Harness Evolution Hits a Ceiling: When Weight Training Should Begin

Anonymous authors
Paper under double-blind review

Abstract

Improving a long-horizon LLM agent means evolving the harness around a frozen model or training its weights. We follow one line through both: let a self-evolving harness make the system stronger first, then cross seed and evolved harnesses with base and trained weights to learn which gains the trained model keeps and which still need the runtime. We show that the right lever can be read off the agent's failure composition. Labelling failed trajectories by the first signal that fires separates process failures (blocked calls, loops, exhausted step budgets) from content failures (a delivered plan that is poor). Harness evolution repairs the former, the behaviour it instils can be trained into the weights, and content failures are what weight training is for. On DeepPlanning, a verifiable planning benchmark with a held-out split, a self-evolving harness loop, in which an LLM proposes edits to its own harness and a judge verifies them, lifts the held-out score of Qwen3.5-4B from 0.16 to 0.30 and of Qwen3.5-9B from 0.32 to 0.44, and the gain lands where the claim places it: for 4B, held-out delivery rises from 55% to 90% and loops fall, while content failures are left for the weights. Ablations pin the gain on capability-granting components rather than on enforcement, and the loop's own evidence package shows why it stops where it does. LoRA adapters trained on evolved-harness trajectories then internalise the gain: under the original harness they add +0.13 on held-out tasks for both sizes; on 4B they stack with the harness to more than double the held-out score, and on 9B the adapter alone matches the full evolution line, cutting content failures from a quarter of trajectories to one in twenty. A placebo adapter trained on answer-shuffled trajectories falls below the base model, so the gain is in the trajectory content; full fine-tuning on a larger trajectory set adds a further +0.04. The same loop transfers to WebArena-Lite (+0.09 on 117 unseen tasks), where its gain lives in what the model sees, two pages of history instead of one, and adapters trained on those trajectories do not add to it. The result is a diagnose-then-intervene rule applied twice: read the failure composition, and evolve the harness for process failures and train the weights for content failures; then read what the accepted edits changed, and train in the gains that changed what the model writes while keeping the harness for the gains that changed what it sees. Scores are four-rollout means against fresh anchors and comparisons are drawn within one night, except where marked, across eight models from six families and two benchmarks.

1 Introduction

A long-horizon LLM agent can be improved in two places. One can change the *harness*, the code and text around a frozen model: system prompts, memory tools, checkpoints, output gates, the tools it may call. Or one can change the *weights*, by fine-tuning on trajectories. Each lever has its own literature, self-evolving harnesses (Lin et al., 2026; Hu et al., 2024; Xia et al., 2026; Zhang et al., 2026a) and trajectory fine-tuning (Zeng et al., 2023; Chen et al., 2023; Qi et al., 2024), and concurrent work has begun to alternate the two (Kim et al., 2026a; Chen et al., 2026; Ye et al., 2026). A self-evolving harness can raise the system's score without touching the weights, but that score does not say whether the model still needs the runtime to reach it. Our line is therefore: let self-evolution make the harnessed system stronger first, then cross seed and evolved harnesses with base and trained weights to learn which gains the trained model keeps and which still need the runtime. Both levers

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.

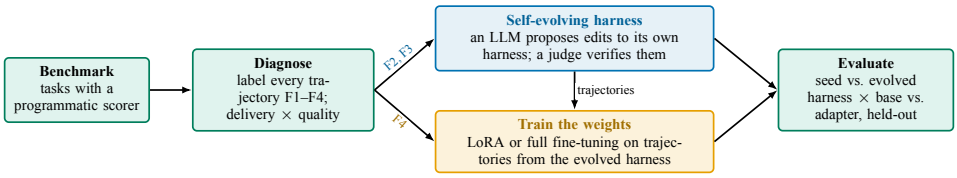

Figure 1: **Overview.** Trajectories under the seed harness are labelled by their first failure signal (Section 3.1) and the composition decides the lever: a self-evolving harness loop removes the process failures, the trajectories it produces train the weights for the content failures, and both levers are measured under the seed and the evolved harness on held-out tasks.

are expensive to evaluate and the per-rollout score is noisy, so the rule for *which* lever to pull must be grounded in measurements that survive the noise.

This paper builds that rule empirically on DeepPlanning (Zhang et al., 2026b), a travel-planning benchmark with deterministic programmatic scoring, using 48 development tasks for every decision and 72 held-out tasks no loop ever saw. We first fix the measurement protocol: with a per-rollout standard deviation of 0.023, every configuration is a four-rollout mean, differences below 0.035 are noise, and a fresh anchor follows every accepted change. We then label every trajectory by the first failure signal that fires (Section 3.1): unbacked assertions in the plan (F1), the harness's own interventions getting in the way (F2), loops and exhausted step budgets (F3), or a delivered plan that is simply poor (F4). The hypothesis the paper tests is that the harness lever removes the process classes (F2, F3) and leaves the content class (F4) in place, that the behaviour the harness instils can be trained into the weights, and that the content class is the target for weight training (Figure 1).

Harness self-evolution (a loop in the style of AHE (Lin et al., 2026), Section 4; Figure 2) lifts Qwen3.5-4B by +0.143 / +0.133 (dev / held-out) over the seed harness and Qwen3.5-9B by +0.056 / +0.119, and the gains land where the taxonomy places them (Figure 3): for 4B, delivery rises from 59% to 88% and loop failures (F3) fall from 64% to 37% of trajectories; for 9B, delivery was already 97% and loops fall from 39% to 17%, so the gain shows up as better delivered plans. What the loop leaves behind is the content class: F4 rises from 10% to 28% of 4B trajectories and stays at 25% for 9B.

Why the loop stops at F4. Component ablations (Section 6) show that the compliance-enforcing components (exit gate, checkpoint, hard blocks) are mostly within noise, whereas the capability-granting components (a state ledger, a distilled lessons file, invented tools) carry the gain. Yet the evidence package the proposal model reads leads with gate and checkpoint metrics and never aggregates the scorer's failure composition: the loop optimises what it can see, and what it can see does not include the cause of F4.

Weight training (LoRA on trajectories collected under the evolved harness, Section 5) transfers to held-out tasks: on the seed harness the adapter alone adds +0.13 for both sizes, matching or exceeding what harness evolution delivered; stacked on the evolved harness it adds a further +0.04 / +0.08 (dev / held-out) for 4B, while for 9B the adapter alone reaches the evolved-harness level, so at that size the two levers substitute for each other. Each adapter moves its own failure class (the 4B adapter raises delivery and cuts loops, the 9B adapter cuts F4 from 23–28% of trajectories to 5–7%); a placebo adapter trained on answer-shuffled trajectories falls 0.16 / 0.23 below the base model, and full fine-tuning adds about +0.04 on held-out over LoRA.

Contributions. (1) A measurement protocol for harness and weight interventions on a noisy long-horizon benchmark, with a bound on selection optimism. (2) Harness self-evolution on two model sizes with held-out validation, decomposed into delivery, loop reduction and plan quality; F4 is the class the loop left in place in both Qwen lines. (3) Component ablations on both sizes: capability-granting components carry the gain, enforcement mostly does not. (4) An account of why the loop stops at F4, from what its evidence package shows the proposer. (5) LoRA and full-parameter training on evolved-harness trajectories, with a placebo control and a rank ablation: the gain transfers to held-out tasks, stacks with the harness on 4B and substitutes for it on 9B, and each adapter moves the failure class the rule assigns to it; on WebArena-Lite, where the gain lives in what the model sees, it did not train in. (6) Generality: across nine evolution lines on eight models from six families, the seed measurements separate the five lines where the harness lever paid (+0.05 to +0.14) from the three where it paid nothing; the loop transfers to a second benchmark.

2 Background and Related Work

2.1 Failure Analysis of LLM Agents

Taxonomies of agent failure have grown fine-grained: MAST (Cemri et al., 2025) annotates fourteen failure modes of multi-agent systems, TRAIL (Deshpande et al., 2025) and AgentErrorTaxonomy (Zhu et al., 2025) label reasoning, execution and planning errors inside traces, and Rahman et al. (2026) catalogue 78 failure types. These are organised by the *kind* of error and need an LLM or a human annotator; ours is organised by the *lever that removes* the error and is computed by a deterministic checker from process signals and the scorer. Ecdysis (Yue et al., 2026) comes closest, using recurrence across tasks to decide which failures the harness should absorb; we test the split by applying each lever and reading which class it moves.

2.2 Self-Evolving Harnesses and Automated Agent Design

A fast-growing line treats the harness around a frozen model as the optimisation target: ADAS (Hu et al., 2024) and AFlow (Zhang et al., 2024) search agent programs and workflow graphs, and AHE (Lin et al., 2026) introduced the propose-verify loop we build on. The loop has since been regularised against overfitting (Xia et al., 2026; Zhang et al., 2026a), and challenged to beat matched-budget test-time search (Wang et al., 2026b); memory architectures (Packer et al., 2023; Wu et al., 2026) and runtime governance (Chen, 2026) are particular harness components, and Zhang et al. (2026c) price planning guidance against a shuffled-text control. We add which class of failure the loop removed, which it left, and what its proposer could see.

2.3 Training Agents on Trajectories, and Moving Harness Gains into the Weights

Fine-tuning agents on trajectories has a settled recipe: AgentTuning (Zeng et al., 2023) and FireAct (Chen et al., 2023) distil GPT-4 trajectories into small models, WebRL (Qi et al., 2024) trains web agents on WebArena-Lite with an online curriculum, and STaR-style self-improvement (Zelikman et al., 2022) trains on the model's own filtered outputs, the recipe we use; Yu et al. (2024) moved a prompting scaffold's gain into the weights for single-turn tasks. Concurrent work brings the question to agent harnesses: Harness-Zero (Ye et al., 2026) distils an optimised harness into the weights and removes it at deployment, Co-Harness (Chen et al., 2026) and WHALE (Kim et al., 2026a) alternate harness search and weight updates, Kim et al. (2026b) show that harness-aware post-training is more robust to tool shift, and Yu et al. (2026) find that imitating an expert under an evolved harness can regress. These works switch lever by schedule or by training signal; we switch on the failure composition, obtain the removable-harness result with plain fine-tuning on the model's own trajectories, show that it is size-dependent, and add a placebo adapter trained on answer-shuffled trajectories. LoRA (Hu et al., 2022) and QLoRA (Dettmers et al., 2023) supply the machinery; long-horizon RL credit assignment (Schulman et al., 2017; Shao et al., 2024; Yan et al., 2026; Tan et al., 2026) is the route we do not take.

