

AlphaEval: Evaluating Agents in Production

Pengrui Lu^{*1,3,4} Bingyu Xu^{*2,6} Wenjun Zhang^{*2,5} Shengjia Hua^{*2}
Xuanjian Gao² Ranxiang Ge² Lyumanshan Ye^{3,4} Linxuan Wu² Yiran Li²
Junfei Fish Yu⁹ Yibo Zhang¹¹ Ruixin Li¹¹ Manxiang Li¹¹ Xiao Han¹¹
Xiaocong Zhou¹² Guangyao Chi⁷ Zisheng Chen⁸ Kaishen Chen¹⁰ Kun Wang¹⁰
Qihua Xu¹⁰ Fengyue Meng² Yuchen Ni² Jiajun Li² Jinxiu Liu²
Danfeng Zhang² Jingru Zhao² Pengfei Liu^{+1,3,4}
¹SII ²MiraclePlus ³SJTU ⁴GAIR ⁵HIT ⁶UCAS
⁷LangCore ⁸Jiqizhixin ⁹HunterAI ¹⁰CinoCore ¹¹KuaFuAI ¹²POET
⁺Equal contribution. ^{*}Corresponding author.

Abstract

AI agents are being deployed commercially faster than evaluation methods have adapted to production realities. Existing benchmarks commonly use retrospectively curated tasks with explicit requirements and deterministic metrics, whereas production work contains implicit constraints, heterogeneous documents, fragmented information, undeclared domain knowledge, and expert-defined success criteria. We present *AlphaEval*, a production-grounded benchmark of 94 tasks sourced from **seven** companies across six O*NET occupational domains ([U.S. Department of Labor, 2025](#)). Unlike model-centric benchmarks, *AlphaEval* evaluates complete agent products—Claude Code, Cursor, Codex, and GitHub Copilot—across 14 scaffold-model configurations. Its evaluation combines reference metrics, formal verification, fixed expert-authored rubrics applied by an LLM judge, and automated UI testing. Beyond the benchmark, we contribute a two-layer requirement-to-benchmark framework: a structured, human-led playbook for eliciting and validating authentic requirements with partner experts, followed by a technical pipeline that turns a specified task package into an automated, reproducible evaluation. The best configuration scores 64.41/100, with large variation across domains and scaffolds.

1 Introduction

Deploying an AI agent in production raises a challenge that is often underestimated relative to building it: knowing whether the agent actually works. In our survey of 27 AI-product companies, 63% report low confidence that a model or prompt update improves their product, and 70.4% rely on developers testing as a side task ([Appendix N](#)). Broader practitioner evidence likewise identifies evaluation infrastructure as a central deployment challenge ([Pan et al., 2025](#)). Production-grounded evaluation has nevertheless received comparatively little attention.

Existing benchmarks like SWE-bench ([Jimenez et al., 2024](#)), WebArena ([Zhou et al., 2024](#)), and OSWorld ([Xie et al., 2024](#)) have been instrumental in driving agent capabilities forward. However, they commonly curate or construct tasks in standardized settings with well-specified requirements and deterministic metrics—conditions that diverge from many production evaluations. A recent systematic analysis of 43 benchmarks covering 72,342 tasks quantifies a related coverage mismatch, showing that agent evaluation remains heavily programming-centric relative to the occupational categories where human labor and economic value concentrate ([Wang et al., 2026](#)).

Production evaluation operates under different constraints: requirements arrive as loosely worded business descriptions with implicit constraints; inputs are heterogeneous multi-modal documents with information fragmented across sources; tasks demand undeclared domain expertise; and success criteria are defined by domain experts. Existing benchmarks capture some of these properties individually, but seldom combine three structural axes: (1)

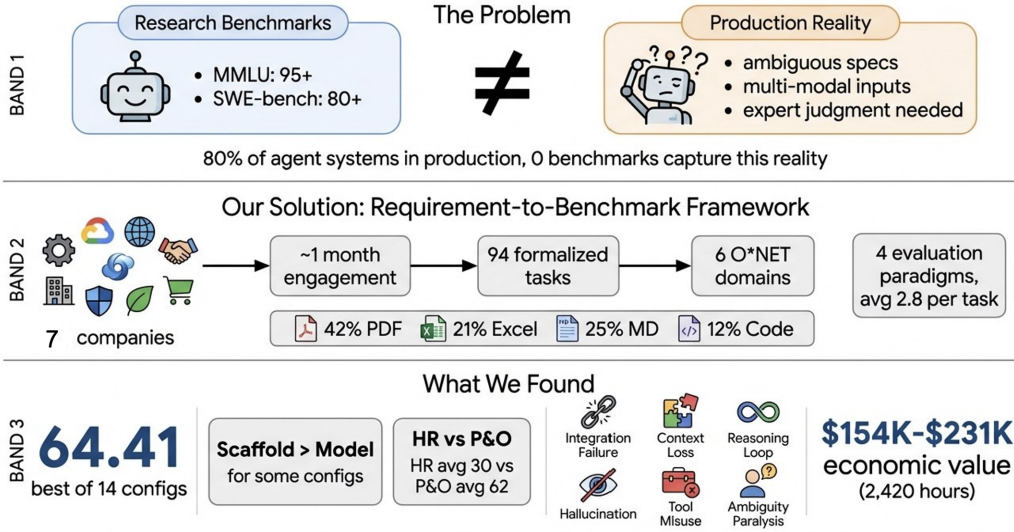


Figure 1: Overview of *AlphaEval*. **Band 1:** The research–production gap. **Band 2:** A human-led elicitation playbook and automated execution pipeline produce 94 tasks across six O*NET domains. **Band 3:** Best configuration 64.41/100, large scaffold effects, and six recurring failure modes.

Task Under-specification; (2) *Expert-grounded, multi-dimensional judgment*; and (3) *Continuous Evolution* as business requirements change.

We address this by partnering with companies deploying AI agents in core workflows. The result is *AlphaEval*¹, a benchmark of 94 production tasks and a reusable construction framework. We evaluate six frontier models through four commercial agent products, yielding 14 model–scaffold configurations. The best configuration (Claude Code + Opus 4.6) achieves 64.41/100; the same model varies by 11 points across scaffolds; and domain averages range from 62.0 to 30.0 (Figure 1). These results characterize task performance, not the replaceability of entire occupations.

Our contributions are fourfold:

- **A requirement-to-benchmark construction framework.** A human-led operational playbook for partner engagement and requirement elicitation, plus a technical execution pipeline that automates evaluation once experts have specified and validated a task package.
- **A production-grounded agent benchmark.** 94 tasks from seven companies classified into six O*NET occupational domains, preserving production complexity—ambiguous specifications, multi-modal inputs, and domain-specific evaluation criteria tied to customer satisfaction.
- **A unified evaluation framework.** Multiple complementary paradigms behind a standardized API and Docker execution, chosen for measurement validity rather than paradigm count.
- **An empirical analysis of the research-production gap.** Frontier agents leave substantial headroom on these production tasks, with the best configuration achieving 64.41 on average. We conduct a systematic error analysis of observed failure cases.

