.20	15.48 ± 2.33	22.58 ± 7.44
8B	Baseline	7.74 ± 1.29	18.06 ± 4.03	28.39 ± 1.94
8B	TS-SFT	7.53 ± 0.99	10.97 ± 0.00	6.67 ± 1.97
8B	IR-SFT	14.62 ± 4.30	28.60 ± 0.37	31.61 ± 2.33
30B	Baseline	8.17 ± 2.27	13.55 ± 2.96	28.82 ± 8.01
30B	TS-SFT	5.81 ± 1.29	3.87 ± 2.81	18.49 ± 6.72
30B	IR-SFT	20.22 ± 1.97	26.88 ± 5.41	29.46 ± 3.67

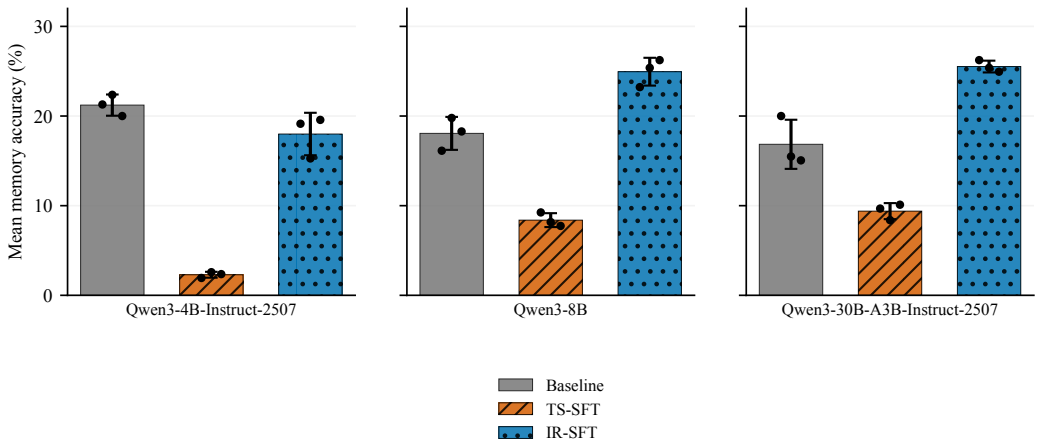


Figure 5: BFCL-V4 Memory accuracy for the baseline, TS-SFT, and IR-SFT on each student. Bars show the mean of the equally averaged interface accuracies across three evaluation repeats; dots show individual repeats, and error bars show their sample standard deviation. They describe fixed-checkpoint evaluation variability.

Table 6: Individual evaluation repeats of each fixed checkpoint on BFCL-V4 Memory. Each repeat reruns memory construction and scores 155 instances per interface (465 total). Mean accuracy equally averages the three interfaces; all scores are percentages.

Model	Condition	Repeat	Key–value interface	Vector interface	Recursive summarization interface	Mean accuracy
4B	Baseline	1	16.13	13.55	34.19	21.29
4B	Baseline	2	18.06	14.19	34.84	22.37
4B	Baseline	3	14.19	14.19	31.61	20.00
4B	TS-SFT	1	1.94	1.94	1.94	1.94
4B	TS-SFT	2	1.94	1.29	4.52	2.58
4B	TS-SFT	3	1.94	2.58	2.58	2.37
4B	IR-SFT	1	20.00	17.42	20.00	19.14
4B	IR-SFT	2	16.13	12.90	16.77	15.27
4B	IR-SFT	3	11.61	16.13	30.97	19.57
8B	Baseline	1	6.45	13.55	28.39	16.13
8B	Baseline	2	7.74	21.29	30.32	19.78
8B	Baseline	3	9.03	19.35	26.45	18.28
8B	TS-SFT	1	8.39	10.97	8.39	9.25
8B	TS-SFT	2	6.45	10.97	7.10	8.17
8B	TS-SFT	3	7.74	10.97	4.52	7.74
8B	IR-SFT	1	10.97	29.03	29.68	23.23
8B	IR-SFT	2	13.55	28.39	34.19	25.38
8B	IR-SFT	3	19.35	28.39	30.97	26.24
30B	Baseline	1	5.81	16.13	38.06	20.00
30B	Baseline	2	8.39	14.19	23.87	15.48
30B	Baseline	3	10.32	10.32	24.52	15.05
30B	TS-SFT	1	5.81	2.58	20.65	9.68
30B	TS-SFT	2	7.10	7.10	10.97	8.39
30B	TS-SFT	3	4.52	1.94	23.87	10.11
30B	IR-SFT	1	20.65	29.68	28.39	26.24
30B	IR-SFT	2	21.94	20.65	33.55	25.38
30B	IR-SFT	3	18.06	30.32	26.45	24.95

A.3 Behavioral Diagnostics and Retention

Table 7 reports query behavior for Qwen3-8B on recursive-summarization memory. IR-SFT uses tools on 461 of 465 query executions, including 105 executions that end without a final answer. Its frequent retrieval is accompanied by both higher accuracy and incomplete executions. The baseline uses query tools on 29 executions and TS-SFT on none.

Table 7: Query behavior of Qwen3-8B on RecSum across 155 questions and three evaluation repeats per condition (465 executions). Columns count executions with at least one query-tool call and executions without a final answer; these categories can overlap.