2.4 Evaluating Agents Under Noise

DeepPlanning (Zhang et al., 2026b) scores plans programmatically and admits a clean development / held-out split; WebArena (Zhou et al., 2024), AgentBench (Liu et al., 2024) and AgentBoard (Ma et al., 2024) are broader but rely on LLM judges or do not expose failure composition. Kapoor et al. (2024) document inadequate hold-outs, Miller (2024) ask for standard errors and paired comparisons, Xu et al. (2026) formalise the winner's curse of adaptive search, and Wang et al. (2026b) show that harness evolution overfits when search and report share a benchmark.

What this paper adds. Relative to these lines, this paper (i) labels failures by the lever that removes them and shows, by applying both levers, where the split holds; (ii) measures both levers on the same models and tasks under one calibrated noise model; (iii) traces the harness ceiling to what the loop's evidence package shows the proposer; and (iv) turns the results into a rule that reads the lever off the failure composition.

3 A Diagnostic Framework

3.1 The failure taxonomy

Each trajectory is labelled post hoc by an automated checker: a trajectory whose delivered plan scores at least 0.5 is *healthy*, and every other trajectory receives the first of the following signals that fires:

- **F1, evidence**: at least half of the plan's factual claims are unbacked by anything retrieved (unbacked-citation ratio ≥ 0.5), or the agent passed the exit gate with unresolved items.
- **F2, harness friction**: a harness intervention itself blocked the agent (a forced checkpoint unlock, or ten or more blocked tool calls) and the score is low.
- **F3, execution control**: the same tool call repeated three or more times, or the trajectory used 90% or more of its step budget, or the harness terminated it.
- **F4, planning capacity**: none of the signals above fired; in practice a plan was delivered and scores below 0.5 (rarely, an undelivered trajectory with no process signal).

Separately, we record the *delivery rate*, the fraction of trajectories that returned a parsable plan at all, and the *quality among delivered plans*, the mean score over those trajectories; the composite score is their product. This is the hypothesis the paper tests: F2 and F3 are the classes a harness can act on directly, and F4 is the class it did not reach in either Qwen line, since by construction nothing in the trajectory's process signals distinguishes an F4 trajectory from a healthy one; a harness gain that consists of removed F2 and F3 trajectories is a behaviour, which the weights can learn. Table 6 (Appendix D) gives the composition before and after evolution for both Qwen sizes.

3.2 Two identities behind the numbers

Selection optimism. A round keeps the best of several candidate readings. A reading is the mean of m rollouts of the 48-task score, with single-rollout standard deviation $\sigma = 0.023$ (Appendix A), so it carries noise of scale $\sigma/\sqrt{m}$.

Proposition 1 (selection optimism)**.** *Let $X_k = \mu_k + \varepsilon_k$, $k = 1, \ldots, K$, be the readings of K candidates with true means μ_k and zero-mean noises ε_k, each sub-Gaussian with parameter s. If the candidate with the largest reading is kept, $k^\star = \arg\max_k X_k$, then*

$$\mathbb{E}\big[X_{k^\star} - \mu_{k^\star}\big] \;\leq\; s\sqrt{2\ln K}. \tag{1}$$

Proof. $X_{k^\star} - \mu_{k^\star} = \varepsilon_{k^\star} \leq \max_k \varepsilon_k$, and the expected maximum of K zero-mean s-sub-Gaussian variables is at most $s\sqrt{2\ln K}$ (Boucheron et al., 2013). □

With 10 to about 30 candidates per line and one rollout per reading ($s = \sigma$) the bound is 0.05–0.06, above the 0.035 line, which is why the three single-rollout lines of Section 5.4 were re-measured under the full protocol before being reported; with four rollouts ($s = \sigma/2$) it is 0.025–0.030, inside the line, and the fresh anchor after every accept removes the optimism from the next round's anchor (Figure 7, Appendix G, shows the gap in our two lines). The bound concerns measurement noise only; whether a gain holds on other tasks is what the 72 held-out tasks answer.

Score identity. Let r be the delivery rate and $\bar{q}$ the mean score among delivered plans. On DeepPlanning an undelivered plan scores zero, so the composite score is $S = r\,\bar{q}$ exactly, and a change from $(r, \bar{q})$ to $(r', \bar{q}')$ decomposes as

$$\Delta S \;=\; \bar{q}\,\Delta r \;+\; r'\,\Delta\bar{q}. \tag{2}$$

A lever that only moves r is capped at $\bar{q}$, the quality of what the model already writes; a lever that raises $\bar{q}$ is worth $r'\,\Delta\bar{q}$, more when delivery is already high. The two factors are complementary, and Table 7 (Appendix D) reports both for the Qwen lines and MiniCPM.

4 Experimental Setup

4.1 Benchmark, models and measurement

DeepPlanning (Zhang et al., 2026b) poses multi-day travel-planning tasks that require tool calls (flights, hotels, attractions, distances), state tracking across many steps, and a final structured plan that is scored programmatically against hard constraints. We use 120 tasks split once, before any experiment, into 48 development tasks and 72 held-out tasks. All evolution, training and ablation decisions use only the 48 development tasks; the 72 held-out tasks are evaluated only for reporting. The official benchmark loop is replaced by our own harness that exposes state to the agent and logs every step; the official tools and scorer are unchanged. Our primary models are Qwen3.5-4B and Qwen3.5-9B, served locally with thinking disabled at temperature 0.7. Three further families (MiniCPM5-2B, Spark-X2.5-4B, Ministral-3-8B) and Gemma 4 12B test the other branch of the rule (Appendix H).

Every configuration is scored as the mean of four rollouts. Across 26 same-night cells the per-rollout standard deviation has median 0.023 (worst 0.042; Appendix A), so we treat a difference between two 4-rollout means as resolvable only above 0.035 and write "within noise" below it; comparisons are made between cells run on the same machine on the same night, since cross-night drift of one configuration reaches 0.05; the exceptions are marked where they occur and listed under Limitations.

4.2 Harness self-evolution protocol

We run a self-evolving loop in the style of AHE (Lin et al., 2026): an LLM reads the evidence from the current harness's own runs and rewrites the harness. One round: (1) run the current harness on the 48 development tasks (4 rollouts); (2) an evidence package is built from the run (score, healthy rate, F1–F4 counts, process metrics, worst cases) and handed to a proposal model (Kimi K3), which returns two candidate edits, each with a manifest and its own rollback condition; (3) each candidate is evaluated with 4 rollouts and judged against the round's anchor by fixed house rules (reject if the mean drops by more than 0.02, if the healthy rate drops by 0.05, or if the no-submission or repeat-call rate rises by 0.05) plus the proposer's own rollback condition; (4) the better accepted candidate is retained, and a *fresh* 4-rollout anchor of the new harness is measured before the next round. The editable surface is the harness definition only: prompt files, policy parameters, middleware and tools. Each line starts from the same seed harness (a state ledger, an exit gate, a mid-trajectory checkpoint and four hard blocks). The 4B line ran 13 rounds (26 candidates) and the 9B line 5 rounds (10 candidates); full logs are in Appendices B and C.

4.3 Weight training

Training data are trajectories collected under the *evolved* harness on the 48 development tasks: 8 rollouts per task for 4B and 12 for 9B, keeping the three best trajectories per task above a score threshold (0.5 for 4B, 0.6 for 9B), which gives 82 trajectories over 33 tasks for 4B and 98 over 40 tasks for 9B. Each assistant step is one sample; the loss is on the assistant output only. LoRA uses rank 32, $\alpha = 64$, dropout 0.05 and learning rate 10^{-4}, on the attention projections (qkvo) or attention plus MLP (qkvo+MLP), taken at the end of the first epoch; full-parameter fine-tuning uses 2×10^{-6} with warm-up and cosine decay (Appendix E). The adapters in Table 1 are evaluated on both harnesses and both splits on the same night as their base model; those of the second textbook (Appendix E) on the seed harness only.

5 Results

5.1 Harness self-evolution works and transfers

Figure 2 shows the per-round anchors of both lines and the readings of both candidates in each round. The 4B line accepted an edit in 10 of 13 rounds (18 of 26 candidates passed the judge) and moved from 0.156 to 0.292 (+0.136); the 9B line accepted 3 of 10 candidates over 5 rounds and moved from 0.305 to 0.390 (+0.085). Table 5 (Appendix D) re-measures seed and final harness outside the loop on both splits: the evolved harness transfers to held-out tasks (4B held-out gain +0.133 against dev +0.143; 9B +0.119 against +0.056).

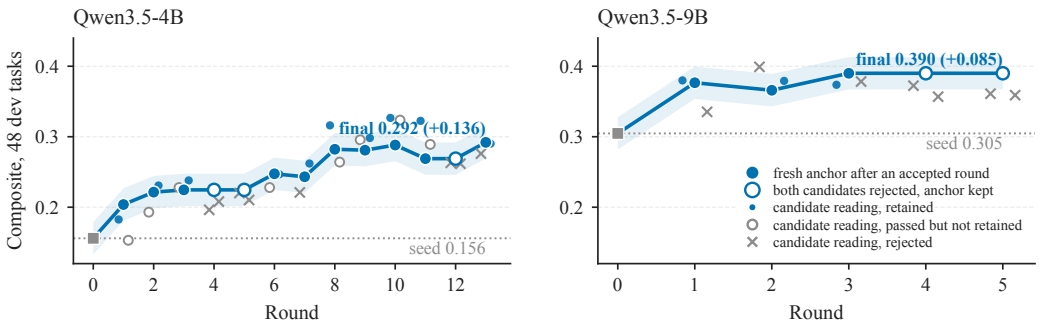

Figure 2: **Per-round fresh anchors of the two evolution lines** (48 development tasks, 4-rollout means; band = ± 0.023; filled: edit accepted, fresh anchor measured; hollow: both candidates rejected, anchor kept). Small markers are the two candidates of each round at their own readings (dot retained, hollow circle passed but not retained, cross rejected); the gap between a retained reading and the anchor that follows is the selection optimism of Proposition 1. The 4B dip at round 11 is a re-anchor after the context window was raised to 262k tokens.

