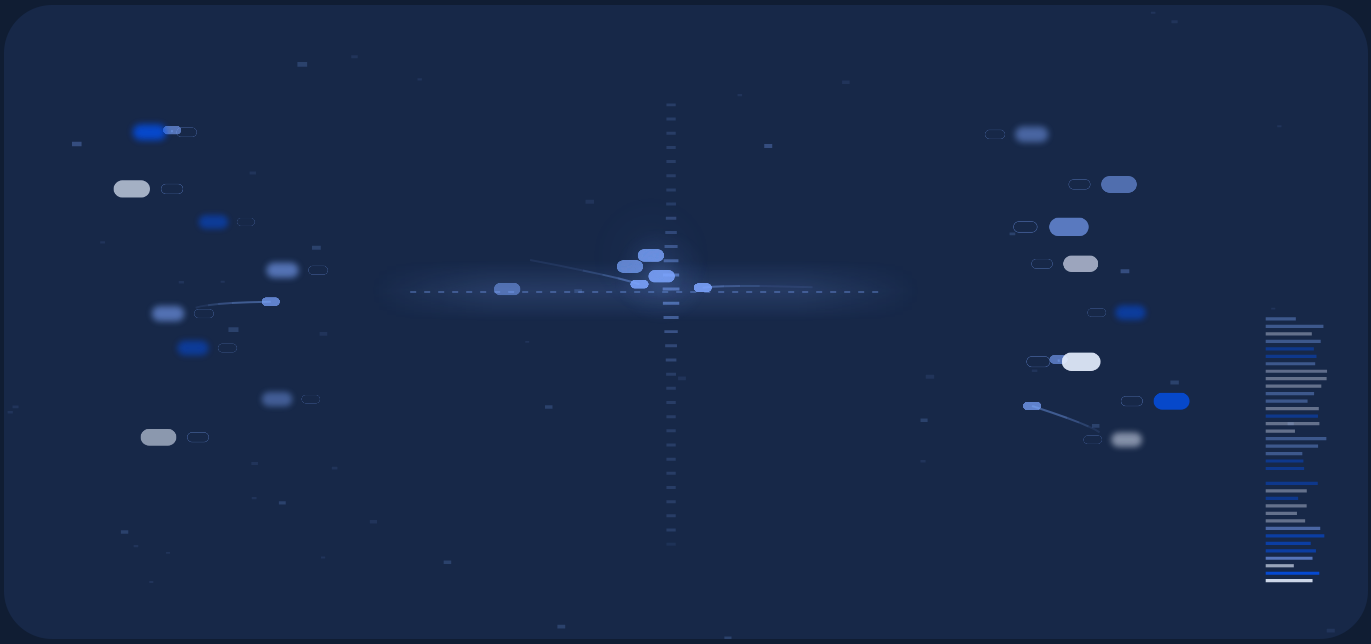
Rillor

EXPLAINER

Agent evaluation metrics and what each one tells you

How task success, pass@k and pass^k, tool-call accuracy, escalation, cost per success and time horizon are computed, from the sources that define them, and what each misses.



Rillor

Contents

01 Summary

02 Background

03 Method

04 Findings

05 Implications

06 Limits of this analysis

07 Sources

Summary

- An agent evaluation runs trials of tasks and scores them with graders that read the transcript, the final state of the environment or both [37]. Each metric built on those scores answers one question: whether a trial succeeded, whether success holds across trials, whether the tool calls were right, whether the agent stopped or handed off when it should, what a success costs and what length of task the agent completes.
- pass@k and pass^k move apart as k grows [37]. At 75% success per trial, pass^3 is about 42% and pass@3 about 98% for independent trials (computed for this report). The top four-trial banking entry on the tau-bench leaderboard, evaluated on 3 August 2026, fell from 55.2% on pass^1 to 35.1% on pass^4 [29, 30].
- Tool-call accuracy has several definitions: exact call matching and per-turn state checks [23, 24], comparison with a reference trajectory [20], and required milestones with forbidden minefields [12].
- Refusal and escalation are scored when the correct action is fixed in advance: transfer rules, missing-parameter cases, paired safe and unsafe prompts and named control states [3, 13, 15, 24].
- Cost-of-pass divides cost per attempt by success rate [9], and agents with close accuracy can differ ninefold in cost [6]. METR's 50% time horizon reached about 12 hours for one model released in February 2026, with wide bounds, on a suite of mostly software tasks [33, 34].
- Outcome-only scores miss part of what matters. In a 2025 METR study, an agent's average success rate on the maintainers' tests was 38% across 18 real repository issues, and none of the 15 pull requests reviewed by hand was mergeable as-is [36].