Condition	Uses query tools / 465	No final answer / 465
Baseline	29	0
TS-SFT	0	0
IR-SFT	461	105

A.4 Profile Specification for Source-Group Construction

TIP profiles each benchmark unit along the same task-semantics and interaction-realization fields used in the main text. A benchmark family may provide several variants, and a variant is split into benchmark units only when its task or interaction properties differ materially. The source groups then pool the selected variants for trajectory collection and SFT.

Shared profile fields. The task-semantics fields record the task goal, task domain, core capability requirements, and success conditions. The interaction-realization fields record interaction structure and feedback, action space, context and state management, and system orchestration. These fields are assigned from the task specification and the executable interaction protocol. They are used to organize the two source groups described in Table 2; this appendix does not introduce a separate profile vocabulary or a second set of group labels.

Benchmark-tagging prompt. The following text is the prompt template used with an agent to label each benchmark. Bracketed fields are filled with the corresponding benchmark materials; the instructions and requested fields are kept fixed across benchmarks.

> **Benchmark to profile:** `<BENCHMARK_FAMILY>/<VARIANT>`
> **Official documentation:** `<DOCUMENTATION_SOURCES>`
> **Data and task specification:** `<DATA_SOURCE>`
> **Executable implementation and revision:** `<CODE_REPOSITORY>`, `<CODE_REVISION>`
> **Runtime and evaluation configuration:** `<RUNTIME_CONFIG>`
>
> Build a task–interaction profile of the benchmark's canonical evaluation from its official documentation, data, and implementation.
>
> **Establish scope.** Identify the official sources and revision. Enumerate the evaluation categories, modes, and settings. Group items with compatible task and interaction properties into benchmark units; split a unit when its core TS/IR fields differ materially. Record the subset and configuration for each unit.
>
> **Trace the interaction.** Follow task loading, the agent's visible input, actions, environment responses, final result, and evaluation. Represent these as a coarse interaction graph. Check whether feedback enters the next agent request, how history and state persist, who controls progress, and how the interaction terminates.
>
> **Assign the eight fields.** TS records requirements that remain under another faithful implementation: final goal, knowledge domain, indispensable capabilities, and successful outcome. IR records the current interface: interaction structure and feedback, actions and encoding, history and state, and control and termination. Assign tools, feedback timing, retries, and subagent use to IR; assign an action to TS when it is itself a task requirement.

> **Choose and justify labels.** Use the controlled vocabulary and the smallest sufficient label set: at most two domains, three capabilities, and three action families. Check each choice against its closest alternative. Attach supporting evidence and a short rationale to every field. Mark insufficient evidence as `unknown` and an absent mechanism as `not_applicable`. Check consistency between the fields, scope, and graph.

Evidence status and field boundaries. The profile specification distinguishes `observed` assignments, directly supported by documentation or code, from `inferred` assignments that normalize evidence into the controlled vocabulary. `unknown` denotes insufficient or conflicting evidence; `not_applicable` denotes an absent mechanism. Evidence identifiers, graph references, and rationales support the eight fields without adding comparison dimensions.

Three boundary checks are particularly relevant to memory environments. First, history presentation and state persistence are separate: a persistent store does not imply that its contents enter every agent request. Second, an available retrieval operation belongs to IR; information seeking belongs to TS only when acquiring missing information is indispensable to the task. Third, an offline correctness score is not agent-visible feedback unless the runtime returns it before the next decision. These checks preserve the distinction between task requirements and the mechanisms of their execution.

Source benchmark variants. Table 8 lists the variants used to construct the two source groups. The family and variant names are the same as those used for the retained training examples in Table 4. A variant with several benchmark units contributes all retained examples to its source group.

Table 8: Source benchmark variants used in the two source groups.

Source group	Benchmark family	Variants
TS	MemGround	`trpg_success`
	GateMem	`long_context`
	StructMemEval	`mem_agent`
	AMemGym	`native`, `awi`
IR	VehicleMemBench	`none`, `gold`, `key_value`, `summary`
	ToolMisuseBench	`api_model_train`
	ToolMaze	`legacy_full`
	TravelBench	`legacy_full`

Attributes within the shared fields. Table 9 lists the categorical and set-valued attributes in the supplied specification, using the field names in Table 1. Summaries, evidence references, and rationales explain assignments without adding distance terms. The specification and prompt do not replace unit-level assignments or numerical distances.

Table 9: Structured attributes in the eight TS/IR fields. Monospace names are profile-schema keys. Set sizes are upper bounds from the controlled vocabulary, not required label counts.

Field	Structured attributes
Task goal	`goal_type`: one final-goal category
Task domain	`domains`: at most two; `knowledge_scope`: one category
Core capability requirements	`core_capabilities`: at most three indispensable capabilities
Success conditions	`success_target`: one category; `hard_constraints`: at most three task requirements
Interaction structure and feedback	`topology`: dominant interaction structure; `feedback_mode`: feedback timing and availability
Action space	`action_families`: at most three; `encoding`: action representation
Context and state management	`history_policy`: presentation of past information; `state_persistence`: state lifetime
System orchestration	`controller`, `termination`, and `subagent_use`: control, stopping, and visible delegation