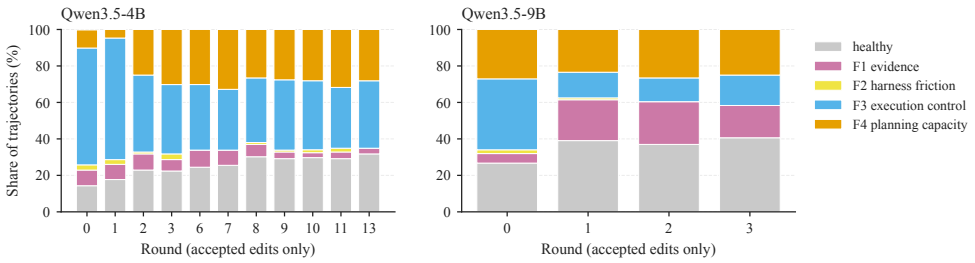

Figure 3: **Failure composition of the fresh anchor after each accepted round**, from the same per-round history files as Figure 2. Loops (F3) fall and the healthy share rises in both lines; F4 is never reduced. In the 9B line the lessons file raises the share of plans with unbacked claims (F1).

Where does the gain come from? Table 7 (Appendix D) splits each score into delivery rate and quality among delivered plans, and Table 6 gives the failure composition. For 4B, delivery rises from 59% to 88% and loop failures (F3) fall from 64% to 37% of trajectories; Equation 2 assigns 0.077 of the +0.136 to delivery and 0.058 to the higher mean among delivered plans (0.266 to 0.332), and a paired comparison between the bare model and the full evolved harness on the tasks both deliver (Appendix F) moves by only +0.011 on dev and +0.022 on held-out, consistent with the higher mean reflecting which plans are now delivered rather than better plans on the same tasks. For 9B, delivery was already 97% and the gain is a better plan among delivered ones (0.309 to 0.392), the $r' \Delta\bar{q}$ term, with loops falling from 39% to 17%. In neither line did the loop reduce F4: it grows from 10% to 28% of 4B trajectories as loops are converted and stays at 25% for 9B, where it is the largest failure class at the end; for 4B the residual F3 (37%) is still larger (Figure 3).

5.2 Training internalises the harness gain

Table 1 (plotted as Figure 5 in Appendix E) reports the same-night evaluation of the LoRA adapters trained on evolved-harness trajectories (Section 4.3). Four findings. **(i) The adapter transfers.** On the seed harness, the better adapter lifts held-out scores by +0.126 (4B, qkvo) and +0.131 (9B, qkvo+MLP), and development scores by +0.151 and +0.050; seven of the eight adapter-on-seed-harness cells are positive. **(ii) Stacking on 4B, substitution on 9B.** On the evolved harness, the 4B adapter still adds +0.041 (dev) and +0.082 (held-out); the 9B adapter adds +0.010 and +0.007, within noise, as is the other 9B adapter. For 9B either lever alone reaches 0.44–0.45 on held-out. **(iii) Content, not adapter.** The placebo (Table 10, Appendix E), trained with the identical recipe on the same trajectories with the assistant outputs permuted across samples, scores 0.158 below the base model on dev and 0.227 below on held-out, and its trajectories run 52–65 rounds instead of 19. Ranks 16, 32 and 64 all gain on held-out (Appendix E); full-parameter fine-tuning on a larger

Table 1: Base model and LoRA adapters trained on evolved-harness trajectories, under the seed and the evolved harness, on the 48 development and 72 held-out tasks. 4 rollouts per cell on one machine and one night, except the base-model cells under the evolved harness on held-out tasks, the lines' closing anchors from two nights earlier (4B: 2 rollouts); differences below 0.035 (0.05 against a 2-rollout cell) are within noise. Base-row delivery rates differ from Table 7 (a different night).

		Qwen3.5-4B				Qwen3.5-9B			
		Seed harness		Evolved harness		Seed harness		Evolved harness	
Quantity	Weights	Dev	Held-out	Dev	Held-out	Dev	Held-out	Dev	Held-out
Score	Base	0.158	0.164	0.301	0.297	0.348	0.323	0.404	0.442
Score	LoRA qkvo	**0.309**	**0.290**	0.342	**0.379**	0.327	0.371	0.395	0.433
Score	LoRA qkvo+MLP	0.247	0.259	**0.351**	0.359	**0.398**	**0.454**	0.414	0.449
Delivery rate	Base	52%	55%	90%	90%	96%	97%	98%	99%
Delivery rate	LoRA qkvo	78%	78%	95%	97%	95%	98%	100%	98%
Delivery rate	LoRA qkvo+MLP	67%	62%	90%	90%	94%	95%	96%	98%

second textbook beats LoRA by +0.044 (4B) and +0.042 (9B) on held-out. **(iv) Each adapter moves its own failure class** (Table 9, Appendix E). The 4B adapter reproduces the harness's behavioural effect: delivery rises from 52% to 78% on dev and from 55% to 78% on held-out, loops fall by 18–19 points, and F4 is unchanged. The 9B adapter, on a model whose delivery is already saturated, moves the content class instead: F4 falls from 23% to 7% on dev and from 28% to 5% on held-out, and the freed trajectories become healthy (34% to 46%, 28% to 53%): training reaches the class the loop did not.

Which placement wins follows what the textbook has to teach. The attention-only adapter wins on 4B and the adapter that also trains the MLP blocks wins on 9B (Table 1); we select the placement per model on the development split. The 4B textbook was collected by making a model that delivers half of its plans deliver, so what it mostly carries is that behaviour, and the attention-only adapter learned it better (delivery 78% against 67%, F3 40% against 51%; Table 9). The 9B textbook comes from a model that already delivers, so it carries plan content, and both adapters cut F4 (from 23% and 28% on dev and held-out), the one that also trains the MLP blocks, where language models hold most of their factual associations (Geva et al., 2021), by more (to 7% and 5% against 13.5% and 16% for attention only). A richer textbook should then make the MLP placement win on 4B as well, which the second textbook shows (Table 8).

5.3 The same loop on a second benchmark: WebArena-Lite

WebArena-Lite (Zhou et al., 2024; Qi et al., 2024) poses 165 web-navigation tasks on five self-hosted sites, with the official environments and scoring; natural-language answers are judged by an LLM. We split it once into 48 development and 117 test tasks and ran the same loop with Qwen3.5-9B under the protocol of Section 4.2, with a line measured on this benchmark ($\sigma = 0.045$, line 0.064). The seed harness reproduces the benchmark agent's interface and adds the ledger, checkpoint and exit gate of the DeepPlanning seed harness. Thirteen rounds accepted ten edits and moved the development anchor from 0.234 to 0.375; one interface edit in round 10, keeping the two most recent full pages in the context instead of one, accounts for +0.12 of it.

The gain transfers, with the same shape. Table 13 (Appendix H) reports seed and closing harness on the 117 test tasks, measured on one night: 0.256 to 0.344 (+0.088), delivery from 49% to 77%, the healthy share from 23% to 34%, and accuracy among delivered answers unchanged within noise (0.47 to 0.44). The failures change kind: at the seed, 165 of 237 undelivered trajectories stopped on a repeated action or three unparseable actions, under the closing harness 13 of 108, the rest exhausting step or call budgets. The harness removed the process failures and carried the agent further into each task, and the correctness of what it submits did not move. Removing the three seed components from the closing harness costs 0.10 on the development tasks (0.396 to 0.297, same night).

Where the gain lives decides whether it can go into the weights. LoRA and full-parameter adapters were trained on the 157 full-score trajectories (35 tasks) that the closing harness produced on the development tasks. Measured the night after the base cells (the closing-harness base anchor re-measured alongside them: 0.375 against 0.396), they are within the 0.064 line of the base model

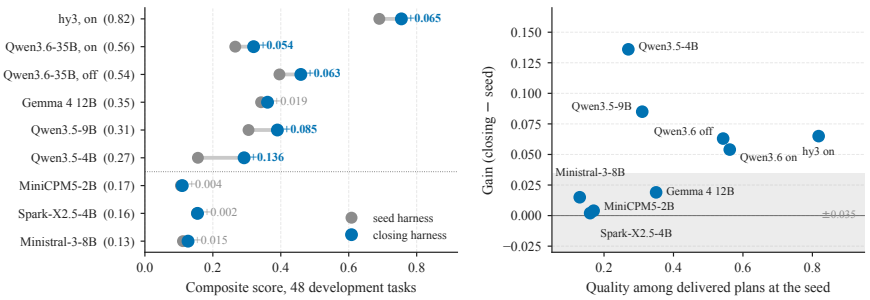

Figure 4: **Harness evolution on the nine DeepPlanning lines reported** (48 development tasks). Left: seed (grey) and closing (blue) score of each line, ordered by the quality of the plans the seed harness delivers (in parentheses; on/off = thinking on or off). Every line is measured under the protocol of Section 4.2, the three early lines and Gemma by a same-night re-measurement of seed and closing compositions (Appendix H). Right: the gain against the seed plan quality; band = ± 0.035, the resolution of the protocol on the Qwen3.5 cells; the Qwen3.6 thinking-on cells are noisier and their +0.053 is within that noise (Appendix H).

on the 117 test tasks under both harnesses (0.344 against 0.327 and 0.331; 0.256 against 0.216 and 0.231); delivery rises further, to 81% and 88%, and accuracy among delivered answers falls, to 0.39 and 0.37. The gain on this benchmark lives in what the model sees, two full pages in the context instead of one, and an adapter that is not shown the second page did not reproduce what the model did with it. Set against Section 5.2, where every accepted edit changed what the model does (tools, prompts, lessons, blockers) and the adapters carried the gain, this gives a reading of when a harness gain can be moved into the weights: gains from edits to what the model writes trained in, gains from edits to what it sees stayed in the harness, and the surface the accepted edits touched is known before any training is run.

5.4 Where the harness lever pays, across nine lines

Figure 4 collects the nine DeepPlanning evolution lines reported in this paper, ordered by the quality of the plans the seed harness delivers. Three were evolved before the protocol of Section 4.2, with one rollout per reading: two on Qwen3.6-35B-A3B (thinking on and off) and one on Tencent hy3 (thinking on); we re-measured their seed and closing compositions on one night with four rollouts each, and the gains re-measure at +0.063 (Qwen3.6 off), +0.053 (Qwen3.6 on) and +0.065 (hy3), all of it delivery; the Qwen3.6 thinking-on cells are noisier than the Qwen3.5 cells (per-rollout standard deviation up to 0.07), so that gain is within its own noise (Appendix H). The seed measurements of Section 3.1, all available before a round is run, separate the nine lines into three groups, and the outcomes follow the grouping. Where the seed harness delivers plans scoring 0.27 or better and F4 is not yet the largest failure class, five lines from a 4B model to hy3, the harness lever paid +0.063 to +0.136 on the development tasks in four and +0.053, within its own noise, in the fifth. Where the delivered plans score 0.13–0.17, three lines, it paid nothing: MiniCPM5-2B, Spark-X2.5-4B and Ministral-3-8B closed within noise of their seeds after five rounds each, although the harness still moved delivery on MiniCPM5-2B (61% to 71%) and F1 on Spark-X2.5-4B (21% to 7%), with the freed trajectories landing in F4 (Table 14): only a model that can write a plan worth scoring can cash the move. Where delivery is saturated and F4 is already the largest class among plans worth scoring, one line, Gemma 4 12B, the rule sends the next move to the weights; the harness bought +0.019, within noise, and what its trajectories then taught the weights is in Appendix H. Two frontier API models run separately by a co-author (Appendix J) point the same way: under an evolved harness Kimi K3 gains +0.058 / +0.049 (dev / held-out) with most of the gain in quality among delivered plans, the 9B pattern, and GLM 5.3 reaches 0.512 / 0.488 (no bare control).

