Reliability Engineering for Pass^3 Tool Use on CAR-bench Track 1: Faithful Simulation, Live-Schema Guarding, and Gated Admission

Maksim Silchenko
Team Thylinao, CAR-bench Challenge at IJCAI-ECAI 2026, Track 1 (Open)
mthylinao@gmail.com

Abstract

We report the Track 1 system of team Thylinao on CAR-bench [Kirmayr *et al.*, 2026], whose **Pass^3** metric rewards an agent only when all three trials of a task pass every applicable sub-metric. Starting from the provided LiteLLM [BerriAI, 2024] tool-calling agent, three validated changes (greedy decoding, a compliant *live-schema guard*, and an agent-model upgrade to `gemini-3.5-flash` [Google DeepMind, 2026], the dominant lever, with the guard as low-cost insurance) plus one general behaviour skill at high reasoning effort reach $\mathbf{80.2}\%$ Pass^3 on the complete 129-task training split, $+\mathbf{18.9}$ points above the strongest published reference on the identical tasks, and $\mathbf{78.0}\%$ in a single pre-registered read of the held-out public test split, where the references we recompute top out at 58%. Both margins are cross-era, and our calibration probes indicate the era gap favours the references. The headline is performance per unit of model: a flash-class agent clears every frontier reference by more than 18 points; the lead costs latency and dollars, and we report the full trade. Beyond the score, two findings transfer to any simulator-judge Pass^k benchmark: a provider-route pitfall that swings hallucination by about 25 points if uncontrolled, and a gated-admission discipline whose full-split re-measurement caught a *prompt-interference tax*, where added prompt text degrades rules it never touches, invisible to targeted probes.

1 Introduction

CAR-bench [Kirmayr *et al.*, 2026] evaluates whether tool-using LLM agents [Yao *et al.*, 2023] remain reliable under real-world uncertainty: missing capabilities (*hallucination* tasks, where a tool or parameter is withheld), underspecified requests (*disambiguation*), and ordinary requests (*base*). In Track 1 a dockerized agent-under-test answers A2A turns from a held-out evaluator that pairs a user simulator with a policy judge, both locked to Gemini-3.5-flash. The metric, in the simulator-plus-judge lineage of τ-bench [Yao *et al.*, 2024], is Pass^3: a nested AND-gate over three trials and five sub-metrics (action, tool-subset, tool-execution, end-conversation, policy), with the overall score the unweighted mean of the three splits; hallucination tasks score only tool-execution and end-conversation.

Our contributions are methodological as much as they are a score: (1) a faithful-evaluation calibration for Track 1, including a provider-route pitfall that moves hallucination scores by about 25 points and a same-agent simulator-era calibration; (2) a *gated admission chain* (probe, sweep, regression, full split) whose negative results quantify a *prompt-interference tax*: prompt additions degrade rules they never touch, invisibly to targeted probes; (3) a four-class failure taxonomy for Pass^k metrics that separates stabilisable agent variance from judge-verdict noise; and (4) a deployment-verification pipeline (config baked into the image, digest-level checks, CI boot inside the official evaluator) plus a live-schema guard whose measured value roughly doubles as the simulator sharpens and shrinks as the agent strengthens.

2 Approach

Deterministic decoding. The starter agent left temperature unset, so the model sampled and produced large per-trial variance that Pass^3 punishes (any one failing trial fails the task). Forcing `temperature=0` removed agent-side flakiness on the ablation slice (flaky=0 at 8 tasks/split with the 2.5-flash agent); the deployed thinking model retains mild trial-to-trial variation even at temperature 0, which is exactly the class-(ii) residual of our failure taxonomy. A packaging detail: LiteLLM's Gemini-3.x parameter mapper compares `value < 1.0` and raises on environment-variable strings, so any LiteLLM-based Gemini-3.x agent configured through the environment must coerce `AGENT_TEMPERATURE` to a float.

Live-schema guard. On every turn, before dispatch, we validate each tool call against the *per-task* tool schema the evaluator sends: (a) the tool name must exist; (b) every argument key must be a declared property and all required parameters present. On violation we inject a capability-unavailable observation and retry; a repeated violation strips the tool calls so an invalid call is never dispatched. The two guard modules are ported from the public CAReful agent [gmsh, 2026] and adapted to consult only the *live* per-task schema, never a bundled catalog or an out-of-band table of expected parameters. The adaptation and the validation discipline around it

are ours. The guard is compliant by construction: absence is reported as capability-absence, never as removal-detection; it emits official names only; it never reasons about the reward or scorer; and a shared retry budget caps sequential LLM calls per action decision at four, inside the five-call ceiling the benchmark sets for Track 2 and we adopt voluntarily.