¹Project page: <https://alphaeval.ai>; public evaluation framework, templates, and examples: <https://github.com/GAIR-NLP/AlphaEval>. The name is unrelated to the formula-alpha-mining framework of Ding et al. (2025).

Benchmark	Provenance	Output Fidelity	Eval Target	#Paradigms
MMLU (Hendrycks et al., 2020)	Research	Short Answer	Model	1
SWE-bench (Jimenez et al., 2024)	Curated	Real-world Code	Agent System	1
WebArena (Zhou et al., 2024)	Inspired	Controlled Web	Agent System	1
OSWorld (Xie et al., 2024)	Inspired	Controlled OS	Agent System	1
TheAgentCompany (Xu et al., 2024)	Simulated	Controlled Workplace	Agent System	2
DevAI (Zhuge et al., 2024)	Workflow-inspired	Code Project	Agent System	1
SWE-bench Pro (Deng et al., 2025)	Curated	Real-world Code	Agent System	1
SWE-Lancer (Miserendino et al., 2025)	Curated	Real-world Freelance	Agent System	1
xbench (xbench Team, 2025)	Production	Real-world Business	Agent System	1
<i>AlphaEval</i> (Ours)	Production	Real Deliverables	Agent System	4

Table 1: Comparison with representative agent benchmarks. Paradigm count is descriptive: value comes from using complementary methods that validly measure different success criteria, not from maximizing the count.

2 Background and Related Work

The rapid advancement of LLMs has catalyzed agents capable of complex, multi-step tasks (Wang et al., 2025a; Xi et al., 2023). Evaluation has progressed from isolated capabilities to interactive environments such as SWE-bench, WebArena, OSWorld, and Agent-Bench (Jimenez et al., 2024; Zhou et al., 2024; Xie et al., 2024; Liu et al., 2023). Rigorous task setup and reward design remain essential: auditing has shown that benchmark defects can substantially distort measured performance (Zhu et al., 2025).

Real-world and profession-aligned benchmarks. Recent work moves toward consequential work: SWE-bench Pro and SWE-Lancer use long-horizon or freelance software tasks (Deng et al., 2025; Miserendino et al., 2025); BountyBench grounds security outcomes in dollar impact (Zhang et al., 2025); TheAgentCompany simulates a workplace (Xu et al., 2024); DeliveryBench studies embodied delivery (Mao et al., 2025); and xbench evaluates profession-aligned recruitment and marketing tasks (xbench Team, 2025). Open-world evaluations deliberately trade standardization for a few long, messy real-world cases (Kapoor et al., 2026); they are complementary to our benchmark-scale, reproducible black-box evaluation. Wang et al. (2026) diagnose occupational coverage by mapping 43 benchmarks (72,342 tasks) to 1,016 O*NET occupations, but do not release an executable benchmark. *AlphaEval* operationalizes their coverage, realism, and granularity principles through partner-sourced tasks across six domains.

Scaffolds and evaluation methods. Prior analyses already show that scaffold choice can materially change agent performance (Brand & Denain, 2025; Fan et al., 2025; Meireles et al., 2025); our contribution is to corroborate and extend this effect across four shipped products, six professional domains, and 14 scaffold-model configurations. LLM-as-Judge enables scalable assessment but can exhibit position, leniency, and self-preference biases (Panickssery et al., 2024; Zheng et al., 2023). Agent-as-a-Judge and its DevAI benchmark add trajectory-aware judging of realistic development tasks (Zhuge et al., 2024). *AlphaEval* instead operationalizes partner-authored acceptance criteria with complementary reference, formal, rubric, and execution-based checks; Appendix A compares 90+ benchmarks.

3 From Production Requirements to Executable Benchmarks

Retrospective benchmarks begin with existing artifacts and then design evaluation criteria around them. *AlphaEval* instead begins with tasks that partner companies need agents to perform and uses two explicit layers (Figure 2). The *upstream operational layer* is a structured but human-intensive playbook: selecting partners, eliciting tacit requirements, and negotiating acceptance criteria. It depends on access to domain experts and is not claimed as an automated algorithm. The *downstream technical layer* starts only after a task package is specified; it standardizes packaging, sandboxed execution, evaluation dispatch, logging, and aggregation so repeated evaluation is automated and reproducible.

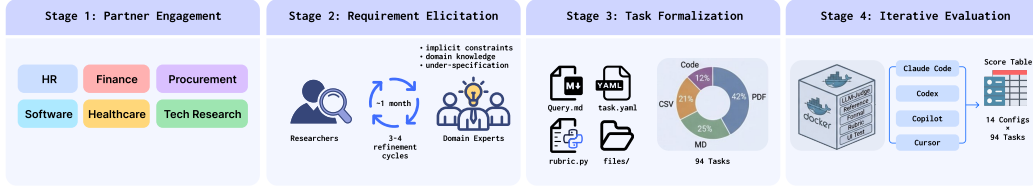


Figure 2: Two-layer construction framework. Partner engagement and requirement elicitation are human-led; after experts specify and validate a task package, the technical pipeline supports reproducible sandboxed execution and evaluation across agent products.

Domain (O*NET)	Representative Task	#	Input	Primary Eval.
Human Resources (13-1071)	Resume screening vs. JD	11	PDF, JPEG	F1 score
Finance & Investment (13-2051)	Segment research & pitch critique	22	PDF, MD, TXT	LLM-as-Judge
Procurement & Operations (13-1020)	BOM cost optimization	23	Excel, CSV, MD	Constraint verif.
Software Engineering (15-1252)	Full-stack app generation	11	YAML, Code	UI testing
Healthcare & Life Sci. (29-9099)	eCRF & insurance policy analysis	16	PDF, MD	LLM + Numerical
Technology Research (15-1221)	AI industry deep analysis	11	MD	LLM-as-Judge
Total		94		

Table 2: Overview of *AlphaEval* task domains, classified following O*NET occupational taxonomy (U.S. Department of Labor, 2025). Each domain maps to a standardized work activity category.

Upstream: Partner Engagement. We partner with companies deploying AI agents as core products or critical workflows for paying customers, targeting diversity in O*NET occupational domains (Table 2). Selection criteria: (1) active customer base with market-validated requirements; (2) AI agents integral to revenue-generating workflows; (3) diverse input modalities; (4) willingness to share anonymized data.

Upstream: Requirement Elicitation. For each company, we conduct structured engagement spanning approximately one month. The critical insight is that production requirements rarely arrive as well-specified task descriptions—they emerge through iterative dialogue. Companies present their workflows, often revealing gaps between described and actual evaluation processes. Through at least weekly meetings, we jointly develop task specifications that preserve the original level of under-specification, implicit constraints, and domain knowledge dependency that make production tasks fundamentally harder than research tasks.

Downstream: Task Formalization. Each task becomes a standardized, self-contained package: (1) `query.md`, preserving the partner-approved level of specification; (2) `task.yaml`, recording metadata and execution limits; (3) raw input files/; and (4) an executable `.eval/rubric.py`, optionally with fixed reference answers. Rubrics and weights are authored and validated by partner domain experts rather than generated by the judge model. Approximately 42% of tasks involve PDFs, 21% structured data, 25% text, and 12% code/YAML.

