BEFORE YOU RANK SKILL MEMORIES, MEASURE WHAT YOUR BENCHMARK CAN EXPRESS

Anonymous authors
Paper under double-blind review

ABSTRACT

Agent memory and skill systems are ranked by success rate on benchmarks whose capacity to express the gaps they report is never measured. We report that capacity for a controlled skill-form protocol, in which the same source experience is serialized four ways—trajectory trace, flat script, hierarchy, and a `SKILL.md` artifact—under a matched retriever, prompt, and injection budget. Four quantities govern what such a comparison can show: the error rate, the repeat-measurement floor (the rate at which per-task outcomes flip when one configuration is rerun), the discriminating fraction (the share of tasks whose outcome moves across arms for reasons a rerun does not explain), and whether the task slice represents the split. On an archived version of this protocol, which controlled variance by deterministic decoding, the floor is 14.7% of per-task outcomes on ALFWorld and 12.9% on WebShop, and only 3 of 50 WebShop tasks respond to skill injection at all. A rerun inflates the number of discordant pairs, never their direction, which motivates a practical reliability rule: a gap should exceed the control's own floor. On the published WebArena Shopping numbers of a recent budget-matched study of web-agent memory, none of its nine vanilla-versus-module gaps clears the vanilla actor's floor under any of its three executors. Rerun under a protocol that passes these checks (the full split, three replicates, a bank where 89 of 140 tasks retrieve a same-type source, length-matched fillers, two executor families), the serialization of the same experience, which bundles its actions, plan, and observations, decides whether it helps. A raw trajectory is indistinguishable from no injection on both executors and on every slice, while the hierarchical form of the same experience adds 15.7 and 19.3 points ($p \leq 0.001$) and beats its own length-matched filler by 10.7 and 31.4 points. On one executor that gain needs a retrieved experience of the right task type; on the other it does not.

1 INTRODUCTION

A growing family of methods stores an agent's past experience and injects it back at test time: as reflections (Shinn et al., 2023), extracted insights (Zhao et al., 2023), workflows (Wang et al., 2024), programs (Wang et al., 2023; 2025; Zheng et al., 2025), procedural memories (Fang et al., 2025), reasoning strategies (Ouyang et al., 2026), or directory-backed `SKILL.md` artifacts (Anthropic, 2025). These methods are compared by a success rate on a fixed task set, usually from one run per arm. The comparison presupposes that the benchmark can express a difference of the size being reported. That presupposition is rarely measured, and when it is measured it does not always hold.

We ran into this on our own protocol. An archived skill-form study compared four serializations of the same experience on ALFWorld (Shridhar et al., 2020) and WebShop (Yao et al., 2022) under deterministic decoding. Rerunning one configuration of that study flips 14.7% of per-task outcomes on ALFWorld (14/95) and 12.9% on WebShop (31/240). On WebShop, 41 of 50 tasks return the same outcome under all sixteen arms, and of the nine that move, six move between four repeats of the same no-injection condition. Three tasks respond to skill injection at all. The paired tests the study reported found discordant outcomes on 16–24% of tasks, barely above what rerunning one arm produces.

None of the twelve paired form comparisons in that study's main matrix reached significance. Its instrument could not have shown otherwise: the gaps it was asked to resolve were no larger than its

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.

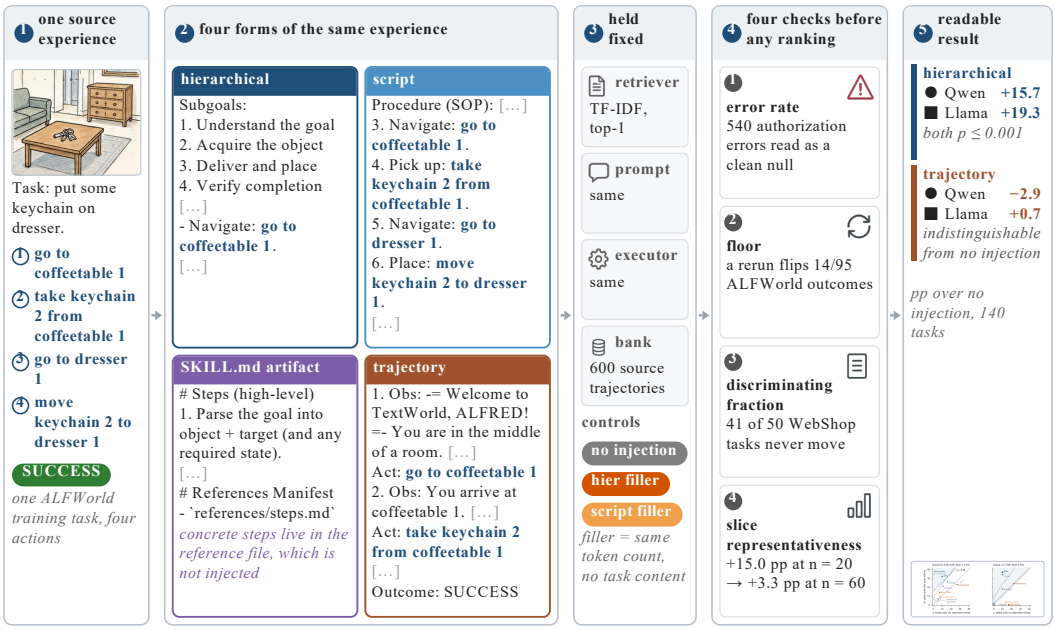

Figure 1: **The argument in one picture.** Results in (5) are majority-vote paired gaps to NO-INJECTION over the 140 `eval_id` tasks, three replicates per arm, exact McNemar test.

own rerun noise, on tasks most of which could not move. This paper measures that capacity first, then asks the archived question again under conditions that can answer it, and finds what the null had hidden: how the same experience is serialized decides whether an agent can use it.

Contributions and findings

- **Four checks before any ranking** (§3): error rate, repeat-measurement floor, discriminating fraction, and slice representativeness, each forced by a comparison that looked clean, none detecting another's failure, all run as gates.
- **A reliability rule from one asymmetry** (§3.1): a rerun inflates the number of discordant pairs and never their direction, so a single-run ranking can be reversed by a rerun unless its gap exceeds the control's floor. On published numbers of a budget-matched web-agent study (Hajimiri et al., 2026), **none of nine** vanilla-versus-module gaps on WebArena Shopping clears it.
- **Under those conditions, the serialization decides whether experience helps**, with actions, plan, and observations varying together (§5; 140 tasks, 3 replicates, 7 arms, 2 executor families, 5,880 episodes). The raw trajectory ties no injection on both executors and every slice; the hierarchical form gains **+15.7 and +19.3 points** (+18.6 and +28.6 with the harness's action guard off) and beats its length-matched filler by **+10.7 and +31.4**. The length effect, the bias of a task-list prefix, and the need for a same-type retrieval each change with the executor.

2 Related Work

Experience and skill memory for agents. Methods differ in what they store and in what form: verbal self-reflections (Shinn et al., 2023), insights and successful trajectories (Zhao et al., 2023), workflows (Wang et al., 2024), executable programs (Wang et al., 2023; 2025; Zheng et al., 2025), procedural memories built and updated as trajectories or scripts (Fang et al., 2025), reasoning strategies from successes and failures (Ouyang et al., 2026), separate planning and execution memories (Ye et al., 2025), multi-granularity skills with routing (Miao et al., 2026), context-conditional guidelines (Fu et al., 2024), and coarse-to-fine memories (Yang et al., 2025); AFTER evaluates procedural-memory control and adaptation on enterprise tasks (Belikova et al., 2026). `SKILL.md` artifacts have become a deployment format (Anthropic, 2025), and how they are written and maintained (Gao et al., 2026) and harnessed at run time (Xia et al., 2026) is now studied. Closest to

Table 1: **Four checks, each forced by a comparison that looked clean.** Error rate and floor are read before any comparison, discriminating fraction and slice after the runs and before a ranking is read; for each, the incident that forced it and its values. Archive: the earlier skill-form protocol. This study: the full-split main comparison; floors are of NO-INJECTION; the slice row gives the first 60 against the other 80 tasks, for SCRIPT on Qwen and HIERARCHICAL on Llama. No quantity detects a failure of another.

Quantity	Definition	Incident that forced it	Archive: ALFWorld · WebShop	This study: Qwen · Llama
Error rate	share of episodes ending in an API or harness error; runs above 20% refused	540 authorization errors read as a clean null: 0/60 in three arms, floor 0.0%	not recorded	0 errors in 114 runs
Floor F	per-task flip rate over all replicate pairs of one arm	greedy decoding still flipped one outcome in seven	14/95 · 31/240	11.0% · 1.0%
Discriminating fraction	tasks moving across arms, minus those moving across control replicates	41 of 50 WebShop tasks constant under all sixteen arms	— · 3/50	55/140 · 68/140
Slice	gap on a prefix against the gap on its complement	$+15.0$ pp at n=20 replicated exactly; $+3.3$ pp at n=60	—	$+13.3$ vs $+2.5$ · $+15.0$ vs $+22.5$

our question, Skill-as-Pseudocode rewrites markdown skills as typed pseudocode and finds on ALFWorld's unseen split that the gain needs typed contracts paired with concrete action templates; the contract alone falls below the prose baseline (Li et al., 2026). Each ranks its method by success rate; our comparison holds the source experience, retriever, and prompt fixed and varies only the serialization.