Background

What an agent evaluation measures

A January 2026 guide to agent evaluation from one model provider sets out the vocabulary used here [37]. A task is "a single test with defined inputs and success criteria", and each attempt at it is a trial. A grader is logic that scores some aspect of the agent's performance. The transcript is the complete record of a trial, including tool calls and intermediate results, and the outcome is the final state of the environment at the end of the trial [37]. Evaluating an agent means evaluating the harness and the model together [37]. Task success reads the outcome. pass@k and pass^k read outcomes across trials. Tool-call measures read the transcript. Cost reads the resources a trial used, and time horizon reads success rates across tasks of different lengths.

Why accuracy alone falls short

In 2024, the authors of "AI Agents That Matter" found that agent benchmarks focused narrowly on accuracy, and that for substantially similar accuracy, cost could differ by almost two orders of magnitude [8]. A 2026 reliability study adds that a single success metric hides whether agents behave consistently across runs, withstand perturbations, fail predictably or keep errors bounded [7]. NIST's generative AI profile asks that performance be measured for conditions similar to deployment and that evaluators avoid extrapolating from narrow, non-systematic and anecdotal assessments [1]. NIST's Center for AI Standards and Innovation released draft practices for automated benchmark evaluations of language models and agent systems in January 2026 [2].

This report is for ML engineers, product owners and QA leads who build or buy agents and need to know how each metric is computed and what it misses.

Method

What was examined. Task success, pass@k, pass^k, other consistency measures, tool-call accuracy, trajectory measures, refusal and escalation, cost and latency, time horizon, and the graders behind them. For each: the defining source, the formula or rule, what the metric needs and what it does not see.

Sources. Papers that define or test the metrics [3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18], NIST's profile and draft [1, 2], framework and service documentation [19, 20], leaderboards and their data [21, 22, 26, 27, 28, 29, 30, 31, 32, 33, 34, 38], benchmark authors' posts [23, 24, 25, 35, 36] and the provider guide [37].

How. Definitions were read in each source's full text and are given in plain notation. Leaderboard figures come from the published data files, each with its date. Numbers computed for this report are labelled. Models are described by kind and release date. Web pages were viewed on 24 October 2026, or on 7 November 2026 where the source entry says so.

Notation. For one task, n is the number of trials, c the number that succeeded and p the success rate. C(a, b) is the number of ways to choose b items from a. pass@k, pass^k and outcome consistency are computed per task and averaged over tasks.

Findings

1. Task success is a final-state check, and graders can be wrong

tau-bench, released in June 2024, gives a customer-service episode a reward of 1 only if the final database is identical to the ground-truth database and the agent's replies contain all necessary information. The average reward across tasks is its main metric, written pass^1 [3]. Its successor, tau2-bench, rewards the overall state, such as a user's issue being fixed as shown in the user's database [5]. CORE-Bench counts a task as solved only if every question about a code repository is answered correctly. In the paper, first posted in September 2024, its best baseline agent solved 21.48% of the hard tasks [11]. The provider guide states the principle with an example: an agent may say a flight is booked, and the outcome is whether a reservation exists in the environment's database [37]. It also recommends partial credit for tasks with several parts [37].

Graders can err in both directions. tau-bench's authors note that a reward of 1 can be necessary without being sufficient, as when an agent issues a return without explicit user confirmation [3]. The Agentic Benchmark Checklist found in 2025 that a trivial agent returning empty responses reached 38% on tau-bench's airline tasks, because impossible tasks counted as successes. Seven of the ten benchmarks it assessed had flaws in outcome validity [14]. In the provider guide's January 2026 account, one frontier model's CORE-Bench score rose from 42% to 95% after fixes for rigid grading, ambiguous task specifications and stochastic tasks that could not be reproduced exactly, together with a less constrained scaffold [37]. HAL describes the same change as an updated scaffold plus fixes to a few grading errors, and declares the benchmark solved [26]. The tau-bench leaderboard records a February 2026 audit that fixed more than 50 airline and retail tasks [28].

A final-state check does not show how the state was reached, how often the agent reaches it or at what cost.

2. pass@k and pass^k answer opposite questions

Definitions. Under pass@k, the 2021 HumanEval paper counts a problem as solved if any of k samples passes the unit tests. It estimates pass@k from n ≥ k samples with c correct as 1 - C(n-c, k) / C(n, k), and shows that the shortcut 1 - (1 - p)^k is biased [4]. tau-bench proposes pass^k for tasks that need consistency, such as customer service: the chance that all k independent trials succeed, estimated as C(c, k) / C(n, k) [3]. At k = 1 the two are identical. As k grows, pass@k approaches 100% and pass^k approaches 0% [37]. The provider guide recommends pass@k where one success matters and pass^k where consistency is essential [37].

k	pass@k = 1 - 0.25^k	pass^k = 0.75^k
1	75.0%	75.0%

k	pass@k = 1 - 0.25^k	pass^k = 0.75^k
2	93.8%	56.3%
3	98.4%	42.2%
4	99.6%	31.6%
8	100.0% (rounded)	10.0%

Table 1. One task at 75% success per trial, independent trials. Computed for this report from the definitions in [3, 37]. The guide gives pass^3 as about 42% [37].