Cross-layer Validation. Frontier-agent dry runs and partner review produce three to four refinement cycles per company on average. Experts validate that fixed rubric points capture the intended acceptance criteria; during evaluation, the LLM judge only applies those rubrics. Live per-output SME scoring would provide a complementary, higher-cost validation signal and remains future work.

4 The *AlphaEval* Benchmark

AlphaEval comprises 94 tasks sourced from and validated against deployed or intended-for-deployment workflows at companies with paying customers, classified into six O*NET

occupational domains (Table 8). It preserves implicit constraints, fragmented information, undeclared domain knowledge, heterogeneous inputs, long-horizon professional deliverables, and complementary evaluation paradigms.

Task Categories. We classify tasks following the O*NET occupational taxonomy (U.S. Department of Labor, 2025), ensuring each domain maps to a standardized work activity category. This classification enables systematic coverage analysis and makes the benchmark extensible—any new production task can be mapped to an existing O*NET domain.

Human Resources (11 tasks, O*NET 13-1071): Agents screen candidate resumes against job descriptions. A representative task provides 24 resumes (PDF and JPEG) for an AI Scientist internship at a startup accelerator, mixing objective criteria (“at least one internship longer than 3 months”) with subjective ones (“passion for cutting-edge technology”). The agent must select exactly 6 candidates; evaluation computes F1 against actual interview decisions.

Finance & Investment (22 tasks, O*NET 13-2051): Agents perform investment research and financial data extraction. Tasks include: (a) generating professional segment research reports from startup business plans, following prescribed templates covering market sizing (TAM-SAM-SOM), competitive landscape, and technology deep-dive (14 tasks); (b) synthesizing unstructured meeting transcripts into actionable investor critiques—e.g., designing a 2-minute pitch storyline and assessing whether a healthcare platform’s 20,000-user reach translates to genuine product-market fit given only 70 core users (5 tasks); and (c) extracting structured financial data from multi-year corporate annual reports (3 tasks). Evaluation combines LLM-as-Judge with structural validation.

Procurement & Operations (23 tasks, O*NET 13-1020): Agents solve constrained optimization and data processing problems grounded in real procurement workflows. A representative task presents 2,000 board cards with specifications and pricing in Excel, plus a natural language requirements document with implicit constraints (20 tasks). Additional tasks involve procurement bidding data analysis and spreadsheet operations (3 tasks). Evaluation programmatically verifies constraint satisfaction and cost optimality.

Software Engineering (11 tasks, O*NET 15-1252): Agents build or modify full-stack mobile mini-programs using the UniApp framework. A representative task provides database schemas, seed data, and requirements for a poetry appreciation app with AI-powered playback, community posting, and collection management across four navigation pages. Evaluation employs automated end-to-end UI testing: a headless browser executes user flows and scores 10+ functional requirements.

Healthcare & Life Sciences (16 tasks, O*NET 29-9099): Agents handle clinical trial management and healthcare policy analysis. Tasks include: (a) computing visit windows for electronic Case Report Form (eCRF) systems, where shifting one visit propagates through subsequent windows via cascade dependencies (10 tasks); and (b) analyzing pharmaceutical reimbursement policies, medical insurance calculations, and drug coverage regulations (6 tasks). Evaluation combines LLM-as-Judge with numerical verification.

Technology Research (11 tasks, O*NET 15-1221): Agents conduct current-technology investigations using a date-filtered retrieval API bounded to each task’s construction cutoff—not unrestricted live internet. A representative task studies the status of AI-agent startups that raised over \$100M. The other five domains are fixed-input and offline. Scoring is anchored to facts and sources frozen at construction time, so later web changes cannot alter the rubric.

Evaluation Methodologies. The framework combines four broad paradigms: **Reference Answer Verification**, **Formal Logic Verification**, **Rubric-based Evaluation**, and **Execution-based Verification** (Figure 3). These labels are multi-label rather than mutually exclusive: a single task may compose several leaf-node checks, whose overlapping incidences are reported in Appendix E. Where semantic rubric scoring is used, Claude Opus 4.6 applies fixed expert-authored points; it does not author the criteria. Composition supplies (i) coverage for qualitatively different success conditions, (ii) robustness against the biases of any single evaluator, and (iii) non-redundant checks within a task—for example, structural checks can

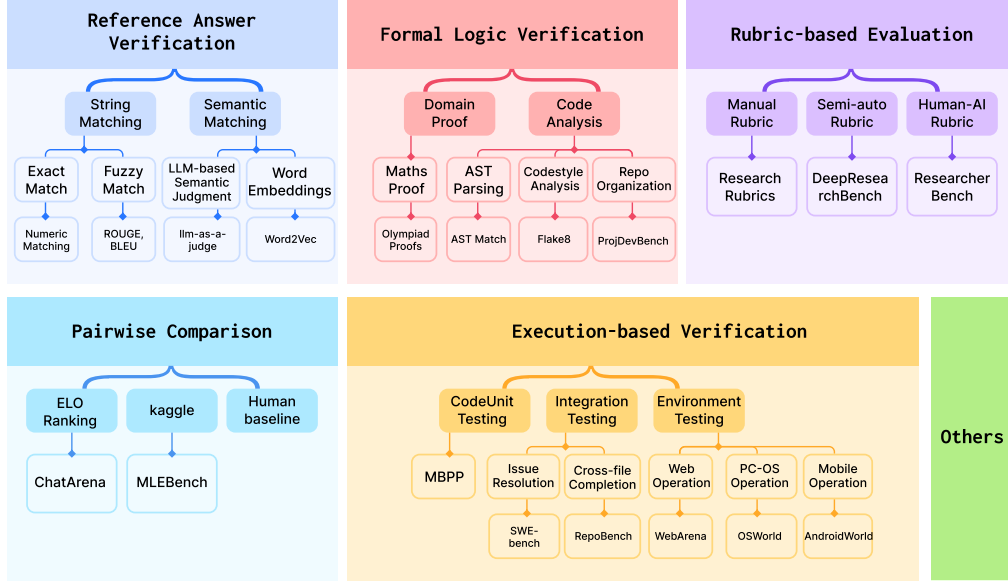


Figure 3: Taxonomy of evaluation methodologies for AI agents. *AlphaEval* covers multiple evaluation paradigms (excluding Pairwise Comparison and Others).

verify report completeness while semantic rubrics detect cross-section contradictions. Every task uses at least two leaf-node checks (mean 2.8; Table 9).

Economic Value Annotation. As a grounded proxy for task difficulty and economic significance—not direct replacement value—we estimate professional time and labor cost, then calibrate with experts (Appendix H). The 94 tasks correspond to an estimated 2,420 professional hours and \$154K–\$231K; absolute values are assumption-dependent, so we use them only as context.

5 Evaluation Infrastructure

The *AlphaEval* framework is organized around a **Task Runner**, an **Evaluator Registry**, and a Docker **Execution Sandbox**. These are enabling capabilities rather than a claim of standalone systems novelty: their role is to invoke heterogeneous commercial products through one interface, record observable trajectories, and compose evaluation paradigms reproducibly. Because Claude Code, Codex, Copilot, and Cursor are closed commercial products, their internals are unavailable; black-box product evaluation is therefore the realistic scope. Each rubric outputs $s_{\text{task}} = \sum_k w_k e_k \in [0, 1]$ with expert-assigned weights. Domain scores are task means and the overall score is the unweighted mean of six domains (Appendix C).