6 Analysis

6.1 What the harness components buy

Appendix F (Tables 11 and 12) reports component ablations on both sizes, each set run in one night (the 9B set on two identical machines, cross-machine offset below 0.03), with the line's closing

harness as the full configuration. For 9B, removing the mid-trajectory checkpoint, the exit gate or the four hard blocks leaves the development score within noise of the full harness (+0.019, +0.017, +0.017); removing the evolved lessons file costs 0.049 on dev and 0.076 on held-out, and removing the ledger, together with the controls that depend on it, costs 0.088 and 0.145. The ledger and the lessons are not additive: with neither or with the ledger alone the score is within noise of the bare model; with both it is +0.076 / +0.092 above it. For 4B, removing the loop's own middleware or its three policy rewrites leaves the score within noise; removing the three invented tools costs 0.071 / 0.090 and removing the lessons file 0.067 / 0.120; removing the exit gate costs nothing on dev but 0.077 on held-out, where delivery drops from 88% to 73%. Tools and lessons account for the largest losses; enforcement is within noise except the 4B exit gate, which holds held-out delivery.

6.2 Why the loop stops at F4

The evolution loop can only optimise what it is shown, and what it is shown is the evidence package. Four properties of it bound what the proposer can see. (a) The package leads with intervention process metrics, the components the ablations show do not move the score; the F1–F4 counts come after them, and the scorer's per-check failure composition was never aggregated into the package. (b) The proposal request is a single call with no tools. (c) The keys that define which metrics are observed live outside the four directories the loop may edit. (d) The accepted edits target what the package makes visible: the 4B tools and prompts raise delivery and cut loops, the 9B lessons cut loops, and no edit changed the rate at which the 9B agent submits several drafts of a plan (25.5% before, 25.0% after), a failure the scorer sees but the package never surfaced. The ceiling is a property of this loop's evidence package, not of harness evolution in general; a first attempt to surface the failure composition to the proposer (Gemma 4 12B, Appendix H) did not produce a resolvable gain.

7 Discussion

A diagnose-then-intervene rule. Measure the noise floor first, then the delivery rate, the quality among delivered plans and the F1–F4 composition on a development set, and branch. (1) Delivered plans score below about 0.18: the harness lever paid nothing on the three such models, moving delivery and F1 while the freed trajectories landed in F4; we did not train them. (2) Delivery well below saturation or F3 dominant: evolve the harness; its gains transferred to held-out tasks in both Qwen lines. (3) Delivery saturated and F4 dominant: train the weights; on 9B the adapter alone was worth as much as the five-round evolution line on held-out, and on 4B the two levers stack. (4) F4 residual after both: the loop as run here cannot see it; expose the scorer's failure composition to the proposer (Section 6.2). (5) A harness gain in hand: read what the accepted edits changed; where they changed what the model does, the adapter carried the gain (Section 5.2), and where the gain came from what the model sees, it stayed in the harness (Section 5.3).

Generality. Across the nine lines of Section 5.4, the seed measurements (delivery, quality, failure composition) separate the five lines on which the harness lever paid (+0.05 to +0.14, the smallest within its own noise) from the three on which it paid nothing; on WebArena-Lite the harness gain held on 117 unseen tasks (+0.088) and training on its trajectories did not add to it.

Limitations. Applying the taxonomy needs trajectory-level process signals and a task-level scorer. The 4B ablation cells were run with two rollouts except the exit-gate cell, and the placebo control exists for 9B only. We do not compare the loop with a matched-budget test-time search baseline (Wang et al., 2026b). The benchmark's plan-to-JSON conversion is an LLM call that inflates scores by about 0.025 on the sample we audited (Appendix I); every cell uses the same converter. Two comparisons cross nights: the base-model cells under the evolved harness on held-out tasks in Tables 1 and 5 are closing anchors from two nights earlier (two rollouts for 4B, so its held-out stacking gain is read against a 0.05 line), and the WebArena-Lite adapters were measured the night after their base cells (Table 13); every other comparison is same-night.

Reproducibility Statement

Every number in this paper is read from a persisted evaluation run (one directory per run, with the configuration, per-task scores and full trajectories) or from the per-round history file that the harness-evolution loop writes after each verdict, except the Kimi K3 and GLM 5.3 cells of Appendix J, which a co-author ran on a separate machine and whose run directories will be released with the code. The benchmark, the task split and the models are described in Section 4 and the labelling rules for F1–F4 in Section 3.1; the evolution protocol and the accept rule are in Section 4.2; the noise floor and the 4-rollout discrimination threshold are in Appendix A; the per-round evolution logs are in Appendices B and C; the training recipe, the placebo construction, the rank ablation and the full-parameter comparison are in Section 4.3 and Appendix E; the component ablations are in Appendix F. The harness-evolution loop, the evaluation wrapper, the evolved harness definitions, the fine-tuning configurations and the run logs will be released with the camera-ready version.

Ethics Statement

This work studies the mechanics of LLM agents on a travel-planning benchmark. It does not involve human subjects, personal data, or any application that could cause direct harm. The one relevant concern is the plan-to-JSON conversion bias documented in Appendix I: the benchmark's default converter silently selects one draft when the agent submits several and inflates scores by about 0.025. We reported this to the benchmark maintainers together with a fix and use the same converter across all conditions and report the multi-draft rate that governs the bias.

AI use statement

Generative AI tools (LLM-based coding and writing assistants) were used as follows. Among the tasks with required disclosure: hypothesis refinement and methodology feedback (discussing experiment design and intermediate results); method implementation (the evaluation wrapper, the evolution loop and the analysis scripts were written with AI coding assistants, then reviewed and tested by the authors); translation and drafting between working notes in Chinese and this manuscript; and results interpretation (every conclusion was checked by the authors against the persisted run logs). The fine-tuning trajectories are model-generated data, as described in the paper; no other synthetic data were generated. The two identities of Section 3.2 (a standard maximal inequality for sub-Gaussian variables, cited, and an exact score decomposition) were drafted with AI assistance and verified by the authors; no other theoretical development or proof assistance was used. Dataset cleaning and qualitative analysis of surveys do not apply. Among the tasks with recommended disclosure: drafting and polishing text, suggesting structure, writing figure code, and locating and formatting references. LLMs are also part of the object of study (the fine-tuning trajectories come from the models under evaluation, the loop's proposal model is an LLM, and the benchmark's plan-to-JSON converter is an LLM parser), as described in the text. All AI-assisted text, code and citations were reviewed by the authors, every reported number was traced to a run log, and we take responsibility for the final content.

References

Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford University Press, 2013.

Mert Cemri, Melissa Z. Pan, Shuyi Yang, Lakshya A. Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. Why do multi-agent LLM systems fail? *arXiv preprint arXiv:2503.13657*, 2025.

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

Xiwei Chen. State-aware runtime for long-horizon LLM agents: A conceptual framework and research agenda. Cambridge Open Engage preprint, 2026.

Zhengyu Chen, Teng Xiao, Huaisheng Zhu, Yige Yuan, Luan Zhang, and Jingang Wang. Co-harness: Co-evolving harnesses and model weights for LLM agents. *arXiv preprint arXiv:2607.22688*, 2026.

Darshan Deshpande, Varun Gangal, Hersh Mehta, Jitin Krishnan, Anand Kannappan, and Rebecca Qian. TRAIL: Trace reasoning and agentic issue localization. *arXiv preprint arXiv:2505.08638*, 2025.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized LLMs. In *NeurIPS*, 2023. arXiv:2305.14314.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

Edward J. Hu, Yelong Shen, Phillip Wallis, et al. Lora: Low-rank adaptation of large language models. In *ICLR*, 2022. arXiv:2106.09685.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. *arXiv preprint arXiv:2408.08435*, 2024.

Sayash Kapoor, Benedikt Stroebl, Zachary S. Siegel, Nitya Nadgir, and Arvind Narayanan. AI agents that matter. *arXiv preprint arXiv:2407.01502*, 2024.

Haechan Kim, Yoonho Lee, Gisang Lee, Chelsea Finn, and Kangwook Lee. WHALE: A simple recipe for joint harness-weight optimization. *arXiv preprint arXiv:2609.00196*, 2026a.

Kyungmin Kim, Youngbin Choi, Seoyeon Lee, Suhyeon Jun, Dongwoo Kim, and Sangdon Park. The interplay of harness design and post-training in LLM agents. *arXiv preprint arXiv:2606.25447*, 2026b.

Jiahang Lin, Shichun Liu, Chengjun Pan, Lizhi Lin, Shihan Dou, Zhiheng Xi, Xuanjing Huang, Hang Yan, Zhenhua Han, Tao Gui, and Yu-Gang Jiang. Agentic harness engineering: Observability-driven automatic evolution of coding-agent harnesses. *arXiv preprint arXiv:2604.25850*, 2026.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. Agentbench: Evaluating LLMs as agents. In *ICLR*, 2024. arXiv:2308.03688.

Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. Agentboard: An analytical evaluation board of multi-turn LLM agents. In *NeurIPS*, 2024. arXiv:2401.13178.

Evan Miller. Adding error bars to evals: A statistical approach to language model evaluations. *arXiv preprint arXiv:2411.00640*, 2024.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. Memgpt: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*, 2023.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Wenyi Zhao, Yu Yang, Xinyue Yang, Jiadai Sun, Shuntian Yao, Tianjie Zhang, Wei Xu, Jie Tang, and Yuxiao Dong. WebRL: Training LLM web agents via self-evolving online curriculum reinforcement learning. *arXiv preprint arXiv:2411.02337*, 2024.