Evaluation of memory and skill modules. Hajimiri et al. (2026) show that a vanilla actor given the same token budget as AWM, ASI, or ReasoningBank matches or exceeds them on WebArena and WorkArena, and that single-run success rates hide task-level instability; they report Any-of-3 and All-of-3 rates for that reason. Their accounting is restricted to online augmentation, and they note that for offline methods "a fair comparison there requires different accounting". Ours is that offline, fixed-bank case, and we test rankings against their instability measure (§3.1).

Variance in benchmarks. Variance from data sampling, initialization, and hyperparameters can match the improvements a benchmark reports (Bouthillier et al., 2021), and seed variance in LLM benchmarks is substantial (Madaan et al., 2024); both concern an aggregate score, while paired agent comparisons need the per-task flips and their direction, tested with McNemar's exact test (McNemar, 1947).

3 WHAT A BENCHMARK CAN EXPRESS

Let an arm a be a fully specified configuration and $y_a^{(r)}(t) \in \{0,1\}$ the outcome of task t in replicate r. Replicates share every setting and differ only in the run. We use three replicates per arm and read each arm by its per-task majority $\bar{y}_a(t)$. Two arms are compared by the paired gap $\Delta_{a,c} = \frac{1}{n}\sum_t[\bar{y}_a(t) - \bar{y}_c(t)]$ and an exact two-sided McNemar test on the discordant counts $b = \#\{\bar{y}_a = 1, \bar{y}_c = 0\}$ and $c = \#\{\bar{y}_a = 0, \bar{y}_c = 1\}$, written b:c.

Error rate. A run whose every model call fails still terminates, writes a summary, and reports a success rate of zero. When one of our waves lost its model endpoint, every downstream number stayed self-consistent (Table 1), because the other three checks presuppose that the run measured something. The error rate is read first.

Repeat-measurement floor. The floor of arm a is the per-task flip rate over all replicate pairs, $F_a = \sum_{r<s}\sum_t \mathbb{1}[y_a^{(r)}(t) \neq y_a^{(s)}(t)] \,/\, \binom{R}{2} n$. With $R = 3$ a task that moves at all flips in exactly two of the three pairs, so $F_a = \frac{2}{3} m_a/n$, where m_a counts tasks whose outcome is not constant

Table 2: **No published vanilla-versus-module gap clears the vanilla floor.** The reliability rule on WebArena Shopping numbers of Hajimiri et al. (2026) (187 tasks, three runs per method). Gap: vanilla actor minus each memory module, from their Table 1 three-run means. Floor: $\frac{2}{3}$(Any-of-3 − All-of-3) of the vanilla actor, from their Table 4. None of the nine gaps exceeds the floor. Arithmetic on published aggregates; no rerun.

	Gap, vanilla minus module (pp)			
Executor	AWM	ASI	ReasoningBank	Vanilla floor (pp)
Gemini 3 Flash	6.59	3.03	2.32	$\frac{2}{3}(54.01 - 42.25) = 7.84$
GPT-5.4-mini	8.55	5.35	12.12	$\frac{2}{3}(46.52 - 27.81) = 12.47$
Qwen 3.6-27B	4.64	2.32	6.24	$\frac{2}{3}(54.01 - 38.50) = 10.34$

across replicates. Greedy decoding does not make it zero: the archived protocol relied on it and flipped one outcome in seven.

Discriminating fraction. A task constant under every arm cannot express an effect at any sample size, and a task that moves between replicates of one arm moves for reasons the manipulation did not cause. We count tasks that move across arms and subtract those that move across control replicates.

Slice representativeness. A `--limit N` flag takes the first N tasks in the order the task files are found on disk, not a sample. Replication across runs rules out run noise on a slice and cannot detect selection. At n=20 a skill arm's +15.0 pp replicated exactly on independent runs; the first 20 tasks contained no `pick_two_obj_and_place`, and at n=60 the gap was +3.3 pp. The subtler case in §5.5 has all six task types in both slices.

3.1 A RELIABILITY RULE: THE GAP AGAINST THE FLOOR

The floor explains how many paired outcomes disagree. It cannot explain in which direction they disagree: two replicates of one arm produce discordant pairs that split near evenly, so a rerun inflates $b + c$ and leaves $b/(b + c)$ near one half. We call a comparison of arm a against control c *admissible* when $\Delta_{a,c} > F_c$, that is, when the treatment separates from the control by more than the control separates from itself. This is a practical reliability rule, not a statistical test. F_c is how often a single rerun changes a task's outcome, so a single-run comparison whose gap is below it can be reversed by rerunning, and most agent-memory results are single runs. With replicates, inference comes from paired tests, which we run on majority votes (exact McNemar) and on every replicate (a task-level permutation test, Appendix B); a well-estimated effect can lie below F_c, and F_c is itself an estimate. In the archive, discordance of 16–24% sat against a 14.7% floor, and the one lopsided split, SCRIPT over TRAJECTORY, was 19:6 (p=0.015).

Applied to a published comparison. For WebArena Shopping (187 tasks) and three executors, Hajimiri et al. (2026) report each method's three-run mean (their Table 1) and its Any-of-3 and All-of-3 rates (their Table 4). Any-of-3 minus All-of-3 is exactly m_a/n, so the vanilla actor's floor is $\frac{2}{3}$(Any − All); their gap, a difference of three-run means, equals the paired gap averaged over runs. None of the nine vanilla-versus-module gaps in Table 2 exceeds that floor (closest: 12.12 against 12.47). None satisfies the reliability rule, consistent with their reading that the vanilla actor matches the modules on Shopping. A directional paired inference, like their four-domain aggregate, needs per-task outcomes, which are not released; we did not test either.

4 PROTOCOL

Source experience and forms. The bank is built from ALFWorld's training split by executing the environment's expert walkthroughs, one source trajectory per training task. Each trajectory becomes a canonical record—task query, extracted facts and pitfalls, and a bounded trace—and a deterministic, rule-based serializer renders the same record in four forms (Figure 2). The construction uses no model calls, so three builds of one form are byte-identical. All four forms open with the same task line, facts, and pitfalls, and differ in three things (Table 3): whether the source task's concrete

Table 3: **The forms differ in three things: source actions, a type-level plan, and the source room.** Measured on the 600-item bank. Shared lines: lines identical across all bank items of the same task type, i.e. the form's task-type template.

Form	Source actions	Type-level plan	Room obs.	Shared lines	Tokens
HIERARCHICAL	yes	subgoals with stop tests	no	79.3%	399
SCRIPT	yes	generic first, last steps	no	49.5%	206
ARTIFACT	no	high-level steps	no	86.2%	312
TRAJECTORY	amid observations	none	yes	45.1%	314

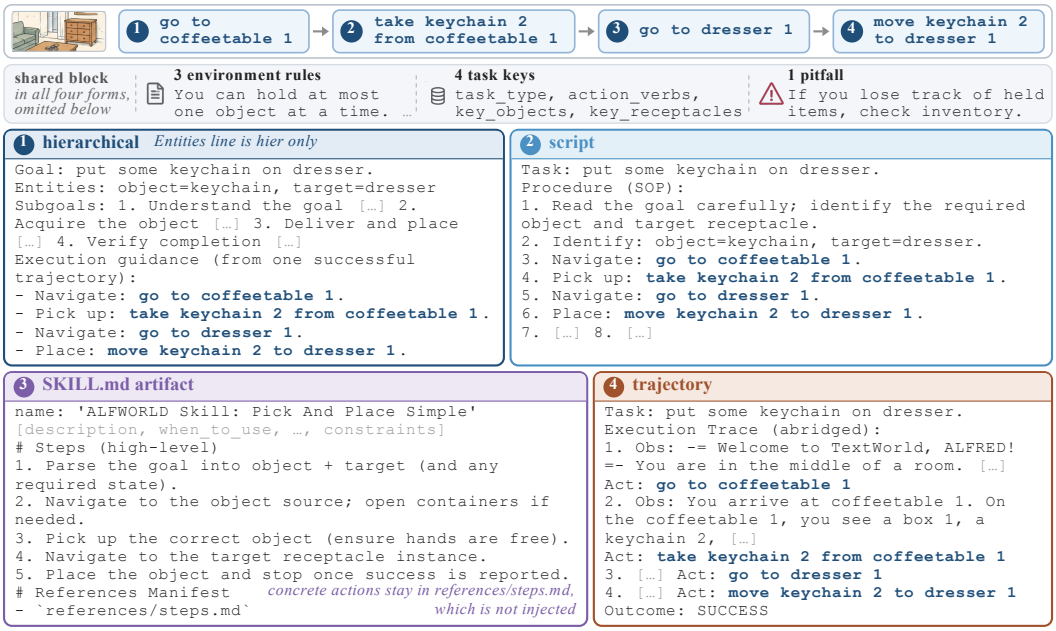

Figure 2: **Same experience, four forms,** abridged from the injected text; grey brackets mark omissions and the four source actions are in bold. HIERARCHICAL and SCRIPT carry them as steps, TRAJECTORY between observations of the source room, and the injected ARTIFACT body not at all.

actions are present, whether they are framed by a task-type plan, and whether the source room's observations come with them. HIERARCHICAL carries both the actions and a plan of four subgoals, each with an intent, a how-to, and a stop condition. SCRIPT carries the actions as a numbered procedure. ARTIFACT injects the `SKILL.md` body (Anthropic, 2025), whose steps are written at the task-type level; the concrete steps live in a reference file that is not injected. TRAJECTORY carries the actions interleaved with the observations of the room they were taken in.