6 Challenges in Production Evaluation

Constructing *AlphaEval* surfaced challenges largely absent from research benchmark development: (1) *Preserving productive ambiguity*—fully disambiguating tasks destroys what makes production tasks difficult; (2) *Implicit constraints*—practitioners consider requirements “obvious” that are absent from written specifications; (3) *Quantifying subjective judgment*—experts assess quality holistically, and decomposing into verifiable dimensions inevitably loses information; (4) *Evaluation criteria drift*—partner companies revised quality standards as agent capabilities improved; (5) *Environment fidelity*—production agents operate within rich software environments difficult to reproduce in sandboxed evaluation; and (6) *Balanc-*

Agent Product	Model	HR (11)	F&I (22)	P&O (23)	SE (11)	H&LS (16)	TR (11)	Avg.
Claude Code	Claude Opus 4.6	<u>35.91</u>	70.35	83.35	70.95	50.06	<u>75.82</u>	64.41
	Gemini 3 Pro	26.73	54.68	65.39	59.47	39.88	58.55	50.78
	GPT-5.2	26.55	38.64	40.22	55.08	27.94	48.36	39.47
	Kimi K2.5	20.18	47.80	35.04	56.77	44.44	59.18	43.90
	GLM-5	31.91	51.18	66.43	52.46	38.88	51.36	48.70
	MiniMax M2.5	26.00	52.55	39.39	42.03	34.62	50.73	40.89
Codex	GPT-5.2	27.36	62.80	52.35	50.56	32.19	60.27	47.59
	GLM-5	24.36	57.85	56.74	53.58	<u>46.69</u>	59.91	49.85
	Claude Opus 4.6	32.18	45.99	79.00	54.74	36.81	72.00	53.45
	Kimi K2.5	35.36	47.27	30.91	52.52	38.75	53.73	43.09
GitHub Copilot	GPT-5.2	32.00	63.39	74.04	61.07	30.69	68.27	54.91
	Gemini 3 Pro	28.27	59.31	65.35	50.46	30.69	65.45	49.92
	Claude Opus 4.6	34.18	62.12	88.09	63.07	44.06	76.36	61.31
Cursor	Claude Opus 4.6	38.91	<u>67.10</u>	88.09	<u>64.77</u>	44.06	68.18	<u>61.85</u>

Table 3: Average scores (0–100) on *AlphaEval* across six O*NET domains. **Bold**: best in column. Underline: second best. HR=Human Resources, F&I=Finance & Investment, P&O=Procurement & Operations, SE=Software Engineering, H&LS=Healthcare & Life Sciences, TR=Technology Research.

ing openness and confidentiality—partner companies are cautious about sharing evaluation criteria encoding competitive knowledge.

7 Experiments

7.1 Setup

Models. We evaluate six frontier models: Claude Opus 4.6 (Anthropic), GPT-5.2 (OpenAI), Gemini 3 Pro Preview (Google), Kimi K2.5 (Moonshot), GLM-5 (Zhipu AI), and MiniMax M2.5 (MiniMax).

Agent Systems. Models are deployed through four commercial agent products: **Claude Code** (Anthropic), **Codex** (OpenAI), **GitHub Copilot** (GitHub), and **Cursor** (Cursor). All agents are invoked via their respective CLI interfaces within Docker-sandboxed environments. We record observable output trajectories (tool calls, exposed intermediate reasoning, and final artifacts) for each run, enabling post-hoc failure analysis. Of the $6 \times 4 = 24$ Cartesian-product pairings, we evaluate 14 selected for operational availability, partner usage, and evaluation cost; all six models and four products are represented (Table 3). A Claude Code + Opus 4.6 run averaged 14 minutes and 46 interaction turns per task, making exhaustive enumeration materially more expensive.

Generalist-human baseline. CS PhD students attempted the full benchmark at a normal working pace without agent timeouts, with one student per task; the same fixed rubrics scored human and agent outputs. This is a generalist baseline, not a domain-expert or production-workflow baseline (Appendix L).

7.2 Results and Domain-Level Analysis

Several key findings emerge from Table 3:

- **Low absolute scores.** The best configuration (Claude Code + Opus 4.6) achieves only 64.41 on average, revealing a substantial research-production gap.
- **Large scaffold effects.** Corroborating prior work (Brand & Denain, 2025; Fan et al., 2025; Meireles et al., 2025), the same Opus 4.6 scores 64.41 via Claude Code but 53.45 via Codex;

GPT-5.2 spans 39.47–54.91. Evaluating the full product, not only the model, is therefore essential.

- **Extreme domain variance.** Procurement & Operations scores range from 30.91 to 88.09, while Human Resources scores never exceed 38.91—no single score captures production readiness.
- **Rank reversals across domains.** Within Claude Code, GLM-5 exceeds Gemini 3 Pro on Procurement & Operations (66.43 vs. 65.39) but trails it on Software Engineering (52.46 vs. 59.47). Aggregate rankings conceal deployment-specific choices.

Uncertainty and multiplicity. A domain-stratified task bootstrap ($B = 10,000$) gives the headline configuration an overall 95% interval of [58.25, 70.45] (Appendix K). A paired Finance & Investment comparison provides a same-model scaffold comparison: Claude Code + Opus exceeds Copilot + Opus by 8.24 points (95% bootstrap CI [2.36, 14.56], paired permutation $p = 0.0193$), while Copilot leads on Procurement & Operations (88.09 vs. 83.35), yielding a cross-domain reversal. In an exploratory family of 546 per-domain configuration comparisons, Benjamini–Hochberg false-discovery-rate control retains 117 tests at $q = .10$ (87 at $q = .05$); the Finance & Investment comparison is rank 113 and clears the $q = .10$ threshold. We do not interpret sub-point gaps as distinguishable from noise.

We order domains from easiest to hardest (highest to lowest average score across configurations):

Technology Research (avg. 62.0): scores range from 48.36 to 76.36. Tasks require cutoff-bounded information retrieval, multi-source synthesis, and technical depth. **Procurement & Operations** (avg. 61.7): Cursor + Opus 4.6 and Copilot + Opus 4.6 reach 88.09; binary pass/fail scoring on the core optimization tasks reflects the zero-tolerance nature of procurement decisions. **Software Engineering** (avg. 56.3): even weaker models exceed 42 points, but the gap between top (70.95) and bottom (42.03) remains substantial. **Finance & Investment** (avg. 55.8): Claude Code + Opus 4.6 leads at 70.35, with strong performance from Codex configurations that achieve near-perfect scores on financial data extraction subtasks. **Healthcare & Life Sciences** (avg. 38.6): Claude Code + Opus 4.6 leads at 50.06; clinical trial eCRF tasks exhibit zero tolerance for numerical errors, while healthcare policy tasks require domain-specific regulatory knowledge. **Human Resources** (avg. 30.0): best score only 38.91—aligning agent judgments with human hiring decisions remains difficult.