Salman Rahman, Yubin Kim, Mihir Parmar, A. Ali Heydari, Genglin Liu, Simon A. Lee, Weizhi Zhang, Arian Hosseini, Ahmed A. Metwally, Yuzhe Yang, Baharan Mirzasoleiman, Xin Liu, Pavel Izmailov, Saadia Gabriel, Mark Malhotra, Shwetak Patel, Daniel McDuff, and Hamid Palangi. Locating hidden failures makes long-horizon agents more reliable. *arXiv preprint arXiv:2609.17930*, 2026.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Hui-Ze Tan, Xiao-Wen Yang, Hao Chen, Jie-Jing Shao, Yi Wen, Yuteng Shen, Weihong Luo, Xiku Du, Lan-Zhe Guo, and Yu-Feng Li. Hindsight credit assignment for long-horizon LLM agents. *arXiv preprint arXiv:2603.08754*, 2026.

Su Wang, Pin Qian, Yifan Lin, Jingzhou Xu, Yihang Chen, Xiaochong Jiang, Lifei Liu, and Haoran Yu. Phantom guardrails: When self-improving agent harnesses fix failures that never happened. *arXiv preprint arXiv:2607.13083*, 2026a.

Yike Wang, Huaisheng Zhu, Zhengyu Hu, Yige Yuan, Zhengyu Chen, Shakti Senthil, Hannaneh Hajishirzi, Yulia Tsvetkov, Pradeep Dasigi, and Teng Xiao. Rethinking the evaluation of harness evolution for agents. *arXiv preprint arXiv:2607.12227*, 2026b.

Shengguang Wu, Hao Zhu, Yuhui Zhang, Xiaohan Wang, and Serena Yeung-Levy. Automem: Automated learning of memory as a cognitive skill. *arXiv preprint arXiv:2607.01224*, 2026.

Peng Xia, Rujun Han, Zifeng Wang, Yanfei Chen, Yufan Zhuang, Yoonho Lee, Chengsong Huang, Han Yu, Zhongying CuiZhu, Yifei Ming, Huaxiu Yao, Burak Gokturk, Tomas Pfister, and Chen-Yu Lee. RRSI: Regularized recursive self-improvement of agent harnesses. *arXiv preprint arXiv:2609.24972*, 2026.

Yang Xu, Jiefu Zhang, Haixiang Sun, Zihan Zhou, Tianyu Cao, and Vaneet Aggarwal. Towards reliable LLM evaluation: Correcting the winner's curse in adaptive benchmarking. *arXiv preprint arXiv:2605.05973*, 2026.

Jiangze Yan, Yi Shen, Wenjing Zhang, Jieyun Huang, Zhaoxiang Liu, Ning Wang, Kai Wang, and Shiguo Lian. HiMPO: Hindsight-informed memory policy optimization for less-entangled credit in long-horizon agents. *arXiv preprint arXiv:2606.16285*, 2026.

Haoran Ye, Yuxing Lu, Haonan Dong, Zhaochen Su, and Guojie Song. Harness-zero: Harness distillation via agent-as-harness. *arXiv preprint arXiv:2609.24974*, 2026.

Ping Yu, Jing Xu, Jason Weston, and Ilia Kulikov. Distilling system 2 into system 1. *arXiv preprint arXiv:2407.06023*, 2024.

Zhou Yu, Bin Bi, Shiva Kumar Pentyala, Shubham Mehrotra, Sougata Chaudhuri, Shilpa Bhagavath, Zeyuan Chen, Ran Xu, Phil Mui, James Zhu, and Sitaram Asur. Co-evolving harnesses and models: On-policy correction helps weaker models catch up where imitation fails. *arXiv preprint arXiv:2609.09134*, 2026.

Ruiqing Yue, Yu Cui, Zhuoyu Sun, Sicheng Pan, Xianhong Xue, Tingyu Li, Ting Li, Wenzhuo Zhu, Yi Chen, Yifei Liu, Baohan Huang, Zhe Cui, Haibin Zhang, and Cong Zuo. Ecdysis: Efficient and effective training of runtime harnesses for LLM agents. *arXiv preprint arXiv:2609.11677*, 2026.

Eric Zelikman, Yuhuai Wu, Jesse Dodge, and Noah D. Goodman. Star: Bootstrapping reasoning with reasoning. In *NeurIPS*, 2022. arXiv:2203.14465.

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.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xionghui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. AFlow: Automating agentic workflow generation. *arXiv preprint arXiv:2410.10762*, 2024.

Luan Zhang, Ruochen Zhou, Dandan Song, Zhengyu Chen, Yuhang Tian, Jun Yang, Huipeng Ma, Chenhao Li, Guangyuan Feng, Xudong Li, Yizhou Jin, and Yan Xu. HarnessCompass: Guiding automatic harness evolution toward generalizable and effective agent harnesses. *arXiv preprint arXiv:2608.01918*, 2026a.

Yinger Zhang, Shutong Jiang, Renhao Li, Jianhong Tu, Yang Su, Lianghao Deng, Xudong Guo, Chenxu Lv, and Junyang Lin. Deepplanning: Benchmarking long-horizon agentic planning with verifiable constraints. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7377–7407, 2026b. arXiv:2601.18137.

Yukun Zhang, Kemu Xu, and Yishen Chen. How do agent harnesses create value? planning information and release control in stateful LLM agents. *arXiv preprint arXiv:2609.20474*, 2026c.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. In *ICLR*, 2024. arXiv:2307.13854.

Kunlun Zhu, Zijia Liu, Bingxuan Li, Muxin Tian, Yingxuan Yang, Jiaxun Zhang, Pengrui Han, Qipeng Xie, Fuyang Cui, Weijia Zhang, Xiaoteng Ma, Xiaodong Yu, Gowtham Ramesh, Jialian Wu, Zicheng Liu, Pan Lu, James Zou, and Jiaxuan You. Where LLM agents fail and how they can learn from failures. *arXiv preprint arXiv:2509.25370*, 2025.

A Noise Floor and Measurement Protocol

Table 2: Smallest difference between two cells that we treat as resolvable, as a function of the number of rollouts averaged per cell, given a per-rollout standard deviation of 0.023 (median over 26 same-night DeepPlanning cells; worst cell 0.042). The paper uses four rollouts per cell.

Rollouts per cell	Resolvable difference
1	0.07
2	0.05
4	**0.035**
8	0.025
16	0.02

The per-rollout standard deviation was measured on 26 same-night cells from the comparison in Table 1 and its companion adapters (4B and 9B, 48 or 72 tasks, temperature 0.7, four rollouts each): median 0.023, worst 0.042. The range across the four rollouts of a single cell has median 0.055 and reaches 0.094 (a 4B seed-harness cell read 0.171, 0.096, 0.176 and 0.190). Table 2 gives the smallest difference between two cells that we treat as resolvable for a given number of rollouts per cell; the paper uses four rollouts and 0.035 everywhere unless stated. The house rule inside the evolution loop is stricter in one direction: a candidate whose 4-rollout mean falls more than 0.02 below the anchor is rejected, which trades some false rejections for protection against ratchet slippage (Appendix G). On WebArena-Lite the per-rollout standard deviation is about 0.045, so the resolvable difference at four rollouts is 0.064. Two configurations of the same model evaluated on different nights, even with the serving stack pinned, differed by up to 0.05, which is why comparisons in this paper are same-night unless marked.

B Full 13-Round Evolution Log (Qwen3.5-4B)

Table 3 lists every candidate with its own reading and the fresh anchor measured after the round. Each round proposed two candidates; 18 of 26 passed the judge, one per round was retained (10 in

Table 3: Qwen3.5-4B evolution line, round by round. Each round proposed two candidates; the retained edit is listed with the harness surface it touched, the other candidate with its verdict. Anchor = fresh 4-rollout mean of the retained harness on the 48 development tasks, measured after the round; F3 = share of loop failures in that anchor. Rounds 4, 5 and 12 rejected both candidates and kept the previous anchor.

Round	Retained edit	Surface	Other candidate (verdict)	Anchor	F3
0	seed harness			0.156	64.1%
1	plan_submission_format_fix	prompts	malformed_plan_tag_middleware (accepted, not retained)	0.204	66.7%
2	plan_emission_lessons (new lessons file)	prompts	plan_tool_call_interceptor (accepted, not retained)	0.221	42.2%
3	gate_self_check_activation	policies	plan_envelope_rescue_middleware (accepted, not retained)	0.225	38.0%
4	none		plan_format_middleware, plan_submission_tool (both rejected)	0.225	
5	none		coordinate_fallback_lesson, plan_truncation_guard (both rejected)	0.225	
6	coordinate_lookup_fallback_tool	tools	plan_emission_force_syntax_fix (accepted, not retained)	0.247	35.9%
7	citation_format_normalizer	middleware	plan_truncation_rescue_middleware (rejected)	0.243	33.3%
8	plan_completeness_checker_tool	tools	budget_constraint_checker_tool (accepted, not retained)	0.282	35.4%
9	checkpoint_fuse_activation	policies	plan_submission_format_enforcement (accepted, not retained)	0.281	38.5%
10	plan_submission_checkpoint	middleware	coordinate_query_fallback_middleware (accepted, not retained)	0.288	37.8%
11	plan_truncation_detection	middleware	citation_format_normalizer_enhancement (accepted, not retained)	0.305 / 0.269[†]	33.3%
12	none		threshold_reduction, fabricated_info_gate_blocker (both rejected)	0.269	
13	flight_time_sorter_tool	tools	citation_bracket_enforcement (rejected)	**0.292**	37.0%

[†] 0.305 with the 131k-token serving context; the same harness re-anchored at 0.269 after the context window was raised to 262k tokens, and all later rounds use the 262k setting.

total) and 8 were rejected, with rounds 4, 5 and 12 rejecting both candidates. The three tool edits (rounds 6, 8, 13) and the lessons file (round 2) are the components that the ablation in Appendix F finds load-bearing. Round 8, a plan-completeness checker tool, is the largest single step (+0.039 on the fresh anchor). One accepted edit turned out to be inert on inspection (the round-3 policy change activated a self-check whose prompt file did not exist) and another silently removed the seed harness's twenty-call brake on identical calls; the judge cannot distinguish an inert edit from a neutral one, which is one reason we report fresh anchors rather than candidate readings. After round 11 the serving context window was raised from 131k to 262k tokens and the current harness was re-anchored (0.305 to 0.269); the final gain is computed from the re-anchored value.

C Full 5-Round Evolution Log (Qwen3.5-9B)

Table 4: Qwen3.5-9B evolution line, candidate by candidate (5 rounds, 10 candidates, 3 accepted). Reading = the candidate's own 4-rollout mean when judged; anchor = fresh 4-rollout mean of the retained harness after the round. All three accepted edits modify the distilled lessons file; every proposed tool was rejected because the agent never called it.