Retrieval and injection. Every form is retrieved by the same TF–IDF match of the test query against the source query, and the top-1 item is injected (k=1). On the full split, all four forms retrieve the same source trajectory for 140 of 140 tasks, so form is the only varying component, and no injected item is truncated (0 of 5,880 episodes). The length-matched filler arm retrieves exactly as its skill arm does and replaces the text with neutral filler words of the same token count under a pseudonymised skill identifier.

Bank size and scale. Bank size fixes how often retrieval returns a relevant neighbour, and we treat it as a variable (§5.1). The usual task-type match rate stays flat (63–70%) across bank sizes, so we also read whether the retrieved source shares the target object, from ALFWorld's structured task identifiers. The main comparison uses 600 source trajectories: 89 of the 140 tasks retrieve a same-type source; at 1,500 the same-type, same-object rate on the first 60 tasks is 93%, close to lookup of a near-duplicate. It runs 7 arms (four forms, two fillers, NO-INJECTION) × 3 replicates

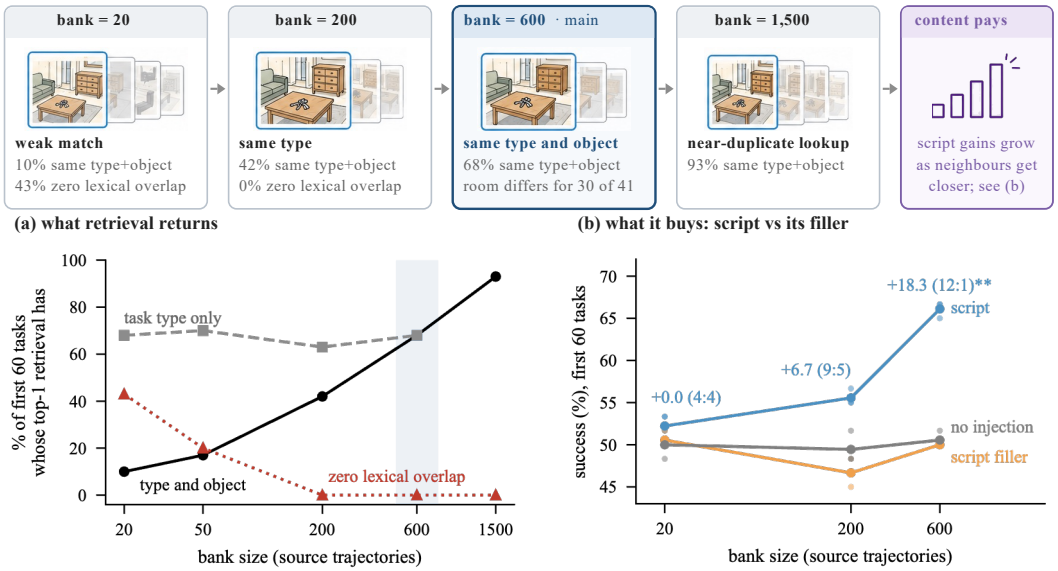

Figure 3: **Content pays only once retrieval finds a same-type, same-object neighbour.** First 60 `eval_id` tasks. **(a)** Offline retrieval audit; the shaded band is the bank of the main comparison. **(b)** Small points are replicates, lines their means; labels give SCRIPT over its length-matched filler as paired gap in pp, discordant counts, and ** for exact McNemar $p < 0.01$. At 600, 11 of the 41 same-type, same-object sources are in the test task's own room.

on all 140 `eval_id` tasks per executor; the bank-size and step-cap studies use the first 60 tasks. In total, 114 runs in ten waves (96 for the main comparison and the bank-size and step-cap studies, 12 for the guard ablation, 6 for the `eval_ood` test); none has an error.

Executors and budget. Two executors from different families act at temperature zero, so every floor we report arises under greedy decoding, through an OpenAI-compatible local server: Qwen3.6-35B-A3B, a mixture-of-experts reasoning model run with reasoning disabled, and Llama-3.3-70B-Instruct (Grattafiori et al., 2024), a dense non-reasoning model. Each step the executor sees the goal, the injected block, the observation, and the valid commands, and emits one action within 256 tokens. Episodes end at 40 steps. The NO-INJECTION control injects nothing. The harness includes an action guard: when the chosen action manipulates an object the goal does not name, it is replaced by a goal-consistent one (taking the goal object, else exploring), and navigation is never rewritten. The rule is identical in every arm and rewrites 1.2–6.2% of steps. Switching it off lowers NO-INJECTION more than HIERARCHICAL, so the hierarchical gain rises from $+15.7$ to $+18.6$ points on Qwen and from $+19.3$ to $+28.6$ on Llama (Appendix D).

5 RESULTS

5.1 RETRIEVAL QUALITY DECIDES WHETHER CONTENT CAN MATTER

At 20 source trajectories, 43% of tasks retrieve an item with zero lexical overlap, and only 10% retrieve a same-type, same-object neighbour (Figure 3a). On such a bank the injected "skill" is close to an arbitrary text. Holding the form fixed at SCRIPT and growing the bank (Figure 3b), the skill arm's replicates rise from 30, 32, 32 to 33, 33, 34 to 40, 39, 40 successes out of 60. The filler, whose content does not depend on the bank, stays at 31, 30, 30; 29, 28, 27; and 30, 30, 30. Skill over filler is $+0.0$ pp (4:4), $+6.7$ pp (9:5, p=0.42), and $+18.3$ pp (12:1, p=0.0034). The three NO-INJECTION arms, run in three separate waves, agree pairwise with p=1.00, so no uncontrolled variable moved between waves. The filler's own effect is $+3.3$, -1.7, and $+0.0$ pp. Content becomes worth something only once retrieval finds a relevant neighbour, and a form comparison run on a thin bank compares arbitrary injections.

Table 4: **Hierarchical is admissible and significant on both executors; trajectory ties no injection on both.** Four forms of the same experience on all 140 `eval_id` tasks. Successes per replicate (of 140), the arm's own floor, and the majority-vote paired gap to NO-INJECTION with discordant counts and exact McNemar p. Shaded and bold: admissible and $p < 0.05$.

	Qwen3.6-35B-A3B					Llama-3.3-70B-Instruct				
Form	reps	floor	gap	b:c	p	reps	floor	gap	b:c	p
HIERARCHICAL	94, 94, 95	7.6%	**+15.7**	32:10	0.0009	71, 72, 71	0.5%	**+19.3**	37:10	0.0001
SCRIPT	81, 89, 84	6.2%	+7.1	26:16	0.16	68, 68, 69	0.5%	**+17.1**	34:10	0.0004
ARTIFACT	81, 84, 82	8.6%	+7.1	21:11	0.11	53, 51, 52	1.0%	+5.7	18:10	0.18
TRAJECTORY	70, 66, 69	5.7%	−2.9	23:27	0.67	46, 45, 45	1.4%	+0.7	24:23	1.00
NO-INJECTION	74, 73, 73	11.0%	—			43, 43, 44	1.0%	—		
HIERARCHICAL filler	79, 78, 77	8.6%	+5.0	14:7	0.19	27, 28, 28	1.0%	−12.1	0:17	<0.0001
SCRIPT filler	76, 74, 75	10.5%	+0.0	9:9	1.00	33, 34, 32	1.0%	−7.9	3:14	0.013