Agent-model upgrade. Because the organizers inject the agent key and fund agent inference, the agent model is a free design choice. Upgrading from `gemini-2.5-flash` to `gemini-3.5-flash` removes most hallucination, wrong-action, and ask-vs-act errors outright, including failure modes no schema guard can structurally catch: semantic substitution, future-tense promises, over-action.

General behaviour skill and reasoning effort. The deployed agent prepends a single, split-agnostic instruction set to the host system prompt (behind a flag; byte-identical to the host prompt when off) and runs at high reasoning effort. The skill encodes only *general* benchmark policy that applies to every task type, never task-type- or removed-tool-conditional behaviour: name the intended action and each parameter value before a confirmation-required tool call; use metric units and 24-hour times everywhere, including inside tool arguments; act on exactly what was asked; report an unavailable capability as capability-absence rather than as a removed tool. High reasoning effort stabilises an otherwise-flaky multi-step disambiguation task. Each rule was individually admitted through the gate chain below.

Gated admission for every change. No prompt or code change ships on a targeted probe alone. A candidate rule must clear, in order: a mechanism probe at nine trials on the tasks it is meant to fix; a sweep of the neighbouring tasks its mechanism could plausibly touch; a fixed cross-split regression gate; and a re-measurement of the *entire* training split at three trials that must hold the frozen floor on every split. Run-to-run noise under the frozen configuration is about one task per split at three trials, and the gates use that band consistently: a candidate that cannot be shown safe within it is treated as unsafe. The chain is deliberately expensive because the failure it screens for is invisible to probes: added prompt text taxes the firing probability of rules it never touches (see Negative results). A candidate that did not reproduce its intended fix under the deployed model was dropped rather than shipped.

Deployment as a reliability surface. The same discipline applies to the deployed artifact itself. A multi-agent audit of the frozen submission package (independent automated reviews of the image config blob, the scenario rendering, and the vendored source) caught a defect no score-side validation can see: the validated behaviour skill would have shipped *disabled*, because the scenario config omitted one environment flag whose code default is off. We therefore bake the whole validated non-secret configuration (model id, temperature, thinking, reasoning effort, guard and skill flags) into the image as `ENV` and `CMD`, re-asserted through scenario `command_args` (only the secret API key is injected at evaluation time), so the container is self-contained however the evaluator consumes the scenario file: a second audit found a compose pipeline that overrides the image `CMD`, which the env-plus-args belts survive. Every published digest is verified end-to-end: the config blob checked field-by-field against the validated configuration, the vendored source byte-identical to the gate-validated revision, and the exact pinned digest boots inside the *official evaluator container* in CI, completing a graded task at reward 1.0 with all five sub-metrics passing. The harness, guard, skill, and gate scripts are in the public submission repository.[1]

3 Experimental Setup

Our harness reproduces the official Pass^k computation exactly (a deterministic recomputation check passes on every run). Faithful conditions place the user simulator and policy judge on the *AI-Studio* `gemini/` route, matching the official evaluator's `GEMINI_API_KEY` keying; development and full-split measurements ran the *agent* on the Vertex route for billing reasons, with agent route-parity verified afterwards (20 training tasks re-run with the agent on the shipped `gemini/` route reproduced 60/60 trials), and the shipped image pins the `gemini/` route. Ablations (Table 1) use 8 tasks per split for lever isolation; the deployed configuration is additionally validated on the *entire* training split and, once, on the held-out public test split. Agent inference is organizer-funded for the hidden run. A full-train measurement at three trials is a 12 to 16 hour run on consumer hardware, so the evaluation infrastructure self-heals the failure classes we observed: resumable batches with completion markers, watchdogs for hung LLM calls, and a pause on sustained provider-side outages, detected by failure signature because we saw waves in which the endpoint accepts connections but generation requests hang.