Round	Candidate	Surface	Reading	Verdict	Anchor after round
0	seed harness				0.305
1	plan_emission_format_lessons	prompts	0.380	accepted	0.377
1	query_constraint_extractor_tool	tools	0.335	rejected (never called)	
2	hotel_location_lessons	prompts	0.379	accepted	0.366
2	hotel_location_validator_tool	tools	0.399	rejected (never called)	
3	standalone_plan_emission_lessons	prompts	0.374	accepted	**0.390**
3	plan_submission_reminder_tool	tools	0.378	rejected (never called)	
4	query_constraint_extractor_v2	tools	0.372	rejected (-0.018 vs anchor)	0.390
4	route_fabrication_lessons	prompts	0.357	rejected (-0.033 vs anchor)	
5	fabricated_data_verification_lessons	prompts	0.361	rejected (-0.029 vs anchor)	0.390
5	plan_emission_force_threshold_reduction	middleware	0.359	rejected (-0.031 vs anchor)	

Table 4 gives the same record for the 9B line. The 9B line accepted three candidates in five rounds, all of them edits to the distilled lessons file (plan-emission format, hotel-location checks, standalone plan emission); every tool the proposer wrote was rejected because the agent never called it, and the last four candidates read 0.018–0.033 below the anchor and were rejected. The closing harness is therefore the seed harness plus one lessons file, which is why the 9B ablation (Appendix F) can isolate the whole gain of the line in one cell. An earlier 9B line that we do not report locked its anchor on a single lucky evaluation and then rejected eight consecutive candidates; the fresh-anchor protocol was introduced after it.

D Evolution Summary Tables and Per-Round Failure Composition

Table 5: Harness evolution before and after. Score = 4-run mean composite; delivery rate = fraction of tasks on which a parsable plan was returned. Seed harness and final evolved harness were re-evaluated outside the loop on the same machine, on the 48 development tasks and the 72 held-out tasks; the evolved-harness held-out cells are the lines' closing anchors measured two nights before the other three cells (4B: 2 rollouts). The 4B line ran 13 rounds (10 accepted edits); the 9B line ran 5 rounds (3 accepted edits). The 4-run discrimination threshold is 0.035; smaller differences are within noise.

		Development (48 tasks)			Held-out (72 tasks)		
Model	Quantity	Seed	Evolved	Gain	Seed	Evolved	Gain
Qwen3.5-4B	Score	0.158	0.301	**+0.143**	0.164	0.297	**+0.133**
Qwen3.5-4B	Delivery rate	52%	90%	+38 pp	55%	90%	+35 pp
Qwen3.5-9B	Score	0.348	0.404	**+0.056**	0.323	0.442	**+0.119**
Qwen3.5-9B	Delivery rate	96%	98%	+2 pp	97%	99%	+2 pp

Table 6: Failure composition of the seed harness and of the final evolved harness on the 48 development tasks (share of trajectories; fresh anchors, 4 rollouts each except the 9B seed row, which is the 6-rollout line baseline; labels as defined in Section 3.1). Rows sum to 100% up to rounding. The largest change in both lines is the fall in loop failures (F3); F4 is not reduced and ends as the largest failure class for 9B and the second-largest, after the residual F3, for 4B.

Model	Harness	Healthy	F1 evidence	F2 harness friction	F3 execution control	F4 planning capacity
Qwen3.5-4B	Seed	14.3%	8.6%	2.9%	**64.1%**	9.9%
Qwen3.5-4B	Evolved (round 13)	31.8%	3.1%	0.0%	**37.0%**	**28.1%**
Qwen3.5-9B	Seed	26.7%	5.2%	2.1%	**38.9%**	27.1%
Qwen3.5-9B	Evolved (round 3)	40.6%	17.7%	0.0%	16.7%	**25.0%**

Table 7: Score decomposition on the 48 development tasks: composite score = delivery rate × mean score among delivered plans. Delivery = the trajectory returned a plan the scorer could parse. Seed and evolved anchors of each evolution line (4 rollouts; the 9B seed row is the 4-rollout subset of the 6-rollout line baseline). MiniCPM5-2B is included as the case in which the two factors cancel.

Model	Harness	Composite score	Delivery rate	Score among delivered
Qwen3.5-4B	Seed	0.156	59%	0.266
Qwen3.5-4B	Evolved (13 rounds)	0.292	**88%**	0.332
Qwen3.5-9B	Seed	0.299	97%	0.309
Qwen3.5-9B	Evolved (5 rounds)	0.390	99%	**0.392**
MiniCPM5-2B	Official benchmark loop	0.047	33%	0.141
MiniCPM5-2B	Seed	0.106	61%	0.173
MiniCPM5-2B	Evolved (5 rounds)	0.110	71%	0.156

Table 5 re-measures the seed and closing harness of both lines on both splits; Tables 6 and 7 give the seed and closing compositions and the delivery / quality decomposition referred to in Sections 3.1 and 5.1. Figure 3 plots, for each round in which an edit was accepted, the failure composition of the fresh anchor measured after that round.

E Training Details, Rank Ablation and Full-Parameter Fine-Tuning

All adapters use rank 32, $\alpha = 64$, dropout 0.05, AdamW with weight decay 0.01, gradient clipping at 1.0, a constant learning rate of 10^{-4} and sixteen step samples per optimiser step; the adapters in Table 1 were trained for two epochs and taken at the end of the first, because the second epoch memorises. The placebo permutes the assistant outputs across samples while leaving every input unchanged, so the adapter sees the same prompts, the same harness structure and the same number

Table 8: Second textbook (198 trajectories over 42 tasks for 4B; 200 over 44 for 9B; one epoch), seed harness, same machine and night per block, 4 rollouts. Top: LoRA versus full-parameter fine-tuning for both sizes. Bottom: rank ablation on 9B. Each adapter in the bottom block is scored against the base model measured on its own machine on the same night (the base drifts by up to 0.035 within seven hours on one machine), so its gains are comparable across ranks but are not differences from the 0.306 / 0.370 row above, and the rank-64 absolute scores are not comparable to the other rows.

Model	Weights	Dev score	Held-out score	Dev gain over base	Held-out gain over base
Qwen3.5-4B	Base	0.156	0.150		
Qwen3.5-4B	LoRA qkvo	0.193	0.212	+0.037	+0.062
Qwen3.5-4B	LoRA qkvo+MLP	0.280	0.267	+0.124	+0.117
Qwen3.5-4B	Full-parameter	**0.294**	**0.311**	+0.138	+0.161
Qwen3.5-9B	Base	0.306	0.370		
Qwen3.5-9B	LoRA qkvo+MLP	0.426	0.442	+0.120	+0.072
Qwen3.5-9B	Full-parameter	**0.438**	**0.484**	+0.132	+0.114
Qwen3.5-9B	LoRA rank 16, $\alpha = 32$	0.425	0.424	+0.091	+0.058
Qwen3.5-9B	LoRA rank 32, $\alpha = 64$	0.419	0.401	+0.085	+0.035
Qwen3.5-9B	LoRA rank 64, $\alpha = 128$ (second machine)	0.415	0.430	+0.116	+0.085
Qwen3.5-9B	Full-parameter (same night as ranks)	0.447	0.456	+0.139	+0.093

Table 9: Failure composition of the base model and the LoRA adapters under the seed harness, same machine and night as Table 1 (share of trajectories, 4 rollouts; labels as in Section 3.1; rows sum to 100% up to rounding). The 4B adapter moves delivery and F3; the 9B adapter moves F4.

Model	Adapter	Split	Delivery	Healthy	F1 evidence	F2 harness friction	F3 execution control	F4 planning capacity
Qwen3.5-4B	none	dev 48	52%	15.6%	10.9%	3.6%	**57.8%**	12.0%
Qwen3.5-4B	LoRA qkvo	dev 48	**78%**	35.4%	8.9%	2.1%	**39.6%**	14.1%
Qwen3.5-4B	LoRA qkvo+MLP	dev 48	67%	26.6%	12.0%	4.7%	51.0%	5.7%
Qwen3.5-4B	none	held-out 72	55%	15.3%	9.0%	3.5%	**64.6%**	7.6%
Qwen3.5-4B	LoRA qkvo	held-out 72	**78%**	30.6%	8.7%	2.1%	**45.1%**	13.5%
Qwen3.5-4B	LoRA qkvo+MLP	held-out 72	62%	30.9%	8.7%	3.5%	48.6%	8.3%
Qwen3.5-9B	none	dev 48	96%	34.4%	8.9%	3.6%	30.2%	**22.9%**
Qwen3.5-9B	LoRA qkvo	dev 48	95%	30.2%	10.9%	1.6%	43.2%	13.5%
Qwen3.5-9B	LoRA qkvo+MLP	dev 48	94%	45.8%	9.9%	1.6%	35.4%	**7.3%**
Qwen3.5-9B	none	held-out 72	97%	28.1%	6.2%	3.8%	33.7%	**28.1%**
Qwen3.5-9B	LoRA qkvo	held-out 72	98%	38.5%	10.4%	1.4%	33.7%	16.0%
Qwen3.5-9B	LoRA qkvo+MLP	held-out 72	96%	52.8%	10.1%	1.7%	30.9%	**4.5%**

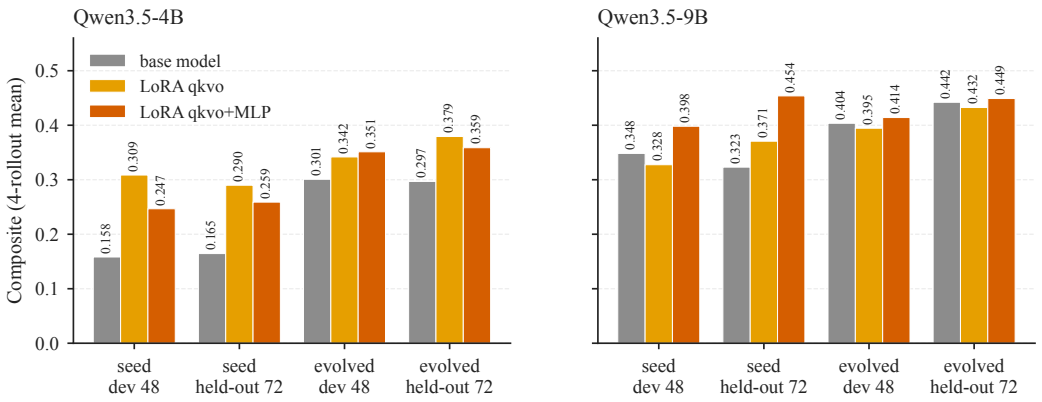

Figure 5: **Base model and LoRA adapters under the seed and the evolved harness**, same machine, 4-rollout means (Table 1; provenance of the evolved-harness held-out base cells in its caption). On the seed harness the adapter alone recovers as much as harness evolution; on the evolved harness the adapter adds to the 4B score, and for 9B the adapter alone already reaches the evolved-harness level.