Finite samples. For a task with 6 successes in 8 trials, pass^3 = C(6, 3) / C(8, 3) = 20/56 = 0.357, against the plug-in $0.75^3 = 0.42$. pass@3 is 1, since every set of three trials includes a success (computed for this report from [3, 4]).

Published values. In June 2024, a function-calling agent on a widely used hosted model reached about 61% pass^1 on tau-bench retail, with pass^8 below 25% [3]. In June 2025, tau2-bench reported that for one model whose pass^1 was similar in its new telecom domain and in airline, pass^k fell faster in telecom as k grew [5].

Domain	Evaluated	pass^1	pass^2	pass^3	pass^4	Source
Banking knowledge, top four-trial entry	3 Aug 2026	55.2%	45.2%	39.7%	35.1%	[30]
Banking knowledge, another model	24 Jul 2026	47.9%	39.5%	35.3%	32.0%	[31]
Retail, one model	27 Feb 2026	84.4%	74.4%	66.5%	59.7%	[32]
Telecom, the same model	27 Feb 2026	97.8%	95.8%	93.9%	92.1%	[32]

Table 2. tau-bench leaderboard submission files, four trials per task, rounded to one decimal. The top entry has the highest banking pass^1 of the four-trial submissions listed in the leaderboard's manifest, checked for this report against each listed file [29]. A submission the leaderboard labels custom, its term for results that change the benchmark-side setup or use benchmark-specific training or data, reports 73.2% banking pass^1 from one trial per task, evaluated on 3 November 2026, with no pass^2 to pass^4 [28, 38].

The same model lost about 25 points from pass^1 to pass^4 in retail and under 6 in telecom (computed for this report from [32]). That submission notes that its banking results were re-graded under tau2-bench v1.0.1 on 15 July 2026, and that earlier scores are not comparable [32].

In practice. Inspect, the UK AI Security Institute's evaluation framework, implements `pass_at_{k}` and `pass_k_{k}` reducers (for example `pass_at_5`) over repeated runs, which it calls epochs [19]. NIST's draft notes that more trials per item raise cost and reduce uncertainty [2]. Both estimators assume independent trials, and the provider guide asks that each trial start from a clean environment [37].

3. Consistency also covers rephrasings and model versions

A 2026 study, listed on arXiv as accepted at ICML 2026, proposes twelve reliability metrics in four dimensions: consistency, robustness, predictability and safety [7]. Table 3 lists those that bear on consistency.

Metric	What it measures	Formula or rule
Outcome consistency	Same success or failure on repeated attempts	$(2p - 1)^2$ over K runs per task
Trajectory consistency	Similar paths across runs	Divergence of action frequencies and distance between action orderings
Resource consistency	Stable cost, time and API calls	exp(-mean coefficient of variation)
Prompt robustness	Same accuracy on reworded instructions	min(accuracy on paraphrases / baseline, 1)
Fault robustness	Same accuracy under injected failures	min(accuracy with faults / baseline, 1)

Table 3. Source: [7].

Outcome consistency is 1 when an agent always succeeds or always fails on a task and 0 at a 50/50 split, the corrected formula published on the HAL reliability dashboard [27]. An agent at 75% success has pass@1 of 0.75 and outcome consistency of 0.25 (computed for this report from [7]).

The study ran 15 models on GAIA and on a 26-task verified subset of tau-bench airline, five runs per task at temperature zero, with five paraphrases per instruction [7]. Outcome consistency was low for every model. Agents repeated similar kinds of actions more reliably than the same order of actions. Sensitivity to paraphrasing varied substantially across models, smaller models often matched larger ones on consistency, and recent capability gains brought only small improvements in reliability [7].

Versions change results too. A 2023 study found that one hosted model's accuracy at telling prime from composite numbers went from 84% in its March version to 51% in its June version [16]. The HAL dashboard recommends repeated runs, perturbed inputs and re-evaluation at regular intervals to catch silent degradation [27]. On error bars, a 2024 paper recommends a standard error with every score, clustered when questions come in related groups, and paired comparisons between models. In its example, clustered standard errors were over three times the naive ones [17]. Inspect offers a Wilson interval for binary scores [19].