A provider-route calibration pitfall. `vertex_ai/gemini-3.5-flash` (callable only in the Vertex `global` region) is a *different, more lenient* snapshot than AI-Studio `gemini/gemini-3.5-flash`. Holding the 2.5-flash agent fixed and flipping only the simulator route moved hallucination Pass^1 on an eight-task single-trial probe from 37.5% (3/8, AI-Studio) to 62.5% (5/8, Vertex): a 25-point swing with the agent unchanged on both sides. A nominally identical model id on another provider route is not a faithful proxy.

4 Results

Table 1 reports overall and per-split scores under the faithful Gemini-3.5-flash simulator and judge.

Schema guard. Under the faithful simulator the guard adds $+8.3$ points (Pass^1, $50.0 \rightarrow 58.3$), roughly twice its $+4.2$ under the more lenient 2.5-flash simulator, via two causal fixes (a removed tool and a removed parameter) and no regression elsewhere.

Agent upgrade. The agent-model change is the dominant lever in our ablations. At the official Pass^3 the `gemini-3.5-flash` + guard configuration scores $\mathbf{79.2}\%$, versus $50.0/58.3$ (Pass^1) for the 2.5-flash baseline and the strongest official leaderboard reference, `claude-opus-4-6` at 57.8% on the full 254-task public

[1] https://github.com/thylinao1/car-bench-track1-submission

Configuration	Base	Hall.	Disamb.	Overall
2.5-flash, no guard (Pass[1])	87.5	37.5	25.0	50.0
2.5-flash + guard (Pass[1])	87.5	62.5	25.0	58.3
3.5-flash, no guard (Pass[1])	100	100	87.5	95.8
3.5-flash + guard (Pass[3])	**87.5**	**75.0**	**75.0**	**79.2**

Table 1: Faithful-simulator scores (train, 8 tasks/split). Pass[1] rows are single-trial lever-isolation probes at temperature 0; Pass[3] is the official metric. The deployed image additionally enables the behaviour skill and high reasoning effort (see text).

set (0.58 on the official leaderboard; 61.3% on our 129-task train slice, Table 2). A single-trial run of the same agent model read 95.8% (Table 1, row 3); we report the 3-trial figure as the honest number, most of the gap being flakiness the 3-trial metric correctly penalises. Under the deployed model the guard still fires, but rarely: on the held-out public test read it corrected 14 parameter-schema violations across 375 episodes, every one resolved by the in-turn retry, with the strip fallback never needed. Its value roughly doubles as the simulator sharpens and shrinks as the agent strengthens, so on the deployed model it is low-cost insurance against residual invalid calls. All published references ran under the earlier 2.5-flash simulator era, so margins against them are cross-era. That simulator scores the same agent about 25 points *higher* on hallucination than the faithful AI-Studio 3.5 simulator does, exactly as the lenient Vertex route did, so era mismatch inflates the references, not us.

Failure taxonomy from full-train forensics. We read every failing full-train trajectory against the grader output and the benchmark source, and each failure lands in one of four classes with different remedies. (i) *Capability walls*: tasks that fail the action or tool-subset gate under both a stronger agent model and a higher reasoning budget; a genuine limitation, not a tuning target. (ii) *Agent-side sampling variance* [Wang *et al.*, 2023]: e.g. a transitive policy cascade (window defrost forces the air conditioning on, which requires closing any wide-open window in the same action batch) completed in two of three trials and omitted in the third. This class has a clean behavioural pass/fail contrast and is the legitimate target for further skill rules. (iii) *Judge-sampling variance at genuine policy conflicts*: on one route-planning task a byte-identical judged turn passes one trial and fails another, because one policy line says ask before starting navigation and another says proactively select the fastest route; the judge samples between the readings. This class bounds achievable Pass[3] from above and cannot be tuned against without gaming the judge, which we do not do. (iv) *Ungraded trials*: episodes the evaluator never scored, because the user simulator crashed before the agent acted or because the conversation hit the benchmark's 40-turn cap. The metric counts them as failures and so do we; only the crash class is infrastructure rather than agent behaviour. Distinguishing (ii) from (iii), behaviour you can stabilise versus verdict noise you must absorb, is in our experience the single most useful discipline for working against a Pass[k] metric.