7.3 Failure Mode Analysis

We identify six production-specific failure modes that are poorly captured by coding-centric benchmarks (detailed case studies in Appendix I and I.2):

(1) **Cascade dependency failure.** In Healthcare & Life Sciences, misidentifying a Day 1 anchor produces systematically incorrect window calculations for all subsequent visits. In Finance & Investment, misidentifying a company’s industry cascades through market sizing, comparable company selection, and competitive analysis.

(2) **Subjective judgment collapse.** In Human Resources, agents extract factual qualifications but fail on soft-skill inference. Tasks with quantifiable criteria score 2–3× higher than those requiring holistic judgment.

(3) **Information retrieval failures.** Technology Research tasks expose factual hallucination, imprecise retrieval, rigid search strategies, attribution confusion, and positive-information bias. The examined runs often miss negative events (startup failures and funding collapses) when default searches disproportionately surface success stories. These categories are qualitative and are not prevalence estimates.

(4) **Cross-section logical inconsistency.** In Finance & Investment, agents produce individually plausible paragraphs that contradict each other—a TAM of \$50B in one section but \$80B two pages later. Models lack a global coherence mechanism across long-form outputs.

(5) Constraint misinterpretation. In Procurement & Operations, agents optimize explicitly stated objectives while violating implicit constraints, and exhibit “synergy blindness”—optimizing components independently rather than jointly.

(6) Format compliance failures. The most production-specific failure: agents produce substantively reasonable analyses that score poorly because the output format is incompatible with downstream consumption.

8 Discussion

AlphaEval shows that partner-grounded tasks expose gaps in ambiguity handling, domain judgment, and format compliance. “Dynamic” denotes design for low-cost extension through the reusable construction pipeline, not evidence of completed maintenance. We will co-maintain tasks with partners and publish versioned releases; extending the six-domain snapshot remains ongoing work.

Meta-Evaluation. We validate evaluation reliability on 20 randomly sampled LLM-as-Judge tasks across all six domains (5 configurations, 1,000 rubric point judgments). Two independent expert annotators (strict A, lenient B) assess each rubric point alongside the automated judge.

Pair	Agr.	κ	ρ	r
A vs. B	84.7%	0.691	0.818	0.870
A vs. LLM-as-Judge	85.0%	0.697	0.820	0.861
B vs. LLM-as-Judge	89.7%	0.780	0.845	0.885
Three-way (Fleiss)	79.7%	0.720	—	—

Table 4: Meta-evaluation results: pairwise agreement between two expert annotators (A, B) and the automated LLM-as-Judge evaluation across 1,000 rubric point judgments.

All pairwise Cohen’s κ values fall within the *substantial agreement* range (0.69–0.78), with Fleiss’ $\kappa = 0.720$ confirming three-way reliability. The automated judge shows higher agreement with the lenient annotator ($\kappa = 0.780$) than the strict one ($\kappa = 0.697$), suggesting a slight leniency bias (Panickssery et al., 2024; Zheng et al., 2023).

Judge self-preference sanity check. On 50 rubric points from three LLM-judged domains, Opus 4.6 had the highest agreement with a human expert (84%, $\kappa = .58$). Its Opus-vs.-non-Opus leniency differential was +16 points, no larger than four independent judges (+20 to +28); the human expert also covered more points in Opus answers (80% vs. 68%). This small sample provides no evidence of extra self-preference by the deployed judge, but cannot rule it out (Appendix M).

9 Limitations

AlphaEval evaluates *delegatable tasks*, not the full contextual, relational, and judgment responsibilities of an occupation; a high task score does not imply that a human role can be removed wholesale. The 94 tasks and seven self-selected partners cover only six domains, so company selection, domain coverage, and generalization remain uncertain; authenticity here trades off against scale. The 14 configurations do not exhaust all 24 model–product pairings, and untested pairings may behave differently. The human comparison is a generalist CS-PhD single-attempt baseline rather than domain experts or incumbent production workflows. Economic value is an assumption-dependent proxy, not realized value or replacement cost. LLM judging remains vulnerable to bias despite meta-evaluation and the small sanity check. Finally, closed commercial products restrict us to observable black-box evaluation, and reproducible rubric-based scoring does not replace live SME review of individual outputs.

10 Conclusion

We present *AlphaEval*, 94 partner-grounded tasks across six O*NET domains, and a two-layer construction framework separating human requirement elicitation from automated execution. Evaluation of 14 product configurations yields a best score of 64.41/100, large domain and scaffold effects, and six recurring production failure modes. The public release includes the evaluation framework, task templates, and representative fictional examples; broader task release remains subject to partner permissions and privacy constraints.

LLM Assistance Disclosure

Language-model tools assisted with editing, consistency checking, and drafting revisions to this manuscript. The authors reviewed the resulting text and take responsibility for all claims, citations, analyses, and the final paper. The use of an LLM as an evaluation component in the research itself is described in Sections 4 and M.

Acknowledgments

We thank Keyu Li for his valuable guidance on the design and development of the evaluation framework, Zhen Huang for his contributions to surveying existing agent benchmarks and preparing the benchmark comparison table, and Tianze Xu for his contributions during the early stages of this project.

References

- AgentIF Team. Agentif-oneday: A task-level instruction-following benchmark for general ai agents in daily scenarios. *arXiv preprint arXiv:2601.20613*, 2026. URL <https://arxiv.org/abs/2601.20613>.
- Victor Barres et al. τ^2 -bench: Evaluating conversational agents in a dual-control environment. *arXiv preprint arXiv:2506.07982*, 2025. URL <https://arxiv.org/abs/2506.07982>.
- Jonathan Bragg, Mike D’Arcy, Nishant Balepur, Dan Bareket, Bhavana Dalvi, et al. Astabench: Rigorous benchmarking of ai agents with a scientific research suite. *arXiv preprint arXiv:2510.21652*, 2025. URL <https://arxiv.org/abs/2510.21652>.
- Florian Brand and Jean-Stanislas Denain. What skills does SWE-bench Verified evaluate? Epoch AI, 2025. URL <https://epoch.ai/publications/what-skills-does-swe-bench-verified-evaluate>.
- Keita Broadwater. Evaluating llm safety under repeated inference via accelerated prompt stress testing. *arXiv preprint arXiv:2602.11786*, 2026. URL <https://arxiv.org/abs/2602.11786>.
- Yuan Cao, Dezhi Ran, Mengzhou Wu, Yuzhe Guo, Xin Chen, et al. Gui-genesis: Automated synthesis of efficient environments with verifiable rewards for gui agent post-training. *arXiv preprint arXiv:2602.14093*, 2026. URL <https://arxiv.org/abs/2602.14093>.
- Arnold Cartagena and Ariane Teixeira. Mind the gap: Text safety does not transfer to tool-call safety in llm agents. *arXiv preprint arXiv:2602.16943*, 2026. URL <https://arxiv.org/abs/2602.16943>.
- Jun Shern Chan et al. Mle-bench: Evaluating machine learning agents on machine learning engineering. *arXiv preprint arXiv:2410.07095*, 2024. URL <https://arxiv.org/abs/2410.07095>.
- Chen Chen, Xinlong Hao, Weiwen Liu, Xu Huang, Xingshan Zeng, et al. Acebench: Who wins the match point in tool usage? *arXiv preprint arXiv:2501.12851*, 2025a. URL <https://arxiv.org/abs/2501.12851>.