4. Tool-call accuracy has several definitions

Exact call matching. The first version of the Berkeley Function Calling Leaderboard (BFCL) introduced AST evaluation. The function name must match, every required parameter must be present with an exact value, only documented parameters may appear and types are checked strictly. Any mismatch fails the answer [23]. Relevance detection offers only irrelevant functions and expects no call at all [23].

State and response checks. BFCL V3, from September 2024, checks multi-turn tasks after every turn against the ground-truth backend state and against the minimal viable execution path. An entry passes only if both checks hold in all turns [24]. The state check allows several valid routes, and the response check catches calls, such as a zip-code lookup, that leave no mark on the state [24].

The overall score. BFCL V4 weights agentic tasks 40%, multi-turn 30%, and live, non-live and hallucination measurement 10% each [25]. The leaderboard page, by contrast, calls overall accuracy an unweighted average of sub-categories [21]. A check for this report found that the V4 weights reproduce the overall scores in the results table, last updated 12 April 2026, to within 0.01 points for the top six entries, and that an unweighted mean does not [22].

Trajectory metrics. One cloud provider's agent evaluation documentation, a pre-release page last updated on 7 November 2026, separates final response evaluation from trajectory evaluation [20]. It defines exact match (the same calls in the same order), in-order match (reference calls in order, extras allowed) and any-order match against a reference trajectory. It adds precision, the share of predicted actions found in the reference, and recall, the share of reference actions found in the prediction [20].

Milestones and minefields. ToolSandbox, a 2024 benchmark with 1,032 test scenarios, scores milestones: the critical steps a goal needs, ordered by temporal dependency and matched with a similarity from 0 to 1. A minefield is an event that must not occur, and hitting one sets the trajectory's score to zero. Turn count tracks efficiency separately [12].

Error types. In tau-bench's 2024 analysis of 36 retail failures, wrong arguments or information caused about 55%, wrong decisions about which call to make 25% and partial resolution of compound requests 19% [3].

Method	Unit scored	Passes when	Source
AST match	One call	Name, required parameters and types match exactly	[23]
State check	Each turn	Backend state equals the ground truth	[24]
Response check	Each turn	Execution path matches the minimal ground-truth path	[24]
Trajectory match	Whole run	Calls match a reference under an order rule	[20]
Precision and recall	Whole run	Share of actions in common with a reference	[20]
Milestones and minefields	Whole run	Required steps present, forbidden events absent	[12]

Table 4. Tool-call and trajectory scoring methods, as each source defines them.

5. Outcome vs trajectory evaluation

The provider guide found tests of fixed tool-call sequences too rigid, since agents regularly find valid approaches that designers did not anticipate, and it advises grading what the agent produced over the path it took [37]. BFCL's authors note that pure path matching marks error recovery and redundant actions as wrong [24].

Path data still shows what an outcome hides. HAL found that on SciCode and CORE-Bench, agents almost never finished a run without a single tool-calling failure, even when they succeeded, and that agents that fixed a tool-call or instruction-following error mid-run were 1.5 to 4 times more likely to succeed [6]. AgentAtlas notes that an agent can reach a correct state through an unsafe path [13]. Milestones and action-level precision and recall check required steps without fixing every step [12, 20].

6. Refusal and escalation are scored when the correct action is known

Each scored case states the right action in advance.

- **Transfer rules.** tau-bench's policies tell the agent to transfer the user to a human agent if and only if the request cannot be handled within the scope of its actions, through a `transfer_to_human_agents` tool [3].
- **Missing information.** BFCL V3 has 200 missing-parameter cases, where the model should request clarification, and 200 cases where no available function fits [24]. ToolSandbox uses minefields to test that an agent recognizes an impossible task instead of hallucinating [12].

- **Both directions.** XSTest pairs 250 safe prompts with 200 unsafe ones and classifies responses as full compliance, full refusal or partial refusal. One open model tested in 2023 fully refused 38% of the 250 safe prompts and partially refused another 21.6% [15]. The provider guide asks for tests of the cases where a behavior should occur and those where it should not [37].
- **Control decisions.** AgentAtlas, posted in May 2026, defines six control states: act, ask, refuse, stop, confirm and recover. Its failure risks include over-acting, unnecessary asking, over-refusal and missing irreversibility [13].

Refusals also touch other scores. NIST's draft asks evaluators to check transcripts for refusals that may affect results [2]. The reliability study scores compliance and harm severity, and keeps safety out of its aggregate because averaging hides tail risks [7]. The HAL dashboard ranks incorrect charges and refunds first among failure modes on its customer-service tasks [27].

7. Cost and latency per success

"AI Agents That Matter" argues that accuracy alone cannot identify progress, because retrying raises it. One code-generation system went from close to 0% to over 30% with a million retries [8]. For a procurement decision, cost is "the actual construct of interest" [8].