Full-train validation. Small 8-task draws are high-variance, so we measured the deployed configuration on the *entire* training set, all 129 tasks across the three splits, at three trials under the faithful simulator. Base scores Pass[3]=$\mathbf{86.0}\%$ (43/50), hallucination $\mathbf{77.1}\%$ (37/48), and disambiguation $\mathbf{77.4}\%$ (24/31), for a measured overall of $\mathbf{80.2}\%$, scored exactly as the official scorer does, which counts an ungraded trial as a failure. Two trials here were never graded: one hallucination trial in which the user simulator crashed before the agent acted, and one disambiguation trial that reached the 40-turn cap on a task that fails its graded trials anyway. Re-measuring the crashed trial cleanly (it passes 3/3) would read 80.9% overall; we report the stricter figure. The larger sample surfaces a hard capability ceiling: ten tasks fail the action or tool-subset gate on every trial under both a stronger model and higher reasoning budget, six of them multi-step routing or tool composition and four climate or sunroof sequences. The remaining addressable gap is variance reduction on flaky tasks, not peak capability.

Negative results: two levers the gates caught. Two plausible skill-rule additions passed their mechanism probes and died only at the final gate. The first, a policy-cascade rule, took both of its target tasks from 2/3 to 9/9 at nine trials and held 27/27 on the neighbouring-task sweep, then failed the full split: the base target task flipped fail-to-pass ($+1$) while five unrelated, previously-passing tasks each lost one trial (-5), netting 39/50=78.0% against the 43/50=86.0% floor; four of the five losers are tasks the new rule cannot even fire on. The drop is not individually significant (McNemar $p \approx 0.22$), but a single-shot submission treats cannot-be-shown-safe as unsafe. The second, a minimal one-bullet variant of the same rule, held base and disambiguation flat but moved hallucination from 77.1% to 75.0%, reading 79.5% overall against the 80.2% floor. Both were reverted. That is the *prompt-interference tax*, and targeted probes are structurally blind to it.

Held-out public test split. Measured in a single pre-registered pass on the held-out public test split (125 tasks: 50 base, 50 hallucination, 25 disambiguation) under the frozen submission configuration (agent on the Vertex route; shipped-route parity verified 60/60), the deployed agent scores Pass[3]=$\mathbf{88.0}\%$ on base (44/50), $\mathbf{70.0}\%$ on hallucination (35/50), and $\mathbf{76.0}\%$ on disambiguation (19/25), for an overall of $\mathbf{78.0}\%$ (unweighted mean). This split was withheld from every tuning, rule-admission, and model-selection decision, and its failures were never mined for levers. The -2.2-point transfer is consistent with the reference-model norm on the same split pair (mean -1.3, range -7.3 to $+8.3$, over the four references with three-trial data on both slices), i.e. with split-mix variance rather than overfitting; the failure texture matches the train classes (navigation walls plus a few flaky trials), with no new failure mode. Five trials here were never graded, every one a conversation that reached the 40-turn cap; counting them as failures, as the metric does, costs two hallucination tasks that pass their other two trials. On the identical unseen tasks the references we recompute top out at 58.0% (`gpt-5.2` 58.0, `sonnet-4-6` 56.7, `opus-4-6` 54.0), so our 78.0 leads the best on-slice reference by $+\mathbf{20.0}$ points

Agent (tasks)	Pass^3	Lat. (s)	Cost ($)
Ours: deployed config (129)	**80.2**	67.6	0.169
claude-opus-4-6 (129)	61.3	28.0	0.162
gpt-5.2 (129)	49.7	76.5	0.064
claude-sonnet-4-6 (254)	56.0	27.0	0.104
gpt-5 (254)	51.4	330.3	0.112
claude-sonnet-4 (254)	45.4	30.1	0.245
gemini-2.5-flash, medium (254)	39.4	16.4	0.014
Qwen3-32B (254)	30.1	336.4	0.001

Table 2: Score versus mean per-episode latency and cost, recomputed from the organizer's raw files with the official analyzer's semantics: Pass^3 over every task with three trials, reference cost and latency over first trials, latency excluding episodes with a logged error (which is why `gpt-5.2`'s latency covers 128 of its 129). Ours averages all three trials and keeps ungraded episodes, so it reads high. Ours, opus-4-6 and gpt-5.2: 129-task train slice; their test-half latency/cost telemetry is zeroed (rewards intact); other rows: all 254 public tasks. Reference scores are 2.5-flash-simulator era; ours the locked 3.5 (see text). Our latency and cost are Vertex-logged; on the parity set the shipped `gemini/` route ran about $1.5\times$ the latency, $1.6\times$ the tokens, and $1.8\times$ the cost, so the deployed figures are correspondingly higher.