Table 10: Placebo control on Qwen3.5-9B: the real adapter and a placebo adapter trained with the identical recipe (rank 32, $\alpha = 64$, qkvo+MLP, one epoch) on the same trajectories, except that the placebo's assistant outputs are permuted across samples. All cells on the same machine and night, seed harness, 4 rollouts. Rounds = mean number of agent turns per trajectory, pooled over both splits; the placebo hit the 100-turn cap in 9 of 96 development and 44 of 288 held-out trajectories.

	Development (48 tasks)		Held-out (72 tasks)		
Weights	Score	Delivery rate	Score	Delivery rate	Rounds per trajectory
Base (no adapter)	0.333	98%	0.379	99%	19
Real LoRA	0.399	not recorded	0.426	not recorded	not recorded
Placebo LoRA	**0.175**	86%	**0.152**	76%	52–65
Placebo minus base	−0.158	−12 pp	−0.227	−23 pp	+33 to +46

of steps. The rank ablation and the full-parameter comparison (Table 8) use a larger second textbook (198 trajectories over 42 tasks for 4B, 200 over 44 for 9B, one epoch) collected the same way; full-parameter training uses learning rate 2×10^{-6} with 5% warm-up and cosine decay, fully sharded, and a higher learning rate without warm-up diverged for both sizes. Table 9 and Figure 6 give the failure composition behind finding (iv) of Section 5.2. On the evolved harness the 9B adapters cut F4 as well (qkvo+MLP: 22% to 6% on dev, 19% to 6% on held-out), but the freed share reappears as F3 on dev and as F1 on held-out rather than as healthy trajectories, and the mean score is unchanged: the adapter and the evolved lessons file move the same failure class, which is the substitution reported in Section 5.2.

F Component Ablations

Both ablations remove components from the closing harness of the corresponding line. The 9B cells were run on two identical machines on one night (full, minus ledger, minus gate, minus checkpoint on one; minus lessons, minus hard blocks and bare on the other), with a cross-machine offset of at most 0.03 checked by re-running the full configuration on both. The 4B cells were run on one machine in one boot with two rollouts per cell (read as resolvable above 0.05), except the exit-gate cell, which was re-run with four. Removing the ledger in the 9B harness also removes the gate and checkpoint that depend on it. In the 9B minus-ledger cell, 66 of 95 final plans still contained a reconciliation section and 61 cited ledger entries that did not exist. The paired comparison quoted in Section 5.1 restricts each pair of 4B cells to the tasks both deliver: full versus bare is +0.011 over 35 development tasks and +0.022 over 48 held-out tasks; the seed kit versus bare is −0.027 / −0.009.

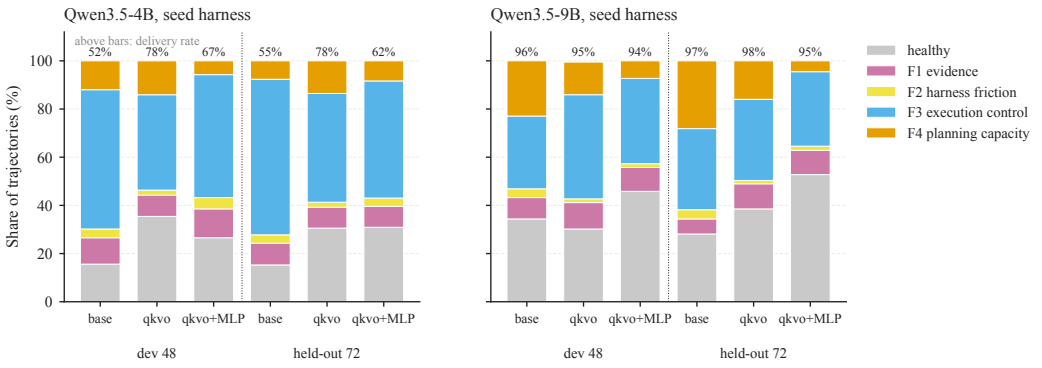

Figure 6: **Failure composition of the base model and the LoRA adapters under the seed harness** (the cells of Table 9; numbers above the bars are delivery rates). The 4B adapters raise delivery and shrink the loop class F3; the 9B adapters shrink the content class F4.

Table 11: Component ablation on Qwen3.5-9B (closing harness of the 9B line; seven cells, 4 rollouts each, two identical machines on one night, cross-machine offset ≤ 0.03). "Removes ledger" also removes the exit gate and the checkpoint, which depend on it. Differences below 0.035 are within noise.

Cell	Removes	Dev score	Held-out score	Dev vs full	Held-out vs full	Dev delivery	Dev score among delivered
Full	nothing	0.366	0.438			99%	0.368
− Checkpoint	mid-trajectory checkpoint	0.385	0.440	+0.019	+0.002		
− Gate	exit gate	0.384	0.410	+0.017	−0.028		
− Hard blocks	four hard blocks	0.383	0.432	+0.017	−0.005		
− Lessons	evolved lessons file	0.317	0.362	−0.049	−0.076		
Bare	everything	0.290	0.346	−0.076	−0.092	85%	0.340
− Ledger	state ledger (with gate and checkpoint)	0.278	0.292	−0.088	−0.145	98%	0.284

Table 12: Component ablation on Qwen3.5-4B (closing harness of the 4B line; ten cells, one machine, one boot, 2 rollouts per cell except the exit-gate cell with 4). With two rollouts, differences below 0.05 are within noise. "Seed kit" keeps the seed harness and removes every product of evolution.

Cell	Removes	Dev score	Held-out score	Dev vs full	Held-out vs full
Full	nothing	0.264	0.314		
− Evolved middleware	citation auto-fix, submission nag	0.310	0.313	+0.046	−0.001
− Evolved rewrites	three policy and prompt rewrites	0.302	0.287	+0.038	−0.027
− Checkpoint	mid-trajectory checkpoint	0.270	0.330	+0.007	+0.016
− Gate (4 rollouts)	exit gate	0.290	0.237	+0.026	−0.077
− Ledger	state ledger	0.253	0.254	−0.011	−0.060
− Lessons	evolved lessons file	0.197	0.194	−0.067	−0.120
− Tools	three invented tools	0.193	0.224	−0.071	−0.090
Seed kit	all products of evolution	0.160	0.187	−0.104	−0.127
Bare	everything	0.150	0.132	−0.114	−0.183

The composition of violated scorer checks (14 check types) has the same shape in every 4B cell, and the rate of trajectories that submit several drafts falls from 8.9% (bare) to 1.0% (full).

G Measurement Pitfalls and Mitigations

We document three pitfalls encountered in our own measurement process.

Winner's curse. Selecting the best of K candidates inflates the selected reading (Xu et al., 2026). With per-rollout $\sigma \approx 0.023$ and two candidates per round, an accepted candidate's own reading overstates its effect; in the 4B line the candidate readings at rounds 8, 10 and 11 were 0.316, 0.327 and 0.323 while the fresh anchors of the same harnesses were 0.282, 0.288 and 0.305. We report

anchors, never candidate readings; Figure 7 plots every retained candidate's reading against the fresh anchor of the same harness.

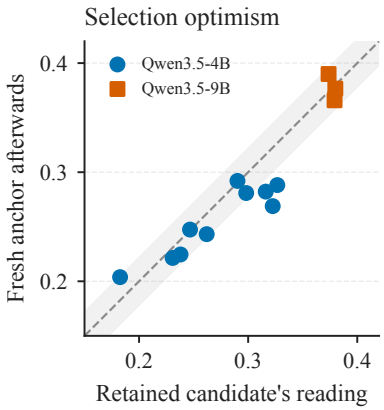

Figure 7: **Selection optimism in the two evolution lines** (Proposition 1). Each point is a round with an accepted edit: the retained candidate's own 4-rollout reading against the fresh anchor of the same harness measured afterwards; band $= \pm\sigma$ around the diagonal. Points below the diagonal are readings that overstated the harness.

Ratchet slippage. In a sequence of accepts, a lucky round sets the next reference above the true value, and a subsequent neutral edit then reads as a regression and is rejected. Our first 9B line suffered a severe form of this: one outlier accept locked the anchor high and eight consecutive candidates were rejected within noise of it. The mitigation is the fresh anchor: after every accept, the new harness is re-measured before the next candidate is judged.

Inert accepts. An edit that never executes (a middleware wired to the wrong round index, a policy that references a missing file) reads as neutral and can pass the judge, a cousin of the phantom guardrails of Wang et al. (2026a), which fix failures that never happened. One such edit was accepted in the 4B line and three in the WebArena-Lite line. We now require, before an edit is judged, evidence that its code path fired at least once in the evaluation run.

H Further Lines in Detail

Table 13: WebArena-Lite with Qwen3.5-9B: seed harness versus the closing harness of the 13-round line, and the adapters trained on the closing harness's trajectories (4 rollouts per cell; the resolvable difference on this benchmark is 0.064). Delivery = the agent returned an answer; healthy, F3 and F4 = share of trajectories labelled as in Section 3.1; accuracy = judged-correct share among delivered answers. Base-model cells: one night, one site host; adapter cells: the following night on that host and a clone, with the closing-harness base anchor on the development tasks re-measured alongside them (0.375; 0.396 the night before).

		Dev score	Test, 117 unseen tasks					
Harness	Weights	(48 tasks)	Score	Delivery	Healthy	F3	F4	Accuracy among delivered
Seed	Base	0.271	0.256	49%	23.1%	45.9%	25.6%	0.47
Evolved	Base	0.375	**0.344**	77%	**33.8%**	40.2%	18.2%	0.44
Seed	LoRA	0.253	0.216	42%	20.3%	52.6%	19.4%	0.49
Seed	Full-parameter	0.250	0.231	45%	20.7%	47.0%	23.9%	0.46
Evolved	LoRA	0.349	0.327	81%	31.4%	38.7%	17.1%	0.39
Evolved	Full-parameter	0.406	0.331	**88%**	32.9%	35.7%	21.2%	0.37

Figure 4 (Section 5.4) summarises the nine DeepPlanning lines reported, and Table 14 gives the failure composition of the three smaller non-Qwen ones; the paragraphs below give the mechanism behind each number.

Two further models, three lines (August 2026). Qwen3.6-35B-A3B was run with thinking off from the seed harness and then with thinking on from the thinking-off line's closing harness, and Tencent hy3 with thinking on from the seed harness, one rollout per reading and the judge comparing each candidate with the previous round's reading; the 4-rollout protocol and fresh anchors of Section 4.2 came later, so the line readings below are single-rollout and the numbers reported in Section 5.4 are the same-night re-measurements at the end of this paragraph.