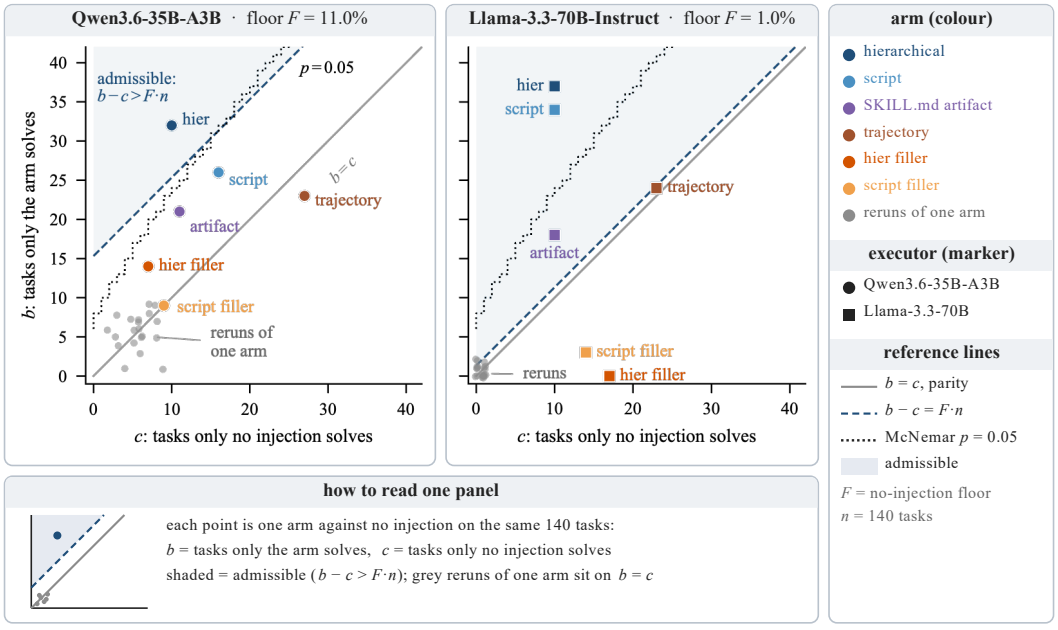

Figure 4: **A rerun spreads along the diagonal; an effect leaves it.** All 140 tasks; each coloured point is an arm's majority vote over three replicates against that of NO-INJECTION, and grey points are pairs of replicates of one arm. On Llama, whose reruns barely move, significance is the binding test.

5.2 SKILL FORM ON THE FULL SPLIT

Table 4 gives the main comparison and Figure 4 places it against what a rerun produces. Three results hold on both executors.

The raw trajectory adds no solved tasks. TRAJECTORY against NO-INJECTION is −2.9 pp (23:27, p=0.67) on Qwen and +0.7 pp (24:23, p=1.00) on Llama. Its discordant pairs split evenly and it sits on the diagonal, where a rerun's pairs lie (Figure 4). The experience it carries is the same experience that helps in other forms: HIERARCHICAL over TRAJECTORY is +18.6 pp on both executors (31:5 and 26:0, both $p < 0.0001$), and SCRIPT over TRAJECTORY is +10.0 pp (21:7, p=0.0125) and +16.4 pp (26:3, $p < 0.0001$).

The hierarchical form is the one admissible form on both executors. HIERARCHICAL beats NO-INJECTION by +15.7 pp against a Qwen floor of 11.0% (floor 95% CI [7.1, 15.2], so it clears the

Table 5: **On Qwen the gain needs a same-type retrieval; on Llama another type's experience helps as much.** Gap to NO-INJECTION split by whether top-1 retrieval returned a source of the current task's type (89 tasks) or of another type (51). Retrieval is identical across forms, so one partition serves every arm. The two groups differ in task-type composition, so the split is conditional and exploratory. Shaded: $p < 0.05$, unadjusted.

	Qwen3.6-35B-A3B						Llama-3.3-70B-Instruct					
	same type (89)			other type (51)			same type (89)			other type (51)		
Arm	gap	b:c	p	gap	b:c	p	gap	b:c	p	gap	b:c	p
HIERARCHICAL	**+21.3**	24:5	0.001	+5.9	8:5	0.58	**+19.1**	23:6	0.002	**+19.6**	14:4	0.031
SCRIPT	+12.4	19:8	0.052	−2.0	7:8	1.00	**+14.6**	19:6	0.015	**+21.6**	15:4	0.019
ARTIFACT	+10.1	15:6	0.078	+2.0	6:5	1.00	+5.6	11:6	0.33	+5.9	7:4	0.55
TRAJECTORY	+2.2	16:14	0.86	−11.8	7:13	0.26	+1.1	16:15	1.00	+0.0	8:8	1.00
HIERARCHICAL filler	+7.9	10:3	0.092	+0.0	4:4	1.00	**−9.0**	0:8	0.008	**−17.6**	0:9	0.004

rule only narrowly), and by +19.3 pp against a Llama floor of 1.0%. It is the only form whose gain survives Holm correction over the twelve comparisons with NO-INJECTION on both executors, and a permutation test on every replicate agrees (+15.0 and +20.0 pp, both $p \leq 0.0004$; Appendix B). On Qwen it also beats SCRIPT (+8.6 pp, 17:5, p=0.017); on Llama the two are tied (+2.1 pp, p=0.61). SCRIPT is significant on Llama (+17.1 pp, p=0.0004) and inadmissible on Qwen, where its +7.1 pp gap is below the 11.0% floor.

The `SKILL.md` artifact sits between. ARTIFACT is +7.1 pp and +5.7 pp over NO-INJECTION, significant on neither executor; a preregistered test adding the 134 unseen-room tasks of `eval_ood` leaves it unresolved (pooled +5.1 pp, 34:20, p=0.076; Appendix D). It beats TRAJECTORY on Qwen (+10.0 pp, 27:13, p=0.039) and loses to SCRIPT and HIERARCHICAL on Llama (−11.4 pp, p=0.014; −13.6 pp, p=0.0034). The ordering common to both executors is {HIERARCHICAL, SCRIPT} > ARTIFACT > {TRAJECTORY, NO-INJECTION}.

5.3 WHAT THE GAIN DEPENDS ON

Tables 3 and 4 line up: source actions inside a task-type plan help most, either one alone helps less, and actions buried among the source room's observations do not help. Skill-as-Pseudocode reached the same pairing by ablating its own bundle (Li et al., 2026). Whether that gain needs the retrieved experience to fit the task is itself executor-dependent (Table 5). On Qwen, HIERARCHICAL gains +21.3 pp when retrieval returns a same-type source and +5.9 pp (p=0.58) when it does not, and SCRIPT falls from +12.4 to −2.0 pp: the gain is carried by the retrieved experience. On Llama, HIERARCHICAL and SCRIPT gain as much from another type's experience (+19.6 and +21.6 pp) as from a matching one, while filler of the same length hurts in both groups: this executor profits from structured procedural text as such. How much of its gain is the experience and how much generic structure cannot be separated here; that needs a mismatched-skill or content-free structure control in the same form, which we did not run.

5.4 CONTENT, LENGTH, AND MECHANISM

The gain comes from content. Against a filler of identical token count, HIERARCHICAL gains +10.7 pp on Qwen (27:12, p=0.024) and +31.4 pp on Llama (48:4, $p < 0.0001$); SCRIPT gains +7.1 pp (p=0.17) and +25.0 pp (42:7, $p < 0.0001$).

The effect of length changes sign between executors. On Qwen the fillers are neutral: +0.0 pp at 206 tokens and +5.0 pp (p=0.19, below the floor) at 399. On Llama both are harmful, and the longer one more so: −7.9 pp (3:14, p=0.013) and −12.1 pp (0:17, $p < 0.0001$). Llama's actions under filler are 100% valid ALFWorld verbs and contain no filler words, and the harm shows in behaviour: under the HIERARCHICAL-length filler 52.9% of its actions are `look`, against 33.1% under NO-INJECTION (Figure 5c; counts in Table 11). A length control calibrated on one executor therefore does not transfer to another, and each executor needs its own filler arm.