Cost-of-pass, from a paper published at ICLR 2026 according to its PDF, is the expected cost of one attempt divided by the probability of success: the expected cost of one correct solution under independent trials. It is infinite when the success rate is zero [9]. The paper tests language models on quantitative and knowledge tasks, and the formula applies to an agent trial unchanged.

Agent	Cost per attempt	Success rate	Cost per success
A	$0.50	40%	$1.25
B	$2.00	90%	$2.22

Table 5. Hypothetical numbers, computed for this report with the formula in [9]. Agent A costs less per success. Agent B succeeds more than twice as often.

HAL's October 2025 paper ran 21,730 rollouts across 9 models and 9 benchmarks for about $40,000, with prices as of 24 September 2025 [6]. Its findings on cost:

- "In only 1 of 9 benchmarks do we observe the most costly model run on the Pareto frontier" [6].
- In 21 of 36 model, agent and benchmark combinations, more reasoning effort gave equal or lower accuracy [6].

- On one web benchmark, two scaffold and model pairs cost $171 and $1,577, a ninefold difference for two points of accuracy [6].
- The average cost of one evaluation ranged from $13 to over $450, by benchmark [6].

The HAL site puts it in one line: "Agents can be 100x more expensive while only being 1% better" [26].

BFCL reports cost as an estimate for the entire benchmark in US dollars and latency in seconds [21]. In its table of 12 April 2026, the first-ranked entry, a proprietary model, scored 77.47% at $86.55 with a mean latency of 4.38 seconds. An MIT-licensed model ranked fourth scored 72.38% at $4.64 [22]. tau-bench put its 2024 agent cost at $0.38 per task [3]. The reliability study found high run-to-run variance in token and compute use [7]. NIST's draft notes that performance can often be raised by running systems in higher-cost modes, and recommends reporting cost alongside performance when systems are not uniformly cost controlled [2].

8. Time horizon converts success rates into task length

METR's paper, listed on arXiv as published at NeurIPS 2025, defines the 50% time horizon as the time humans typically take to complete tasks that models can complete with 50% success [10]. Each task's length is the geometric mean time of successful human baselines, and a logistic fit of success against the log of task length gives the horizon [10]. METR runs six independent attempts per task [33]. Its FAQ answers no to whether the horizon means the length of time an agent can act autonomously, and describes it as a measure of task difficulty [33].

Model released	50% horizon	Bounds	80% horizon
March 2023	4.0 min	1.9 to 8.0	0.9 min
February 2025	60.4 min	33.0 to 104.2	12.1 min
April 2025	119.7 min	74.6 to 190.9	30.0 min
November 2025	293.0 min	161.7 to 623.7	49.4 min
February 2026	718.8 min	316.7 to 3,633.8	69.9 min

Table 6. One model per row, in minutes of human expert time, from METR's Time Horizon 1.1 data file [34]. The page was last updated on 8 September 2026 [33]. The file does not state the confidence level of its bounds.

For the four models released in 2025 and 2026, the 80% horizon was 4.0 to 10.3 times shorter than the 50% horizon (computed for this report from [34]). The paper reported 4 to 6 times on its original suite [10]. It estimated a doubling time of 207 days [10]. Time Horizon 1.1, released on 29 January 2026, grew the suite from 170 to 228 tasks and changed or removed tasks that were confusing, easy to reward-hack or wrongly scored. At that release the post-2023 doubling time moved from 165 to 131 days [35], against 128.7 days in the data file viewed for this report [34], and only 5 of the 31 tasks of 8 hours or longer had measured human times [35]. The data file includes models released after that post [34].

METR treats measurements above 16 hours as unreliable, its suite is mostly software engineering, machine learning and cybersecurity tasks, and the page is no longer actively updated [33]. The paper calls its projection of a one-month horizon between mid-2028 and mid-2031 a naive extrapolation [10]. At $143.61 an hour for an expert, it found that more than 80% of successful runs cost less than 10% of what a human would cost [10].

9. What outcome-only scores leave out

In an August 2025 study, METR tested an early-2025 agent on 18 real issues from two open-source repositories. The agent's average success rate on the maintainers' tests was 38%, with a 95% confidence interval of plus or minus 19 points, and none of the 15 pull requests reviewed by hand was mergeable as-is. Reviewers estimated an average of 26 minutes of fixes even for those that passed [36]. HAL's log analysis found that web benchmarks score an agent that abstains the same as one that leaks a user's credit card details, and found eight cases where agents located gold answers online [6]. The HAL dashboard documents ground truth reaching GAIA agents through input files and online mirrors [27]. NIST's draft warns of grader gaming, such as removing failing tests instead of fixing the bug [2]. AgentAtlas argues that outcome, control decisions and trajectory need separate scores. In its illustrative run no model was top-ranked on every axis, and its authors say the run supports no stable ranking [13].