Qwen3.6-35B-A3B, thinking off: 25 rounds, 6 accepts, 0.404 to 0.496, with the no-submission rate falling from 23% to zero and every accept a selective blocker or a tool. Thinking on: 19 rounds, 5 accepts, 0.387 to 0.535, every accept an instruction-layer edit and four of the five on how the ledger is written; on the 72 held-out tasks the closing harness scored 0.512 against 0.325 for the seed (one rollout each). hy3, thinking on: a first batch of 10 rounds in which four legal accepts summed to a net loss (the ratchet slippage of Appendix G), then a clean restart, 5 rounds, 2 accepts, 0.732 to 0.807, again both lessons.

On the night of submission we re-measured the seed and closing compositions of all three lines under the full protocol (two runs of two rollouts per cell, temperature 0 as in August). Qwen3.6, thinking off: seed 0.396 (delivery 73%, healthy 44%, F1 10%, F2 32%, F3 9%, F4 5%), closing 0.459 (delivery 91%, healthy 48%, F1 14%, F2 4%, F3 24%, F4 10%), accuracy among delivered plans 0.54 against 0.50: +0.063, all of it delivery, with the harness-friction class F2 removed and the freed trajectories landing mostly in F3. Thinking on: seed 0.266 (delivery 47%, healthy 26%, F1 58%, F3 6%, F4 10%), closing 0.320 (delivery 62%, healthy 29%, F1 46%, F3 12%, F4 13%), accuracy 0.56 against 0.52: +0.053, again delivery, with the evidence class F1 shrinking. All shares use every trajectory as the denominator. The per-rollout standard deviation in these four cells is 0.02–0.07 against 0.023 for Qwen3.5, so the thinking-off gain is resolvable and the thinking-on gain is within the noise of its own cells. hy3: seed 0.690 (delivery 84%, healthy 71%, F1 15%, F2 6%, F3 8%), closing 0.755 (delivery 92%, healthy 80%, F1 8%, F2 5%, F3 6%), accuracy 0.818 against 0.824: +0.065, the blank-submission lesson.

In all three lines the gain is delivery, not the accuracy of what is delivered; which process class the harness removes differs by line (F2 for Qwen3.6 off, F1 for Qwen3.6 on and hy3), and in the Qwen3.6 lines the loop class F3 grows as more trajectories run to submission. A 4-rollout re-reading of the Qwen3.6 closing harness four days after the line closed came in at 0.44–0.46 against the single-rollout 0.535; that drift is what led to the same-night rule of Section 4.2.

Three non-Qwen families. The same loop, judge and seed harness were run for five rounds (ten candidates) on MiniCPM5-2B, Spark-X2.5-4B and Ministral-3-8B. The diagnosis step of the rule separates these three from the Qwen lines before any evolution: the healthy share is 0–6% against 15–34% for the Qwen seeds (Table 9), the plans that are delivered score 0.13–0.17 against 0.27–0.31 (Table 7), and on Spark and Ministral F4 is the largest class at the seed (38% and 65%). All three lines closed within noise of their seed anchor, and the composition says why in each case.

MiniCPM5-2B accepted five of ten candidates and gained delivery (61% to 71%), but the plans it was pushed to deliver score lower than the ones it delivered already (mean among delivered 0.173 to 0.156), so the product did not move; for this model the seed harness itself had been the large gain (0.047 in the official loop to 0.106, more than double).

Spark-X2.5-4B accepted six candidates that cut F1 from 21% to 7% without moving delivery (97% to 95%) or the score: the freed trajectories landed in F4 (38% to 60%).

Ministral-3-8B starts with F2 and F3 at 7% together, so the harness has almost no process failure to act on; five of its ten candidates edited the exit gate, eight of the nine rejections were triggered by the proposer's own rollback conditions, and the one accept was a syntax fix.

On all three the loop's candidates went where Section 6.2 predicts, to the enforcement layer. These lines ran with the evidence package as it was before the failure composition was added to it; the compositions in Table 14 are computed after the fact from the anchor trajectories.

Gemma 4 12B. A ten-round line moved the anchor from 0.330 to 0.386, but the anchors were collected across several days on different machines; a same-night re-measurement of seed and evolved harness on the same machine gave +0.019 (dev) and +0.012 (held-out), within noise.

Training on this line's trajectories reproduced the ordering of Table 1 at a magnitude within the resolution line: under the seed harness, LoRA gave +0.034 / +0.018 and full-parameter +0.038 / −0.003 (dev / held-out), which puts the trained weights under the seed harness where the base model sits under the evolved harness, and stacking them on the evolved harness added nothing (LoRA −0.042 / −0.013, full-parameter +0.019 / −0.040).

This model enters with delivery at 98% and F4 the largest class (40% of trajectories), the composition the rule assigns to the weights, so a small harness gain is the branch the rule predicts; the current training recipe, on trajectories from a harness that had barely moved, bought development-task gains that did not carry to held-out, so on this model the weights branch remains open.

For this loop the evidence package was extended to lead with the scorer's aggregated failure composition (the change proposed in Section 6.2), which is why the Gemma line is the first on which the proposer could see F4-relevant evidence; its gains were nonetheless within noise.

WebArena-Lite. Section 5.3 and Table 13 report this line. Two details belong here. Three middleware edits accepted in rounds 1–9 had never executed because of a round-indexing bug in the loop (Appendix G); with the bug fixed, the closing composition re-anchored from 0.396 to 0.375, within noise, so the line's gain rests on the interface edits alone. Under the closing harness the undelivered test trajectories stop mostly on the 40-call budget (77 of 108): the closing harness reads more of each page and walks further, which is the observational nature of its gain.

Table 14: Failure composition of the three smaller non-Qwen lines at their seed and closing anchors (48 development tasks, 4 rollouts; share of trajectories, labels as in Section 3.1; rows sum to 100% up to rounding; computed after the fact from the anchor trajectories; the Ministral seed anchor logged 178 of 192 trajectories). Compare the Qwen seed rows of Table 9: healthy 15–34%, F4 12–28%. On all three the healthy share is at most 6%, and on Spark and Ministral F4 is the largest class before any evolution.

Model	Harness	Score	Delivery	Score among delivered	Healthy	F1 evidence	F2 harness friction	F3 execution control	F4 planning capacity
MiniCPM5-2B	seed	0.106	61%	0.173	5.7%	**42.7%**	2.1%	32.3%	17.2%
MiniCPM5-2B	evolved (5 rounds)	0.110	**71%**	0.156	4.7%	35.9%	2.6%	34.4%	22.4%
Spark-X2.5-4B	seed	0.153	97%	0.158	6.2%	20.8%	2.1%	33.3%	**37.5%**
Spark-X2.5-4B	evolved (5 rounds)	0.155	95%	0.164	5.2%	6.8%	0.0%	28.1%	**59.9%**
Ministral-3-8B	seed	0.112	90%	0.125	0.0%	28.1%	0.0%	7.3%	**64.6%**
Ministral-3-8B	evolved (5 rounds)	0.127	93%	0.136	3.1%	30.7%	0.5%	5.2%	**60.4%**

I Plan-to-JSON Conversion Bias

DeepPlanning scores a plan only after an LLM converts the agent's free-text plan into a structured form. We audited this step because our agents often submit several drafts of a plan in one answer (25.0% of the 9B closing harness's answers, counting the 188 scored answers; 1.0% of the 4B closing harness's). On 20 such multi-draft answers, four converters (the default `qwen-plus`, DeepSeek V4.1 Flash, Kimi K3 and GPT-5.6 Luna) each silently selected a single draft in 20 of 20 cases, hiding the fact that the agent never converged; only GPT-6 Astra, at 64 times the cost, kept all drafts in 11 cases and selected in 3. A manual audit of 19 answers on which Astra and the default converter disagreed found Astra faithful in 18. Unfaithful conversion inflates the composite score by about 0.025 on average (0.09 on the five hardest answers) relative to faithful conversion. Eleven further converters tested on the five hardest answers against 23 manually established checkpoints did no better than 13 of 23, against 7–9 for the default. The root cause is that the official conversion prompt states no rules and its examples demonstrate rewriting; appending a short rule list to the prompt brings GPT-5.6 Terra to 23 of 23 and the default converter to 20 of 23. We reported the issue and the fix upstream. All numbers in this paper use the default converter, whose repeat agreement with itself is 93.5% at the default temperature (96.8% at temperature 0), so every cell is converted the same way; because the bias depends on the multi-draft rate, which differs across configurations (Appendix F), we report that rate and read differences near 0.025 with care.

J Two Frontier API Models Under the Evolved Harness

A co-author ran Kimi K3 and GLM 5.3 as the acting agent on DeepPlanning on a separate machine, four rollouts per cell on the 48 development and 72 held-out tasks, without and with an evolved harness from this paper's lines (Table 15). Kimi K3 gains +0.058 (development) and +0.049 (held-out); Equation 2 on the development means puts 0.010 of it on delivery (96.9% to 99.3%) and 0.048 on the quality among delivered plans (0.431 to 0.479), the shape of the 9B line, and mean steps fall by more than half (19.7 to 8.4 and 23.3 to 10.1), the signature of removed loops. GLM 5.3 reaches 0.512 / 0.488 under the evolved harness; without a bare control its harness gain is not identified, and its 0.036 / 0.037 margin over Kimi is a difference between models, not a gain. Per-run readings, run identifiers, serving settings and the mapping to a versioned harness checkpoint are not in the repository at submission and will be released with the code; these cells are therefore not same-night matched with any other cell, carry no noise estimate of their own, and their failure labels come from an earlier labelling scheme and are not pooled with Section 3.1. They are an aggregate check of harness transfer, not evolution lines, and are not counted among the nine lines of Section 5.4.

Table 15: **Kimi K3 and GLM 5.3 as the acting agent** (four-rollout means on 48 development (D) / 72 held-out (H) tasks, run by a co-author on a separate machine; quality = mean score among delivered development plans). GLM 5.3 has no bare control; no confidence intervals are given because per-run readings are not attached.

Model	Harness	D score	H score	Delivery D / H	Mean steps D / H	Quality D
Kimi K3	Bare	0.418	0.402	96.9% / 95.2%	19.7 / 23.3	0.431
Kimi K3	Evolved	**0.476**	**0.451**	99.3% / 98.7%	8.4 / 10.1	0.479
GLM 5.3	Evolved	0.512	0.488	99.6% / 99.1%	7.6 / 9.2	0.514