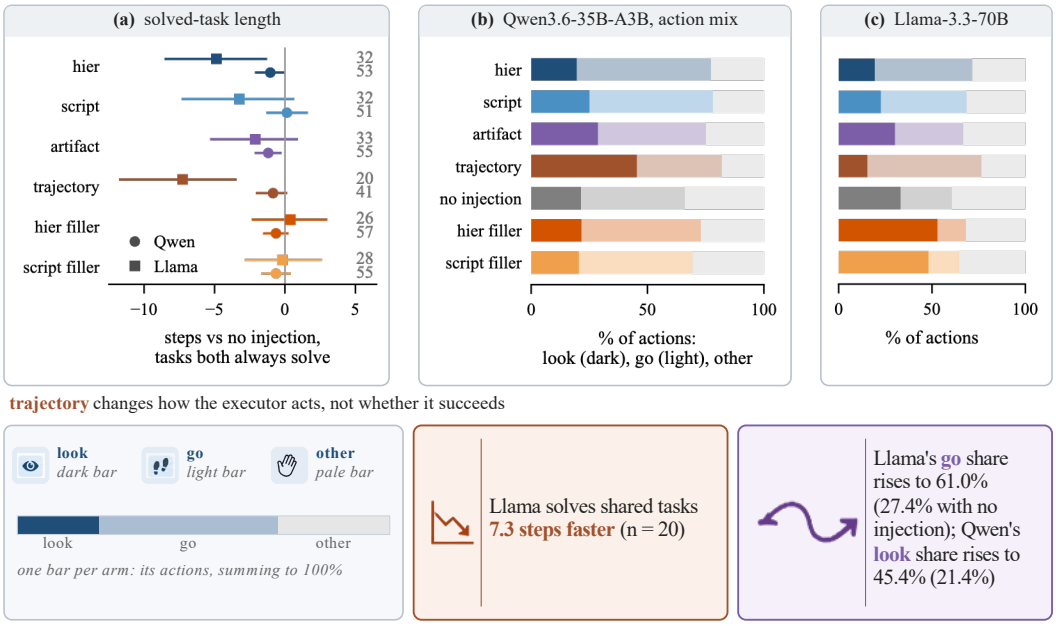

Figure 5: **Real content shortens solved episodes on Llama; fillers leave them unchanged.** All 140 tasks, three replicates pooled. **(a)** Steps against NO-INJECTION on the tasks both arms solve in every replicate (their number at right), 95% percentile bootstrap over tasks. **(b, c)** Action mix per arm.

Content speeds up solved tasks; form decides how many are solved. Every failure ends at the 40-step cap, so mean episode length only restates success; we compare it on tasks both arms solve in all three replicates (Figure 5a). On Llama every real-content form shortens them, TRAJECTORY most (7.3 steps over 20 tasks), and the fillers do not; on Qwen every difference is within about one step. The raw trajectory thus helps on tasks the agent already solves and adds none it does not.

5.5 WHICH CONCLUSIONS SURVIVE A CHANGE OF SLICE

On the first 60 tasks alone, Qwen's SCRIPT gains +13.3 pp against +2.5 on the other 80, so a prefix study would have called HIERARCHICAL and SCRIPT tied; on Llama the prefix understates HIERARCHICAL (+15.0 against +22.5). The bias changes sign with the executor, so only the full split removes it (Appendix E); a 60-step cap leaves the ranking unchanged (Appendix F).

6 LIMITATIONS AND CONCLUSION

Scope. One benchmark and split, and two executors differing in family, size, and training (no capability ladder). Each form is one deterministic serializer that changes actions, plan, and observations at once, so §5.3 is an observational comparison, not a factorial ablation; methods that change what is stored, such as Reflexion or AWM, were not run here. **Success is convergence within the budget**: every failure exhausts the step cap, so early-quitting executors may rank forms differently. **Unresolved or untested**: the artifact's gap to NO-INJECTION stays unresolved with `eval_ood` added; no reasoning-enabled executor was run; the bank-size and step-cap studies use the first-60 prefix; the guard ablation covers only HIERARCHICAL against NO-INJECTION. **Conclusion.** Measure what a benchmark can express before ranking on it; then serialization decides whether experience helps.

AI USE STATEMENT

In this work, we used generative AI tools, a coding and writing assistant used through a command-line agent, for tasks with required disclosure: designing and giving feedback on the experimental methodology (including the replicate design, the length-matched filler control, the action-guard ablation, and the preregistered `eval_ood` test), proposing and refining hypotheses, implementing methods (the measurement gates, changes to the run harness, and the analysis scripts), and interpreting results. We have not used generative AI tools to generate synthetic data, formulate or prove mathematical claims, or translate text, and the remaining required-disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for tasks with recommended disclosure: drafting and editing the paper, suggesting its structure, title, and keywords, identifying and summarizing related literature, formatting references, creating figures, and simulating reviews of drafts. Data plots were drawn by AI-written plotting code; illustrative elements in Figures 1–3 were produced with image-generation tools and laid out by hand, and every data panel in them is embedded from the analysis code without being redrawn.

We have reviewed all AI-assisted work. Every number in the paper is regenerated from released per-task outcomes by one released script and was checked against our evidence log; every cited work was checked against its arXiv record before being cited; AI-written code is covered by unit tests and was run by the authors; and figures produced with image-generation tools were checked against the data, with factual errors corrected in the final versions. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Appendix A specifies the bank construction, serializer, retriever, filler, executor prompt, action guard, and serving settings; Appendix B the statistics; Appendix C every run. The supplementary material contains the per-task outcomes of all 114 runs, the experiment harness with its unit tests, and a single script that recomputes every number, table, and figure from those outcomes. Banks are built deterministically and frozen to files, and decoding is greedy; the repeat-measurement floors we report quantify the variation that remains across reruns.

ETHICS STATEMENT

The study uses two public benchmarks and open-weight executors, involves no human subjects or personal data, and reanalyses another paper only through its published aggregate tables.

REFERENCES

Anthropic. Agent skills: Filesystem-backed skill artifacts (documentation and specification), 2025. Online documentation/specification for `SKILL.md`-based skill directories.

Julia Belikova, Rauf Parchiev, Evgeny Egorov, Grigorii Davydenko, Gleb Gusev, Andrey Savchenko, and Maksim Makarenko. Managing procedural memory in LLM agents: Control, adaptation, and evaluation, 2026.

Xavier Bouthillier, Pierre Delaunay, Mirko Bronzi, Assya Trofimov, Brennan Nichyporuk, Justin Szeto, Naz Sepah, Edward Raff, Kanika Madan, Vikram Voleti, Samira Ebrahimi Kahou, Vincent Michalski, Dmitriy Serdyuk, Tal Arbel, Chris Pal, Gaël Varoquaux, and Pascal Vincent. Accounting for variance in machine learning benchmarks. In *Proceedings of Machine Learning and Systems*, 2021. arXiv:2103.03098.

Runnan Fang, Yuan Liang, Xiaobin Wang, Jialong Wu, Shuofei Qiao, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. Memp: Exploring agent procedural memory, 2025.

Yao Fu, Dong-Ki Kim, Jaekyeom Kim, Sungryull Sohn, Lajanugen Logeswaran, Kyunghoon Bae, and Honglak Lee. AutoGuide: Automated generation and selection of context-aware guidelines for large language model agents, 2024.

Haoyu Gao, Jai Lal Lulla, Hong Yi Lin, Sebastian Baltes, Christoph Treude, and Mansooreh Zahedi. From registry to repository: How AI agent skills are written, adapted, and maintained, 2026.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 herd of models, 2024.

Sina Hajimiri, Masih Aminbeidokhti, Jose Dolz, Ismail Ben Ayed, Issam H. Laradji, Spandana Gella, and Nicolas Gontier. Are online skill and memory modules always worth their tokens? A budget-constrained study of web agents. In *Conference on Empirical Methods in Natural Language Processing*, 2026. arXiv:2606.15017.

Xinze Li, Yuhang Zang, Yixin Cao, and Aixin Sun. Skill-as-pseudocode: Refactoring skill libraries to pseudocode for LLM agents, 2026.

Lovish Madaan, Aaditya K. Singh, Rylan Schaeffer, Andrew Poulton, Sanmi Koyejo, Pontus Stenetorp, Sharan Narang, and Dieuwke Hupkes. Quantifying variance in evaluation benchmarks, 2024.

Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947.

Yongliang Miao, Ziyang Yu, Liang Zhao, Bowen Zhu, and Hasibul Haque. SkillLens: Adaptive multi-granularity skill reuse for cost-efficient LLM agents, 2026.

Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long T. Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. ReasoningBank: Scaling agent self-evolving with reasoning memory. In *International Conference on Learning Representations*, 2026. arXiv:2509.25140.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning, 2020.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory, 2024.

Zora Zhiruo Wang, Apurva Gandhi, Graham Neubig, and Daniel Fried. Inducing programmatic skills for agentic tasks, 2025.

Boming Xia, Liming Zhu, Zhenchang Xing, Qinghua Lu, Dino Sejdinovic, and Xiwei Xu. Harnessing agent skills: Architectural patterns and a reference architecture for skill-mediated LLM agents, 2026.

Wei Yang, Jinwei Xiao, Hongming Zhang, Qingyang Zhang, Yanna Wang, and Bo Xu. Coarse-to-fine grounded memory for LLM agent planning, 2025.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Shicheng Ye, Chao Yu, Kaiqiang Ke, Chengdong Xu, and Yinqi Wei. H^2R: Hierarchical hindsight reflection for multi-task LLM agents, 2025.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: LLM agents are experiential learners, 2023.

Boyuan Zheng, Michael Y. Fatemi, Xiaolong Jin, Zora Zhiruo Wang, Apurva Gandhi, Yueqi Song, Yu Gu, Jayanth Srinivasa, Gaowen Liu, Graham Neubig, and Yu Su. Skillweaver: Web agents can self-improve by discovering and honing skills, 2025.