10. Graders: code, model and human

Grader	Methods	Strengths	Weaknesses
Code	String match, binary tests, outcome and tool-call verification, transcript analysis	Fast, cheap, objective, reproducible	Brittle to valid variations
Model	Rubrics, assertions, pairwise and reference-based comparison	Flexible, scalable, captures nuance	Non-deterministic, needs calibration with humans
Human	Expert review, spot checks, inter-annotator agreement	Gold-standard quality, calibrates model graders	Expensive, slow

Table 7. Grader types in the provider guide [37].

The guide asks that model graders be calibrated with human experts and given a way out, such as answering "Unknown" when information is missing [37]. A 2023 study found that strong model judges agreed with human preferences on chat answers over 80% of the time, the level humans reach with each other, and documented position, verbosity and self-enhancement biases [18]. The guide adds that scores should not be taken at face value until someone reads the transcripts, and that with frontier models a 0% pass rate across many trials is "most often a signal of a broken task" [37].

Implications

For teams building agents. Report the per-trial success rate with the number of trials behind it, and add pass^k where users expect the same result every time [3, 37]. Use the unbiased estimators when trials are few [3, 4]. Test reworded instructions and each new model version, since both moved results in the studies reviewed [7, 16]. Track tool-call errors by type, because they appear even in runs that succeed [3, 6].

For product owners and buyers. Ask what a success figure counts. In METR's study, a test-pass score and a reviewed-output score differed widely for the same agent [36]. Ask for cost reported with accuracy [2, 6], for the score of a do-nothing agent on the same tasks [14] and for the benchmark and model versions behind every figure [2]. A leaderboard figure is a dated measurement [30, 31, 32].

For QA leads. Write the correct action into every refusal and escalation case, including cases where the agent should proceed, so that over-refusal and under-refusal both show up [15, 37]. Keep the outcome check separate from the path check, and where several paths are valid, limit path checks to required steps and forbidden actions [12, 37].

For researchers and benchmark authors. The checklist asks for confidence intervals, baselines that use no model and trivial-agent results [14]. NIST's draft adds benchmark and model versions, test-item counts and cost controls [2].

Pre-deployment testing. Rillor tests the agents an organization plans to deploy on its own tasks, measured through inputs and outputs on pinned model versions. It scores task success on cases with known answers, refusal and escalation, tool-use errors, and consistency across repeated runs, rephrasings and model versions.

Limits of this analysis

- This is an explainer of published definitions. It does not rank models or agents, and leaderboard figures are dated snapshots that change with new submissions and grading fixes [28, 29].
- No source reviewed compares the tool-call scoring methods on the same agent, so this report sets them side by side without ranking them.
- No source reviewed gives accepted thresholds for pass^k, consistency or escalation rates, or measures escalation in production. The report states none.
- Cost-of-pass was tested on language-model tasks [9]. Table 5 applies the formula to hypothetical agents.
- Time horizons come from a suite of software, machine learning and cybersecurity tasks, on a page that is no longer actively updated [33].
- The model-judge agreement figure comes from 2023 chat answers [18].
- NIST AI 800-2 is cited as an initial public draft [2].
- The pass@k and pass^k estimators assume independent trials [3, 4]. Shared state, caching or rate limits can break that assumption, and no source reviewed measures the effect.
- Several sources are preprints. The reliability study is listed as accepted at ICML 2026 [7], and the HAL site states that its paper was accepted at ICLR 2026 [26].

Sources

1. Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile, NIST AI 600-1. National Institute of Standards and Technology, July 2024. https://nvlpubs.nist.gov/nistpubs/ai/NIST.AI.600-1.pdf

2. Practices for Automated Benchmark Evaluations of Language Models, NIST AI 800-2, Initial Public Draft. National Institute of Standards and Technology, Center for AI Standards and Innovation, January 2026. https://nvlpubs.nist.gov/nistpubs/ai/NIST.AI.800-2.ipd.pdf

3. τ-bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains. Shunyu Yao, Noah Shinn, Pedram Razavi and Karthik Narasimhan, arXiv:2406.12045, 17 June 2024. https://arxiv.org/abs/2406.12045

4. Evaluating Large Language Models Trained on Code. Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan and others, arXiv:2107.03374, version 2, 14 July 2021. https://arxiv.org/abs/2107.03374

5. τ^2-Bench: Evaluating Conversational Agents in a Dual-Control Environment. Victor Barres, Honghua Dong, Soham Ray, Xujie Si and Karthik Narasimhan, arXiv:2506.07982, 9 June 2025. https://arxiv.org/abs/2506.07982