- Haorui Chen, Chengze Li, and Jia Li. Featbench: Towards more realistic evaluation of feature-level code generation. *arXiv preprint arXiv:2509.22237*, 2025b. URL <https://arxiv.org/abs/2509.22237>.
- Tingting Chen, Srinivas Anumasa, Beibei Lin, Vedant Shah, Anirudh Goyal, and Dianbo Liu. Auto-bench: An automated benchmark for scientific discovery in llms. *arXiv preprint arXiv:2502.15224*, 2025c. URL <https://arxiv.org/abs/2502.15224>.
- François Chollet and ARC Prize Team. Arc-agi-2: A new challenge for frontier ai reasoning systems. *arXiv preprint arXiv:2505.11831*, 2025. URL <https://arxiv.org/abs/2505.11831>.
- Yun-Shiuan Chuang, Chaitanya Kulkarni, Alec Chiu, Avinash Thangali, Zijie Pan, et al. Toward scalable verifiable reward: Proxy state-based evaluation for multi-turn tool-calling llm agents. *arXiv preprint arXiv:2602.16246*, 2026. URL <https://arxiv.org/abs/2602.16246>.
- Pepijn Cobben, Xuanqiang Angelo Huang, Thao Amelia Pham, Isabel Dahlgren, Terry Jingchen Zhang, and Zhijing Jin. Gt-harmbench: Benchmarking ai safety risks through the lens of game theory. *arXiv preprint arXiv:2602.12316*, 2026. URL <https://arxiv.org/abs/2602.12316>.
- Cybergod AGI Research. Vimgolff-gym: Openai gym style vimgolff environment and benchmark, 2025. URL <https://github.com/james4Ever0/vimgolff-gym>.
- Databricks. Introducing officeqa: A benchmark for end-to-end grounded reasoning. <https://www.databricks.com/blog/introducing-officeqa-benchmark-end-to-end-grounded-reasoning>, 2025.
- Mariya Davydova, Daniel Jeffries, Patrick Barker, Arturo Márquez Flores, and Sinéad Ryan. Osuniverse: Benchmark for multimodal gui-navigation ai agents. *arXiv preprint arXiv:2505.03570*, 2025. URL <https://arxiv.org/abs/2505.03570>.
- Xiang Deng et al. Swe-bench pro: Can ai agents solve long-horizon software engineering tasks? *arXiv preprint arXiv:2509.16941*, 2025. URL <https://arxiv.org/abs/2509.16941>. 1,865 tasks, avg 107.4 lines across 4.1 files, enterprise-grade.
- Hongjun Ding, Binqi Chen, Jinsheng Huang, Taian Guo, Zhengyang Mao, Guoyi Shao, Lutong Zou, Luchen Liu, and Ming Zhang. AlphaEval: A comprehensive and efficient evaluation framework for formula alpha mining, 2025. URL <https://arxiv.org/abs/2508.13174>.
- Titouan Duston, Shuo Xin, Yang Sun, Daoguang Zan, Aoyan Li, et al. Ainsteinbench: Benchmarking coding agents on scientific repositories. *arXiv preprint arXiv:2512.21373*, 2025. URL <https://arxiv.org/abs/2512.21373>.
- Zhiyu Fan, Kirill Vasilevski, Dayi Lin, Boyuan Chen, Yihao Chen, Zhiqing Zhong, Jie M. Zhang, Pinjia He, and Ahmed E. Hassan. SWE-Effi: Re-evaluating software ai agent system effectiveness under resource constraints, 2025. URL <https://arxiv.org/abs/2509.09853>.
- Yukang Feng, Jianwen Sun, Zelai Yang, Jiaxin Ai, Chuanhao Li, et al. Longcli-bench: A preliminary benchmark and study for long-horizon agentic programming in command-line interfaces. *arXiv preprint arXiv:2602.14337*, 2026. URL <https://arxiv.org/abs/2602.14337>.
- Romain Froger, Pierre Andrews, Matteo Bettini, Amar Budhiraja, Ricardo Silveira Cabral, et al. Gaia2: Benchmarking llm agents on dynamic and asynchronous environments. *arXiv preprint arXiv:2602.11964*, 2026. URL <https://arxiv.org/abs/2602.11964>.
- Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. *arXiv preprint arXiv:2405.21075*, 2024. URL <https://arxiv.org/abs/2405.21075>.

- Aniketh Garikaparathi, Manasi Patwardhan, and Arman Cohan. Researchgym: Evaluating language model agents on real-world ai research. *arXiv preprint arXiv:2602.15112*, 2026. URL <https://arxiv.org/abs/2602.15112>.
- Paul Gauthier. Aider polyglot benchmark. <https://aider.chat/2024/12/21/polyglot.html>, 2024.
- Elliot Glazer, Ege Erdil, Tamay Besiroglu, Diego Chicharro, Evan Chen, et al. Frontiermath: A benchmark for evaluating advanced mathematical reasoning in ai. *arXiv preprint arXiv:2411.04872*, 2024. URL <https://arxiv.org/abs/2411.04872>.
- Thibaud Gloaguen, Niels Mündler, Mark Müller, Veselin Raychev, and Martin Vechev. Evaluating agents.md: Are repository-level context files helpful for coding agents? *arXiv preprint arXiv:2602.11988*, 2026. URL <https://arxiv.org/abs/2602.11988>.
- Harvard-MIT. Harvard-mit mathematics tournament 2025. <https://www.hmmt.org/>, 2025.
- Zexue He, Yu Wang, Churan Zhi, Yuanzhe Hu, Tzu-Ping Chen, et al. Memoryarena: Benchmarking agent memory in interdependent multi-session agentic tasks. *arXiv preprint arXiv:2602.16313*, 2026. URL <https://arxiv.org/abs/2602.16313>.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020. URL <https://arxiv.org/abs/2009.03300>.
- Lingxiang Hu, Yiding Sun, Tianle Xia, Wenwei Li, Ming Xu, et al. Ad-bench: A real-world, trajectory-aware advertising analytics benchmark for llm agents. *arXiv preprint arXiv:2602.14257*, 2026. URL <https://arxiv.org/abs/2602.14257>.
- Tianyu Hua, Harper Hua, Violet Xiang, Benjamin Klieger, Sang T. Truong, et al. Research-codebench: Benchmarking llms on implementing novel machine learning research code. *arXiv preprint arXiv:2506.02314*, 2025. URL <https://arxiv.org/abs/2506.02314>.
- Naman Jain, King Han, Alex Gu, et al. Livecodebench: Holistic and contamination free evaluation of llms for code. *arXiv preprint arXiv:2403.07974*, 2024. URL <https://arxiv.org/abs/2403.07974>.
- JetBrains. Dpaa arena. <https://dpaa.dev/>, 2025.
- Tanqiu Jiang, Yuhui Wang, Jiacheng Liang, and Ting Wang. Agentlab: Benchmarking llm agents against long-horizon attacks. *arXiv preprint arXiv:2602.16901*, 2026. URL <https://arxiv.org/abs/2602.16901>.
- Carlos E Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. Swe-bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*, 2024. URL <https://arxiv.org/abs/2310.06770>.
- Liqiang Jing et al. Dsbench: How far are data science agents from becoming data science experts? *arXiv preprint arXiv:2409.07703*, 2024. URL <https://arxiv.org/abs/2409.07703>.
- Sayash Kapoor, Peter Kirgis, Andrew Schwartz, Stephan Rabanser, J. J. Allaire, Rishi Bommasani, Magda Dubois, Gillian Hadfield, Andy Hall, Sara Hooker, et al. Open-world evaluations for measuring frontier AI capabilities, 2026. URL <https://arxiv.org/abs/2605.20520>.
- Serin Kim, Sangam Lee, and Dongha Lee. Persona2web: Benchmarking personalized web agents for contextual reasoning with user history. *arXiv preprint arXiv:2602.17003*, 2026a. URL <https://arxiv.org/abs/2602.17003>.
- Sunghwan Kim, Ryang Heo, Yongsik Seo, Jinyoung Yeo, and Dongha Lee. Agenticshop: Benchmarking agentic product curation for personalized web shopping. *arXiv preprint arXiv:2602.12315*, 2026b. URL <https://arxiv.org/abs/2602.12315>.