APPENDIX OVERVIEW

Section	Content	Supports
A Protocol details	bank, serializer, retrieval, filler, prompt, guard, serving	§4
B Statistics	majority vote, exact test, floor, bootstrap, replicate key	§3
C Run inventory	all 114 runs in ten waves	every table
D Additional results	bank size, first-60 wave, task types, guard rates and ablation, artifact on unseen rooms, solved-task length, archive, external	§5.1–5.4
E Slices	first 60 against the other 80, both executors; task-type composition; replicated prefix	§5.5
F Step-cap sensitivity	caps of 40 and 60 steps	§5.4
G Pairwise comparisons	every pair of forms, both executors	§5.2
H Reproducibility	released outcomes and the regenerating script	all

A PROTOCOL DETAILS

Bank construction. For each ALFWorld training task with an expert walkthrough, in the order the task files are found on disk, the walkthrough is executed in the environment with entity names demangled, and the resulting observation–action sequence is kept as the source trajectory. The first N such tasks form a bank of size N. That order is a property of the file system, so another machine can draw a different prefix and a different bank; our banks are frozen to files and every outcome is keyed by task identifier. A canonical record holds the task query, facts extracted from the task (task type, action verbs, key objects, key receptacles, and three environment rules), pitfalls, and the trajectory. The serializer (version `may2026_serializer_v2`, profile `parity_v2`) is rule-based: TRAJECTORY cuts each observation at 137 characters and keeps every action; SCRIPT writes the actions as a numbered procedure with generic first and last steps; HIERARCHICAL writes four task-type subgoals with intent, how-to, and stop condition, followed by the actions as execution guidance; ARTIFACT writes a `SKILL.md` with YAML frontmatter and five task-type steps, and puts the concrete steps in `references/steps.md`. The injected artifact view is the full `SKILL.md`; the reference file is not injected. Across the 600-item bank the numbers of distinct payloads are 599 (TRAJECTORY), 560 (SCRIPT), 560 (HIERARCHICAL), and 600 (ARTIFACT).

Retrieval. A scikit-learn TF–IDF vectorizer (English stop words, at most 50,000 features) is fit on the source queries of the bank; the test query is scored by the dot product with each source and the top-1 item is injected. The same vectorizer and ranking serve every form, and on the full split all four forms receive the same source for 140 of 140 tasks.

Filler. The length-matched filler arm takes the item its skill arm retrieves, replaces the text by neutral filler words until it reaches the same estimated token count, and replaces the skill identifier by a hash pseudonym of equal length, because descriptive identifiers name the task. Mean injected lengths match exactly: 399 tokens for HIERARCHICAL and its filler, 206 for SCRIPT and its filler.

Executor prompt. Every step sends one system message and one user message. The system message is: *"You are an expert ALFWorld agent. You must output ONLY actions. Output up to 1 actions, one per line. The FIRST action MUST be copied verbatim from the admissible commands list for the current step. Additional actions should be plausible next commands; if unsure, output 'look'. You may think silently, but do not output explanations or extra text. Use retrieved skills as guidance when relevant, but current task goal constraints override skill examples. Avoid repeating the same action in a loop."* The user message contains, in order, the goal; a block of goal constraints parsed from the goal (required count, target object type, movable container, required state, transformation tool, target receptacle); the injected block; the last eight actions; the observation; the list of valid commands; and the step index. The first returned line is matched to a valid command exactly, then after normalization, then as a substring, longest command first.

Action guard. If the matched command manipulates an object whose type the goal does not name, or applies the wrong transformation, it is replaced by the least recently used valid command that

satisfies the goal constraints (`take`, the required transformation, or `move`), or else by an exploration command; navigation and `look` are never rewritten. Table 11 gives its rate per arm.

Serving. Both executors are served by an OpenAI-compatible local server at temperature 0 with at most 256 completion tokens per step. Qwen3.6-35B-A3B runs with `reasoning_effort=none`; on this server the low, medium, and default settings each spend about 100 reasoning tokens per action, so the setting is effectively on or off. Episodes end at 40 steps unless stated.

B STATISTICS

Each arm has $R = 3$ replicates, so the per-task majority is always defined. Pairs of arms are compared on the majority outcomes with an exact two-sided McNemar test, $p = \min\{1,\ 2\sum_{k\le\min(b,c)} \binom{b+c}{k} 2^{-(b+c)}\}$; the implementation is tested against three archived values (19:6, p=0.014633; 14:5, p=0.063568; 17:12, p=0.458258). The floor F_a uses all $\binom{3}{2}$ replicate pairs. The discriminating fraction takes the five full-split arms (four forms and NO-INJECTION) on their majority outcomes and removes tasks that move across the three NO-INJECTION replicates. Intervals in the figures are 95% percentile bootstrap intervals over tasks, 2,000 resamples, fixed seed. Two runs are grouped as replicates only when they agree on benchmark, split, form, artifact view, retriever, budget regime, construction profile, bank identity, executor, reasoning setting, injected-content type, step bonus, granted step budget, and guard setting.

Uncertainty, every replicate, and multiplicity. The floor is itself an estimate: a task bootstrap gives 11.0% $[7.1, 15.2]$ for NO-INJECTION on Qwen and 1.0% $[0.0, 2.4]$ on Llama. Majority votes discard how many of three replicates succeeded, so Table 6 also reports a replicate-level paired effect: for each task, the arm's success rate over its three replicates minus that of NO-INJECTION, averaged over tasks, with a task bootstrap interval and a sign-flip permutation test (10,000 draws). We take the twelve comparisons with NO-INJECTION in Table 4 as the primary family and Holm-adjust them; this designation was made after the runs, and every other test in the paper is exploratory.

Table 6: Every arm against NO-INJECTION, all 140 tasks: majority-vote gap with exact McNemar p and its Holm adjustment over the twelve comparisons, and the replicate-level paired effect with 95% bootstrap interval and permutation p.

		majority vote			every replicate		
Executor	Arm	gap	p	Holm	gap	95% interval	p_{perm}
Qwen	HIERARCHICAL	$+15.7$	0.0009	0.0085	$+15.0$	$[+7.4, +22.4]$	0.0004
	SCRIPT	$+7.1$	0.1641	0.9849	$+8.1$	$[+0.2, +15.7]$	0.0487
	ARTIFACT	$+7.1$	0.1102	0.7713	$+6.4$	$[+0.2, +12.9]$	0.0662
	TRAJECTORY	-2.9	0.6718	1.0000	-3.6	$[-12.4, +5.2]$	0.4585
	HIERARCHICAL filler	$+5.0$	0.1892	0.9849	$+3.3$	$[-1.7, +8.6]$	0.2464
	SCRIPT filler	$+0.0$	1.0000	1.0000	$+1.2$	$[-3.6, +6.2]$	0.7104
Llama	HIERARCHICAL	$+19.3$	<0.0001	0.0011	$+20.0$	$[+11.0, +28.8]$	<0.0001
	SCRIPT	$+17.1$	0.0004	0.0039	$+17.9$	$[+8.8, +26.7]$	0.0002
	ARTIFACT	$+5.7$	0.1849	0.9849	$+6.2$	$[-0.7, +13.1]$	0.1074
	TRAJECTORY	$+0.7$	1.0000	1.0000	$+1.4$	$[-8.1, +11.0]$	0.8066
	HIERARCHICAL filler	-12.1	<0.0001	0.0002	-11.2	$[-16.4, -6.0]$	<0.0001
	SCRIPT filler	-7.9	0.0127	0.1018	-7.4	$[-12.9, -2.1]$	0.0118

C RUN INVENTORY

Table 7: All 114 runs, 11,364 episodes. Every run has zero errors; 21 further runs lost to a GPU failure on the host are excluded. The first wave predates the injected-content field; its retrieval arm carries real SCRIPT content.

Wave	Executor	Tasks	Bank	Cap	Arms	Runs
1: first probe	Qwen	20	20	40	SCRIPT, NO-INJECTION	6
2: bank 20	Qwen	60	20	40	SCRIPT, its filler, NO-INJECTION	9
3: bank 200	Qwen	60	200	40	SCRIPT, its filler, NO-INJECTION	9
4: bank 600, four forms	Qwen	60	600	40	four forms, two fillers, NO-INJECTION	21
5: step cap	Qwen	60	600	60	SCRIPT, TRAJECTORY, NO-INJECTION	9
6: full split	Qwen	140	600	40	four forms, two fillers, NO-INJECTION	21
7: full split	Llama	140	600	40	four forms, two fillers, NO-INJECTION	21
8: guard off	Qwen	140	600	40	HIERARCHICAL, NO-INJECTION (guard off)	6
9: guard off	Llama	140	600	40	HIERARCHICAL, NO-INJECTION (guard off)	6
10: unseen rooms	Qwen	134	600	40	ARTIFACT, NO-INJECTION (`eval_ood`)	6

D ADDITIONAL RESULTS

Bank size. Table 8 gives every replicate and pairing behind Figure 3b.