6. Holistic Agent Leaderboard: The Missing Infrastructure for AI Agent Evaluation. Sayash Kapoor, Benedikt Stroebl, Peter Kirgis, Nitya Nadgir, Zachary S. Siegel, Arvind Narayanan and others, arXiv:2510.11977, 13 October 2025. https://arxiv.org/abs/2510.11977

7. Towards a Science of AI Agent Reliability. Stephan Rabanser, Sayash Kapoor, Peter Kirgis, Kangheng Liu, Saiteja Utpala and Arvind Narayanan, arXiv:2602.16666, version 3, 2 June 2026 (first version 18 February 2026). https://arxiv.org/abs/2602.16666

8. AI Agents That Matter. Sayash Kapoor, Benedikt Stroebl, Zachary S. Siegel, Nitya Nadgir and Arvind Narayanan, arXiv:2407.01502, 1 July 2024. https://arxiv.org/abs/2407.01502

9. Cost-of-Pass: An Economic Framework for Evaluating Language Models. Mehmet Hamza Erol, Batu El, Mirac Suzgun, Mert Yuksekgonul and James Zou, arXiv:2504.13359, version 2, 26 February 2026 (first version 17 April 2025). https://arxiv.org/abs/2504.13359

10. Measuring AI Ability to Complete Long Software Tasks. Thomas Kwa, Ben West, Joel Becker, Amy Deng and others, METR, arXiv:2503.14499, version 4, 10 July 2026 (first version 18 March 2025). https://arxiv.org/abs/2503.14499

11. CORE-Bench: Fostering the Credibility of Published Research Through a Computational Reproducibility Agent Benchmark. Zachary S. Siegel, Sayash Kapoor, Nitya Nadgir, Benedikt Stroebl and Arvind Narayanan, arXiv:2409.11363, version 2, 22 June 2026 (first version 17 September 2024). https://arxiv.org/abs/2409.11363

12. ToolSandbox: A Stateful, Conversational, Interactive Evaluation Benchmark for LLM Tool Use Capabilities. Jiarui Lu, Thomas Holleis, Yizhe Zhang, Bernhard Aumayer, Feng Nan and others, arXiv:2408.04682, version 2, 16 April 2025 (first version 8 August 2024). https://arxiv.org/abs/2408.04682

13. AgentAtlas: Beyond Outcome Leaderboards for LLM Agents. Parsa Mazaheri and Kasra Mazaheri, arXiv:2605.20530, version 2, 26 May 2026 (first version 19 May 2026). https://arxiv.org/abs/2605.20530

14. Establishing Best Practices for Building Rigorous Agentic Benchmarks. Yuxuan Zhu, Tengjun Jin, Yada Pruksachatkun, Andy Zhang, Shu Liu, Sasha Cui, Sayash Kapoor, Shayne Longpre and others, arXiv:2507.02825, version 5, 7 August 2025 (first version 3 July 2025). https://arxiv.org/abs/2507.02825

15. XSTest: A Test Suite for Identifying Exaggerated Safety Behaviours in Large Language Models. Paul Röttger, Hannah Rose Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi and Dirk Hovy, NAACL 2024, arXiv:2308.01263, version 3, 1 April 2024. https://arxiv.org/abs/2308.01263

16. How Is ChatGPT's Behavior Changing over Time? Lingjiao Chen, Matei Zaharia and James Zou, arXiv:2307.09009, version 3, 31 October 2023. https://arxiv.org/abs/2307.09009

17. Adding Error Bars to Evals: A Statistical Approach to Language Model Evaluations. Evan Miller, arXiv:2411.00640, 1 November 2024. https://arxiv.org/abs/2411.00640

18. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang and others, NeurIPS 2023 Datasets and Benchmarks Track, arXiv:2306.05685, version 4, 24 December 2023. https://arxiv.org/abs/2306.05685

19. Scoring Metrics, Inspect user guide. UK AI Security Institute, viewed 24 October 2026. https://inspect.aisi.org.uk/metrics.html

20. Evaluate Gen AI agents, Gemini Enterprise Agent Platform documentation. Google Cloud, last updated 7 November 2026, viewed 7 November 2026. https://docs.cloud.google.com/gemini-enterprise-agent-platform/models/evaluation-agents

21. Berkeley Function Calling Leaderboard (BFCL) V4. Gorilla project, UC Berkeley, last updated 12 April 2026, viewed 24 October 2026. https://gorilla.cs.berkeley.edu/leaderboard.html

22. BFCL V4 overall results table (data_overall.csv). Gorilla project, UC Berkeley, last updated 12 April 2026, viewed 24 October 2026. https://gorilla.cs.berkeley.edu/data_overall.csv