(cross-era; see the calibration above). `opus-4-6`, strongest overall, degrades more on this split than `gpt-5.2`, so its wider $+24.0$ gap here is not a real gain over the $+18.9$ train margin. A provider-capacity wave slowed part of this measurement (mean trial latency 974 s inside the window versus 99 s outside); Pass^3 grades the final trajectory, not wall-clock time, and failures do not cluster in the wave (fail rate 10.3% inside versus 16.4% outside); we exclude the window from latency claims.

Token efficiency, latency, and cost. Table 2 places the deployed agent on the score-cost-latency frontier against the organizer's published references, recomputed from their raw per-episode files under the official analyzer's semantics (era caveat in the caption). The award-relevant reading is performance per unit of model: no reference reaches our Pass^3 at any cost or latency, and a flash-class agent outscores every frontier reference by more than 18 points. On the efficiency axis that Pass^3 rewards, consistency per token, our spend is not in reasoning: on the training read thinking is only 2.4 to 11.2% of an episode, so most tokens are fixed prompt and context. That profile is unusually cache-friendly (prompt tokens are 85.8% of the total), so prompt caching is the largest cost lever here; we quantified it as headroom but left it off, blocked per-trial by an unverifiable organizer key tier, a deployment constraint, not a method limit. The dollar and wall-clock cost we pay by choice: reasoning effort is a dial set to the consistency end. At that setting the shipped agent carries the highest per-episode cost in the table once the route is accounted for (Vertex-logged \$0.169, behind `claude-sonnet-4`'s \$0.245, but the shipped `gemini/` route ran $1.6\times$ the tokens and $1.8\times$ the cost on the parity set, near \$0.30) and is among the slowest (67.6 s Vertex-logged, about $1.5\times$ that shipped, from a lighter 20-task parity subset of 47.9 versus 31.2 s, so roughly 100 s per episode, 2.4 to $3.6\times$ `opus-4-6`), faster only than the reasoning-heavy tail (`gpt-5`: 330 s mean, 8,400 s max). The trade is deliberate: a lower effort setting would trade Pass^3 back for speed. A second quantified lever, stripping the benchmark's penalty-free planning-scratchpad tool (31 to 44% of tool calls, a projected 30 to 44% latency cut on disambiguation), stayed unshipped because it could not bound its Pass^3 risk in the noise band without another full-split run: a validated floor beats an unvalidated improvement on a secondary axis.

5 Discussion and Limitations

Ablations are 8 tasks/split; headline figures are the full training split, checked once against the held-out public test; the hidden set may still differ in mix. The behaviour skill and high reasoning effort were validated jointly, not by isolated full-split A/Bs, so their individual contributions are not separated. Even at temperature 0 the simulator and judge drift by about one task per split, so headline results are confirmed at three trials. Once a strong agent is available the model choice dominates and the marginal wins move to variance reduction, where the admission gates earn their cost.

6 Conclusion

From a flash-class model the deployed configuration reaches Pass^3=80.2% on the full training split and 78.0% on the held-out public test split, $+18.9$ and $+20.0$ points over the strongest reference on each. Its lasting contribution is the transferable methodology, not the funded model swap that set the score.

References

[BerriAI, 2024] BerriAI. LiteLLM: Call 100+ llm apis in the openai format, 2024. https://github.com/BerriAI/litellm.

[gmsh, 2026] gmsh. CAReful: an exploration agent for CAR-bench, 2026. https://github.com/gmsh/car-bench-exp-agent.

[Google DeepMind, 2026] Google DeepMind. Gemini 3.5 flash model card, gemini API, 2026. https://ai.google.dev/gemini-api/docs/models.

[Kirmayr *et al.*, 2026] Johannes Kirmayr, Lukas Stappen, and Elisabeth André. CAR-bench: Evaluating the consistency and limit-awareness of LLM agents under real-world uncertainty. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens, editors, *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 40599–40618, San Diego, California, United States, July 2026. Association for Computational Linguistics.

[Wang *et al.*, 2023] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023.

[Yao *et al.*, 2023] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language

models. In *International Conference on Learning Representations*, 2023.

[Yao *et al.*, 2024] Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains, 2024. arXiv preprint arXiv:2406.12045.