Table 8: SCRIPT, its filler, and NO-INJECTION at three bank sizes, first 60 tasks. Successes per replicate (of 60) and majority-vote pairings.

	bank 20	bank 200	bank 600
SCRIPT	30, 32, 32	33, 33, 34	40, 39, 40
filler	31, 30, 30	29, 28, 27	30, 30, 30
NO-INJECTION	31, 29, 30	29, 31, 29	30, 31, 30
SCRIPT vs NO-INJECTION	$+3.3$ 7:5 p=0.77	$+5.0$ 10:7 p=0.63	$+18.3$ 14:3 p=0.0127
filler vs NO-INJECTION	$+3.3$ 4:2 p=0.69	-1.7 2:3 p=1.00	$+0.0$ 2:2 p=1.00
SCRIPT vs filler	$+0.0$ 4:4 p=1.00	$+6.7$ 9:5 p=0.42	$+18.3$ 12:1 p=0.0034

The first-60 wave. Table 9 gives the bank-600 wave on the first 60 tasks, every arm. It is the slice whose prefix bias §5.5 documents, and it is where HIERARCHICAL and SCRIPT look tied.

Table 9: Bank 600, first 60 tasks, Qwen3.6-35B-A3B. Successes per replicate (of 60), the arm's floor, and the gap to NO-INJECTION.

Arm	replicates	floor	vs NO-INJECTION
HIERARCHICAL	41, 40, 45	8.9%	$+20.0$ 15:3 p=0.0075
SCRIPT	40, 39, 40	4.4%	$+18.3$ 14:3 p=0.013
ARTIFACT	35, 33, 35	6.7%	$+8.3$ 10:5 p=0.3
TRAJECTORY	28, 29, 28	4.4%	-1.7 12:13 p=1.00
NO-INJECTION	30, 31, 30	5.6%	—
HIERARCHICAL filler	31, 30, 30	3.3%	
SCRIPT filler	30, 30, 30	4.4%	

Task types. Table 10 splits the full-split majority outcomes by ALFWorld task type. Each type has 13 to 35 tasks, so we report counts without tests. HIERARCHICAL exceeds NO-INJECTION on both executors for pick and place, clean then place, and cool then place. In two cells NO-INJECTION beats every form: heat then place on Llama (14 of 16 against 6 to 10) and pick two and place on

Qwen (13 of 24 against 8 to 11). The aggregate ranking therefore averages over types with opposite signs.

Table 10: Majority-vote successes by ALFWorld task type, all 140 tasks. Bold: the best arm in the row for that executor.

		Qwen3.6-35B-A3B					Llama-3.3-70B-Instruct				
Task type	n	hier	script	artif.	traj	none	hier	script	artif.	traj	none
look at object in light	13	5	4	**6**	3	**6**	3	2	2	2	**4**
pick and place	35	**30**	26	22	23	21	**21**	19	14	14	12
clean then place	27	**17**	**17**	15	15	11	**16**	**16**	8	13	5
cool then place	25	**19**	17	16	11	10	**12**	**12**	11	5	4
heat then place	16	13	10	**14**	9	12	10	10	10	6	**14**
pick two and place	24	11	9	10	8	**13**	**9**	**9**	7	5	5

Guard and action mix. Table 11 lists, for each arm on the full split, the share of steps the guard rewrote and the shares of `look` and `go` actions (three replicates pooled).

Table 11: Guard rewrites and action mix per arm, all 140 tasks, three replicates pooled.

	Qwen3.6-35B-A3B			Llama-3.3-70B-Instruct		
Arm	guard	`look`	`go`	guard	`look`	`go`
HIERARCHICAL	253/8,338 (3.0%)	19.6%	57.5%	423/10,514 (4.0%)	19.3%	52.1%
SCRIPT	260/9,223 (2.8%)	25.0%	53.1%	454/10,822 (4.2%)	22.6%	45.8%
ARTIFACT	313/9,462 (3.3%)	28.7%	46.3%	644/12,625 (5.1%)	30.1%	36.4%
TRAJECTORY	135/10,315 (1.3%)	45.4%	36.4%	152/12,448 (1.2%)	15.3%	61.0%
NO-INJECTION	593/10,557 (5.6%)	21.4%	44.5%	221/13,542 (1.6%)	33.1%	27.4%
HIERARCHICAL filler	435/9,911 (4.4%)	21.6%	51.2%	181/14,680 (1.2%)	52.9%	15.1%
SCRIPT filler	630/10,183 (6.2%)	20.5%	48.8%	245/14,420 (1.7%)	48.1%	16.3%

Guard ablation. Table 12 reruns HIERARCHICAL and NO-INJECTION on all 140 tasks with the guard off (three replicates each, both executors, same frozen bank), and every guard-off episode has zero rewrites. With the guard off the replicates are 89, 88, 87 (HIERARCHICAL) and 64, 64, 62 (NO-INJECTION) on Qwen, and 65, 66, 66 and 26, 25, 25 on Llama. The guard mostly helps the arm with no memory, which picks a wrong object more often; the hierarchical gain it hides grows when it is removed. Only this pair was ablated, so the full form ranking without the guard is untested.

Table 12: Guard ablation, all 140 tasks, majority vote over three replicates, exact McNemar. Last row: change in the per-task HIERARCHICAL gain between guard on and off, 95% bootstrap over tasks.

	Qwen3.6-35B-A3B			Llama-3.3-70B-Instruct		
Comparison	gap	b:c	p	gap	b:c	p
HIERARCHICAL vs NO-INJECTION, guard on	$+15.7$	32:10	0.0009	$+19.3$	37:10	<0.0001
HIERARCHICAL vs NO-INJECTION, guard off	$+18.6$	37:11	0.0002	$+28.6$	42:2	<0.0001
HIERARCHICAL, guard off vs on	-4.3	4:10	0.1796	-3.6	3:8	0.2266
NO-INJECTION, guard off vs on	-7.1	5:15	0.0414	-12.9	1:19	<0.0001
gain change, on $-$ off	-2.9	$[-10.0, +3.6]$		-9.3	$[-16.4, -2.9]$	

The artifact on unseen rooms. Before running it, we fixed one test for the artifact's unresolved gap: artifact against NO-INJECTION on Qwen, three replicates on all 134 `eval_ood` tasks (ALFWorld's unseen rooms), the same frozen bank and settings, pooled with the 140 `eval_id` tasks and tested once with exact McNemar at α=0.05. The replicates on `eval_ood` are 79, 78, 80 (ARTIFACT) and 76, 72, 76 (NO-INJECTION) of 134. Pooled over 274 tasks the gap is $+5.1$ pp (34:20, p=0.076). On `eval_ood` alone it is $+3.0$ pp (13:9, p=0.52, 95% bootstrap $[-3.7, +9.7]$),

below that split's NO-INJECTION floor of 4.0%. The artifact's effect stays unresolved, and it is smaller in rooms the bank never saw.

Solved-task length. Table 13 gives the numbers behind Figure 5a.

Table 13: Mean steps, arm minus NO-INJECTION, on tasks both solve in all three replicates; 95% bootstrap intervals.

	Qwen3.6-35B-A3B		Llama-3.3-70B-Instruct	
Arm	tasks	steps	tasks	steps
HIERARCHICAL	53	-1.0 $[-2.1, -0.0]$	32	-4.9 $[-8.6, -1.3]$
SCRIPT	51	$+0.1$ $[-1.3, +1.6]$	32	-3.2 $[-7.3, +0.7]$
ARTIFACT	55	-1.2 $[-2.2, -0.2]$	33	-2.1 $[-5.3, +0.9]$
TRAJECTORY	41	-0.9 $[-2.1, +0.2]$	20	-7.3 $[-11.8, -3.4]$
HIERARCHICAL filler	57	-0.6 $[-1.6, +0.3]$	26	$+0.4$ $[-2.4, +3.0]$
SCRIPT filler	55	-0.6 $[-1.7, +0.4]$	28	-0.2 $[-2.9, +2.6]$

The archive. The archived floors come from runs that repeated a configuration by accident: on ALFWorld, three run families overlapping on their shared first five tasks (95 paired episodes); on WebShop, the four no-injection arms of a 4-form $\times$ 4-retriever matrix, which inject nothing, see identical prompts, and are therefore four replicates of one condition. Neither was designed to measure a floor, and the three replicates per arm in this paper are.

The external reading. From Hajimiri et al. (2026)'s Table 4, the floors of the modules themselves are 7.49 (AWM), 9.27 (ASI), and 9.99 (ReasoningBank) under Gemini 3 Flash; 10.69, 9.27, and 11.77 under GPT-5.4-mini; and 11.05, 10.70, and 8.56 under Qwen 3.6-27B. Every floor on that domain lies between 7.49 and 12.47 points; the largest gap, 12.12, is below the vanilla floor of its executor.