- Patrick Tser Jern Kon, Jiachen Liu, Xinyi Zhu, Qiuyi Ding, Jingjia Peng, et al. Exp-bench: Can ai conduct ai research experiments? *arXiv preprint arXiv:2505.24785*, 2025. URL <https://arxiv.org/abs/2505.24785>.
- Laude Institute. Terminal-bench. <https://github.com/laude-institute/terminal-bench>, 2025.
- Fangyu Lei, Jinxiang Meng, Yiming Huang, Junjie Zhao, Yitong Zhang, et al. Dacomp: Benchmarking data agents across the full data intelligence lifecycle. *arXiv preprint arXiv:2512.04324*, 2025. URL <https://arxiv.org/abs/2512.04324>.
- Letta. Context-bench: A benchmark for agentic context engineering. <https://www.sundeepTEKI.org/blog/context-bench>, 2025a.
- Letta. Letta evals: Evaluating agents that learn. <https://www.letta.com/blog/letta-evals>, 2025b.
- Bowen Li, Wenhan Wu, Ziwei Tang, Lin Shi, John Yang, Jinyang Li, Shunyu Yao, Chen Qian, Binyuan Hui, Qicheng Zhang, Zhiyin Yu, He Du, Ping Yang, Dahua Lin, Chao Peng, and Kai Chen. Prompting large language models to tackle the full software development lifecycle: A case study. *arXiv preprint arXiv:2403.08604*, 2024. URL <https://arxiv.org/abs/2403.08604>.
- Eric Li and Jim Waldo. Websuite: Systematically evaluating why web agents fail. *arXiv preprint arXiv:2406.01623*, 2024. URL <https://arxiv.org/abs/2406.01623>.
- Junlong Li, Wenshuo Zhao, Jian Zhao, Weihao Zeng, Haoze Wu, et al. The tool decathlon: Benchmarking language agents for diverse, realistic, and long-horizon task execution. *arXiv preprint arXiv:2510.25726*, 2025. URL <https://arxiv.org/abs/2510.25726>.
- Keyu Li, Junhao Shi, Yang Xiao, Mohan Jiang, et al. Agencybench: Benchmarking the frontiers of autonomous agents in 1m-token real-world contexts. *arXiv preprint arXiv:2601.11044*, 2026a. URL <https://arxiv.org/abs/2601.11044>.
- Xiangyi Li, Wenbo Chen, Yimin Liu, Shenghan Zheng, Xiaokun Chen, et al. Skills-bench: Benchmarking how well agent skills work across diverse tasks. *arXiv preprint arXiv:2602.12670*, 2026b. URL <https://arxiv.org/abs/2602.12670>.
- Xu Li, Simon Yu, Minzhou Pan, Yiyu Sun, Bo Li, et al. Unsafer in many turns: Benchmarking and defending multi-turn safety risks in tool-using agents. *arXiv preprint*, 2026c. URL <https://github.com/CHATS-lab/ToolShield>.
- Kaiqu Liang, Julia Kruk, Shengyi Qian, Xianjun Yang, Shengjie Bi, et al. Learning personalized agents from human feedback. *arXiv preprint arXiv:2602.16173*, 2026. URL <https://arxiv.org/abs/2602.16173>.
- Ling Lin, Yang Bai, Heng Su, Congcong Zhu, Yaoxing Wang, et al. Oodbench: Out-of-distribution benchmark for large vision-language models. *arXiv preprint arXiv:2602.18094*, 2026. URL <https://arxiv.org/abs/2602.18094>.
- 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. *arXiv preprint arXiv:2308.03688*, 2023. URL <https://arxiv.org/abs/2308.03688>. Published at ICLR 2024.
- Lajanugen Logeswaran, Jaekyeom Kim, Sungryull Sohn, Creighton Glasscock, and Honglak Lee. Scaling web agent training through automatic data generation and fine-grained evaluation. *arXiv preprint arXiv:2602.12544*, 2026. URL <https://arxiv.org/abs/2602.12544>.
- Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. *arXiv preprint arXiv:2310.02255*, 2024. URL <https://arxiv.org/abs/2310.02255>.