23. Berkeley Function-Calling Leaderboard. Fanjia Yan, Huanzhi Mao, Charlie Cheng-Jie Ji, Ion Stoica, Joseph E. Gonzalez, Tianjun Zhang and Shishir G. Patil, Gorilla project, UC Berkeley, last updated 19 August 2024. https://gorilla.cs.berkeley.edu/blogs/8_berkeley_function_calling_leaderboard.html

24. BFCL V3: Multi-Turn and Multi-Step Function Calling Evaluation. Huanzhi Mao and others, Gorilla project, UC Berkeley, 19 September 2024, last updated 10 December 2024. https://gorilla.cs.berkeley.edu/blogs/13_bfcl_v3_multi_turn.html

25. BFCL V4 Agentic, Part 1: Web Search. Huanzhi Mao, Raymond Tsao, Jingzhuo Zhou, Shishir G. Patil and Joseph E. Gonzalez, Gorilla project, UC Berkeley, 17 July 2025, last updated 20 July 2025. https://gorilla.cs.berkeley.edu/blogs/15_bfcl_v4_web_search.html

26. HAL: Holistic Agent Leaderboard. SAgE team, Princeton University, viewed 24 October 2026. https://hal.cs.princeton.edu/

27. AI Agent Reliability Tracker (HAL reliability dashboard). SAgE team, Princeton University, viewed 24 October 2026. https://hal.cs.princeton.edu/reliability

28. τ-bench leaderboard. Sierra, viewed 24 October and 7 November 2026. https://taubench.com/

29. τ-bench leaderboard submission manifest (submissions dated 26 February to 20 October 2026). Sierra, viewed 7 November 2026. https://sierra-tau-bench-public.s3.us-west-2.amazonaws.com/submissions/manifest.json

30. τ-bench leaderboard submission qwen3-8-max_sierra_2026-08-04 (submission.json, banking knowledge, evaluated 3 August 2026). Sierra, 4 August 2026. https://sierra-tau-bench-public.s3.us-west-2.amazonaws.com/submissions/qwen3-8-max_sierra_2026-08-04/submission.json

31. τ-bench leaderboard submission grok-4-5_sierra_2026-08-04 (submission.json, banking knowledge, evaluated 24 July 2026). Sierra, 4 August 2026. https://sierra-tau-bench-public.s3.us-west-2.amazonaws.com/submissions/grok-4-5_sierra_2026-08-04/submission.json

32. τ-bench leaderboard submission qwen3.5-397b-a17b-think_sierra_2026-03-02 (submission.json, all domains, evaluated 27 February 2026). Sierra, 2 March 2026. https://sierra-tau-bench-public.s3.us-west-2.amazonaws.com/submissions/qwen3.5-397b-a17b-think_sierra_2026-03-02/submission.json

33. Task-Completion Time Horizons of Frontier AI Models. METR, last updated 8 September 2026, viewed 24 October 2026. https://metr.org/time-horizons/

34. Time horizon benchmark results, Time Horizon 1.1 data file. METR, viewed 24 October 2026. https://metr.org/assets/benchmark_results_1_1.yaml

35. Time Horizon 1.1. METR, 29 January 2026. https://metr.org/blog/2026-01-29-time-horizon-1-1/

36. Research Update: Algorithmic vs. Holistic Evaluation. David Rein, METR, 13 August 2025. https://metr.org/blog/2025-08-12-research-update-towards-reconciling-slowdown-with-time-horizons/

37. Demystifying evals for AI agents. Mikaela Grace, Jeremy Hadfield, Rodrigo Olivares and Jiri De Jonghe, Anthropic Engineering, 9 January 2026. https://www.anthropic.com/engineering/demystifying-evals-for-ai-agents

38. τ-bench leaderboard submission cortex-0-1_principledmachines_2026-10-20 (submission.json, banking knowledge, evaluated 3 November 2026). Sierra, 4 November 2026, viewed 7 November 2026. https://sierra-tau-bench-public.s3.us-west-2.amazonaws.com/submissions/cortex-0-1_principledmachines_2026-10-20/submission.json

About this report

Published by Rillor on 7 November 2026. Superintelligence (SI) tools assist research and drafting, and every figure cites its source. Rillor builds and secures SI systems, supplies the data they learn from and delivers the infrastructure they run on. We apply SI to financial and compute markets.

Cite as: Rillor. Agent evaluation metrics and what each one tells you. 7 November 2026. https://rillor.com/insights/agent-evaluation-metrics

Notices

Superintelligence (SI) outputs can be wrong. Important decisions should include human review.

Work with Rillor

Send us the tasks your agent handles, the model versions it runs on and the decisions it is trusted with. Contact us at rillor.com/contact?type=evaluation or contact@rillor.com. More on SI evaluation: rillor.com/si/evaluation