E SLICES

Figure 6 and Table 14 are the evidence behind §5.5. Each form is compared with NO-INJECTION on the first 60 tasks that a `--limit 60` run takes, on the other 80, and on all 140, inside the same full-split waves, so the slices differ only in which tasks they hold.

Table 14: **No form reaches $p < 0.05$ on both halves of the split.** Majority-vote paired gap to NO-INJECTION (pp), discordant counts, and exact McNemar p per slice. Bold: $p < 0.05$. Shaded: the rows of Figure 6's callouts.

		first 60			other 80			all 140		
Executor	Form	gap	b:c	p	gap	b:c	p	gap	b:c	p
Qwen3.6-35B	HIERARCHICAL	$+\mathbf{23.3}$	17:3	0.003	$+10.0$	15:7	0.13	$+\mathbf{15.7}$	32:10	0.0009
	SCRIPT	$+13.3$	12:4	0.077	$+2.5$	14:12	0.85	$+7.1$	26:16	0.16
	ARTIFACT	$+6.7$	10:6	0.45	$+7.5$	11:5	0.21	$+7.1$	21:11	0.11
	TRAJECTORY	-3.3	12:14	0.85	-2.5	11:13	0.84	-2.9	23:27	0.67
Llama-3.3-70B	HIERARCHICAL	$+15.0$	15:6	0.078	$+\mathbf{22.5}$	22:4	0.0005	$+\mathbf{19.3}$	37:10	0.0001
	SCRIPT	$+15.0$	14:5	0.064	$+\mathbf{18.8}$	20:5	0.004	$+\mathbf{17.1}$	34:10	0.0004
	ARTIFACT	$+0.0$	6:6	1.00	$+10.0$	12:4	0.077	$+5.7$	18:10	0.18
	TRAJECTORY	$+3.3$	11:9	0.82	-1.2	13:14	1.00	$+0.7$	24:23	1.00

Composition does not explain the flips. Both slices hold all six ALFWorld task types, and no type's share differs between them by more than 5 points (Figure 6, top): the prefix is not missing a task type the way the n=20 slice of §3 was. Selection acts below the level of task type.

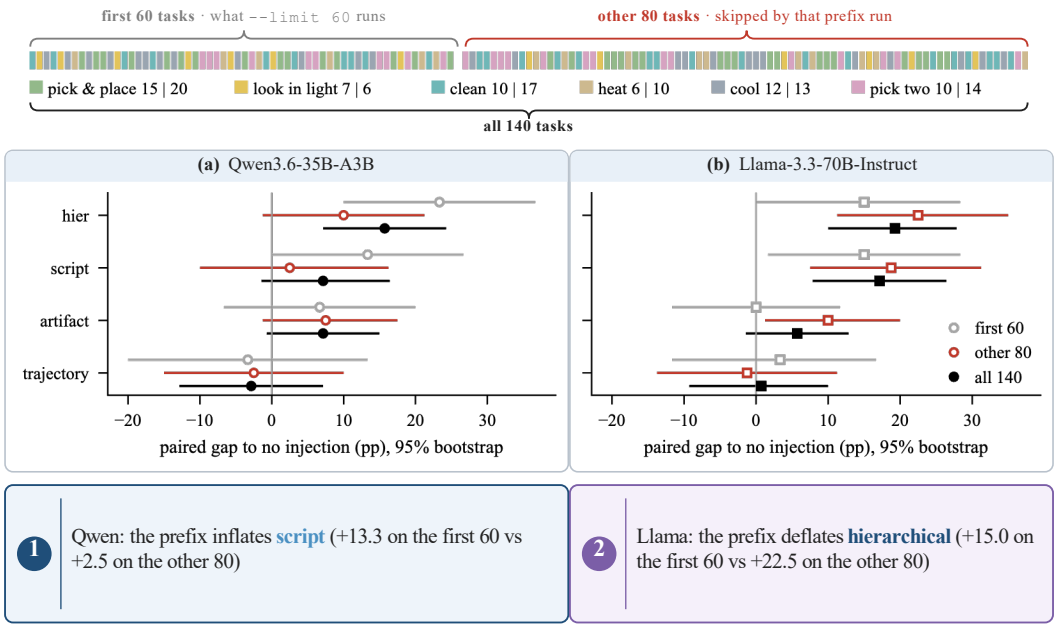

Figure 6: **A prefix's bias changes sign with the executor.** Tasks in the order their files are found on disk; paired gaps to NO-INJECTION on the first 60, the other 80, and all 140 tasks of the same full-split waves, with 95% percentile bootstrap intervals over tasks.

The prefix's bias is a property of its tasks, not of one wave. The bank-600 wave of Table 9 is an independent set of three replicates on the same first 60 tasks. It gives SCRIPT +18.3 pp (14:3, p=0.013) and HIERARCHICAL +20.0 pp (15:3, p=0.0075), against +13.3 and +23.3 pp for the same slice of the full-split wave. Both waves put SCRIPT well above its +2.5 pp on the other 80, so a replicated prefix result still does not transfer to the rest of the split.

Each half alone is underpowered. On Qwen, HIERARCHICAL is significant on the first 60 and not on the other 80 (p=0.13). On Llama, HIERARCHICAL and SCRIPT are significant on the other 80 and not on the first 60 (p=0.078 and 0.064). A study run on either half would have reported a different set of significant forms on each executor; only the full 140-task split supports the ranking of §5.2 on both.

F STEP-CAP SENSITIVITY

Table 15: First 60 tasks, same frozen bank at 600, three replicates; only the step cap changes. Successes per replicate (of 60) with the arm's floor, and failures ending at the cap.

	40-step cap		60-step cap	
Arm	reps (floor)	fail at cap	reps (floor)	fail at cap
SCRIPT	40, 39, 40 (4.4%)	61/61	42, 40, 41 (5.6%)	57/57
TRAJECTORY	28, 29, 28 (4.4%)	95/95	29, 32, 30 (6.7%)	89/89
NO-INJECTION	30, 31, 30 (5.6%)	89/89	30, 34, 32 (6.7%)	84/84
SCRIPT vs NO-INJECTION	+18.3 pp 14:3 p=0.0127		+16.7 pp 13:3 p=0.0213	
TRAJECTORY vs NO-INJECTION	−1.7 pp 12:13 p=1.00		−1.7 pp 11:12 p=1.00	
SCRIPT vs TRAJECTORY	+20.0 pp 14:2 p=0.0042		+18.3 pp 13:2 p=0.0074	

Raising the cap by 50% lifts all three arms by similar amounts (SCRIPT 66.1% to 68.3%, TRAJECTORY 47.2% to 50.6%, NO-INJECTION 50.6% to 53.3%), and mean episode length grows with the

cap (SCRIPT 20.9 to 27.2 steps, TRAJECTORY 24.4 to 34.0) because failed episodes use every step they are given.

G REMAINING PAIRWISE COMPARISONS

Table 16: Pairs between forms on all 140 tasks (majority vote, exact McNemar).

Pair	Qwen3.6-35B-A3B	Llama-3.3-70B-Instruct
HIERARCHICAL vs TRAJECTORY	+18.6 pp 31:5 $p<0.0001$	+18.6 pp 26:0 $p<0.0001$
HIERARCHICAL vs SCRIPT	+8.6 pp 17:5 $p=0.017$	+2.1 pp 9:6 $p=0.61$
HIERARCHICAL vs ARTIFACT	+8.6 pp 23:11 $p=0.058$	+13.6 pp 29:10 $p=0.0034$
SCRIPT vs TRAJECTORY	+10.0 pp 21:7 $p=0.0125$	+16.4 pp 26:3 $p<0.0001$
SCRIPT vs ARTIFACT	+0.0 pp 17:17 $p=1.00$	+11.4 pp 27:11 $p=0.014$
ARTIFACT vs TRAJECTORY	+10.0 pp 27:13 $p=0.039$	+5.0 pp 26:19 $p=0.37$
HIERARCHICAL vs its filler	+10.7 pp 27:12 $p=0.024$	+31.4 pp 48:4 $p<0.0001$
SCRIPT vs its filler	+7.1 pp 27:17 $p=0.17$	+25.0 pp 42:7 $p<0.0001$

H REPRODUCIBILITY

Every run records its full configuration, including the injected-content type, the replicate index, the step budget granted, and the reasoning setting; two runs are grouped as replicates only when all of these agree. The gates in §3 are implemented as functions over per-episode records and are covered by unit tests, including tests that pin three archived p-values so that a change to the statistics cannot silently move an earlier result. The supplementary material contains the per-task outcomes of all 114 runs (`runs.json`) and a single script (`analyze.py`) that recomputes every number, table, and figure in this paper from that file.