- Pengrui Lu, Shiqi Zhang, Yunzhong Hou, Lyumanshan Ye, Chaoyi Huang, et al. Projdevbench: Benchmarking ai coding agents on end-to-end project development. *arXiv preprint arXiv:2602.01655*, 2026. URL <https://arxiv.org/abs/2602.01655>.
- Ziyang Luo, Zhiqi Shen, Wenzhuo Yang, Zirui Zhao, Prathyusha Jwalapuram, Amrita Saha, Doyen Sahoo, Silvio Savarese, Caiming Xiong, and Junnan Li. Mcp-universe: Benchmarking large language models with real-world model context protocol servers. *arXiv preprint arXiv:2508.14704*, 2025. URL <https://arxiv.org/abs/2508.14704>. 6 domains, 11 MCP servers, 231 tasks.
- Chang Ma et al. Agentboard: An analytical evaluation board of multi-turn llm agents. *OpenReview*, 2024. URL <https://openreview.net/pdf?id=4S8agvKjle>.
- MAA. American invitational mathematics examination (aime) 2025. <https://www.kaggle.com/benchmarks/open-benchmarks/aime-2025>, 2025.
- Lingjun Mao, Jiawei Ren, Kun Zhou, Jixuan Chen, Ziqiao Ma, and Lianhui Qin. Delivery-bench: Can agents earn profit in real world? *arXiv preprint arXiv:2512.19234*, 2025. URL <https://arxiv.org/abs/2512.19234>.
- Mariana Meireles, Rupali Bhati, Niklas Lauffer, and Cameron Allen. The influence of scaffolds on coordination scaling laws in LLM agents. NeurIPS 2025 MTI-LLM Workshop, 2025. URL <https://openreview.net/forum?id=iS49teIy7h>.
- Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: A benchmark for general ai assistants. *arXiv preprint arXiv:2311.12983*, 2023. URL <https://arxiv.org/abs/2311.12983>.
- Rui Min, Zile Qiao, Ze Xu, Jiawen Zhai, Wenyu Gao, et al. Ecombench: Towards holistic evaluation of foundation agents in e-commerce. *arXiv preprint arXiv:2512.08868*, 2025. URL <https://arxiv.org/abs/2512.08868>. 7 categories, quarterly updates, human-in-the-loop framework.
- Samuel Miserendino, Michele Wang, Tejal Patwardhan, and Johannes Heidecke. Swe-lancer: Can frontier llms earn \$1 million from real-world freelance software engineering? *arXiv preprint arXiv:2502.12115*, 2025. URL <https://arxiv.org/abs/2502.12115>.
- Niels Mundler et al. Swt-bench: Testing and validating real-world bug-fixes with code agents. *OpenReview*, 2024. URL <https://openreview.net/forum?id=9Y8zuU011EQ>.
- OpenAI. Mmmlu: Massive multilingual multitask language understanding. *HuggingFace Dataset*, 2024. URL <https://huggingface.co/datasets/openai/MMMLU>.
- OpenAI. Openai-mrcr: Multi-round coreference resolution, 2025. URL <https://huggingface.co/datasets/openai/mrcr>.
- Anne Ouyang, Simon Guo, Simran Arora, Alex L. Zhang, William Hu, et al. Kernelbench: Can llms write efficient gpu kernels? *arXiv preprint arXiv:2502.10517*, 2025. URL <https://arxiv.org/abs/2502.10517>.
- Melissa Z. Pan et al. Measuring agents in production. *arXiv preprint arXiv:2512.04123*, 2025. URL <https://arxiv.org/abs/2512.04123>. Survey of 306 practitioners with 20 in-depth case studies.
- Arjun Panickssery, Samuel R Bowman, and Shi Feng. Llm evaluators recognize and favor their own generations. *arXiv preprint arXiv:2404.13076*, 2024. URL <https://arxiv.org/abs/2404.13076>. Published at NeurIPS 2024.
- Dhaval Patel, Shuxin Lin, James Rayfield, Nianjun Zhou, Chathurangi Shyalika, Suryanarayana R. Yarrabothula, Roman Vaculin, Natalia Martinez, Fearghal O'Donncha, and Jayant Kalagnanam. AssetOpsBench: A real-world evaluation benchmark for AI-driven task automation in industrial asset management. *arXiv preprint arXiv:2506.03828*, 2026. URL <https://arxiv.org/abs/2506.03828>. Accepted to the KDD 2026 Datasets and Benchmarks Track.

- Shishir G. Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, et al. Bfcl: From tool use to agentic evaluation of large language models, 2025. URL https://gorilla.cs.berkeley.edu/blogs/15.bfcl_v4_web_search.html.
- Ivo Petrov, Jasper Dekoninck, Lyuben Baltadzhiev, Maria Drencheva, Kristian Minchev, et al. Proof or bluff? evaluating llms on 2025 usa math olympiad. *arXiv preprint arXiv:2503.21934*, 2025. URL <https://arxiv.org/abs/2503.21934>.
- Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, et al. Humanity’s last exam. *arXiv preprint arXiv:2501.14249*, 2025. URL <https://arxiv.org/abs/2501.14249>.
- Shuofei Qiao, Runnan Fang, Zhisong Qiu, Xiaobin Wang, Ningyu Zhang, et al. Benchmarking agentic workflow generation. *arXiv preprint arXiv:2410.07869*, 2024. URL <https://arxiv.org/abs/2410.07869>.
- Shanghaoran Quan, Jiaxi Yang, Bowen Yu, Bo Zheng, Dayiheng Liu, et al. Codeelo: Benchmarking competition-level code generation of llms with human-comparable elo ratings. *arXiv preprint arXiv:2501.01257*, 2025. URL <https://arxiv.org/abs/2501.01257>.
- David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023. URL <https://arxiv.org/abs/2311.12022>.
- Gabriel Roccabruna, Olha Khomyn, and Giuseppe Riccardi. Mateo: A multimodal benchmark for temporal reasoning and planning in lvlms. *arXiv preprint arXiv:2602.14589*, 2026. URL <https://arxiv.org/abs/2602.14589>.
- Yujiong Shen, Yajie Yang, Zhiheng Xi, Binze Hu, Huayu Sha, et al. Sciagentgym: Benchmarking multi-step scientific tool-use in llm agents. *arXiv preprint arXiv:2602.12984*, 2026. URL <https://arxiv.org/abs/2602.12984>.
- Zachary S. Siegel, Sayash Kapoor, Nitya Nagdir, Benedikt Stroebel, and Arvind Narayanan. Core-bench: Fostering the credibility of published research through a computational reproducibility agent benchmark. *arXiv preprint arXiv:2409.11363*, 2024. URL <https://arxiv.org/abs/2409.11363>.
- Spring Community. Spring ai bench, 2025. URL <https://github.com/spring-ai-community/spring-ai-bench>.
- Giulio Starace, Oliver Jaffe, Dane Sherburn, James Aung, Jun Shern Chan, et al. Paperbench: Evaluating ai’s ability to replicate ai research. *arXiv preprint arXiv:2504.01848*, 2025. URL <https://arxiv.org/abs/2504.01848>.
- Jiazheng Sun, Mingxuan Li, Yingying Zhang, Jiayang Niu, Yachen Wu, et al. Ambibench: Benchmarking mobile gui agents beyond one-shot instructions in the wild. *arXiv preprint arXiv:2602.11750*, 2026. URL <https://arxiv.org/abs/2602.11750>.
- Nivya Talokar, Ayush K Tarun, Murari Mandal, Maksym Andriushchenko, and Antoine Bosselut. Helpful to a fault: Measuring illicit assistance in multi-turn, multilingual llm agents. *arXiv preprint arXiv:2602.16346*, 2026. URL <https://arxiv.org/abs/2602.16346>.
- Haibo Tong, Feifei Zhao, et al. Foresightsafety bench: A frontier risk evaluation and governance framework towards safe ai. *arXiv preprint arXiv:2602.14135*, 2026. URL <https://arxiv.org/abs/2602.14135>.
- Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, et al. Appworld: A controllable world of apps and people for benchmarking interactive coding agents. *arXiv preprint arXiv:2407.18901*, 2024. URL <https://arxiv.org/abs/2407.18901>.
- U.S. Department of Labor. O*NET OnLine, 2025. URL <https://www.onetonline.org/>. Accessed: 2026-03-01.

- Alisa Vinogradova, Vlad Vinogradov, Luba Greenwood, Ilya Yasny, Dmitry Kobzyev, et al. Hunt globally: Wide search ai agents for drug asset scouting in investing, business development, and competitive intelligence. *arXiv preprint arXiv:2602.15019*, 2026. URL <https://arxiv.org/abs/2602.15019>.
- Jun Wang, Jiamu Zhou, Muning Wen, Xiaoyun Mo, Haoyu Zhang, et al. Hammerbench: Fine-grained function-calling evaluation in real mobile device scenarios. *arXiv preprint arXiv:2412.16516*, 2024. URL <https://arxiv.org/abs/2412.16516>.
- Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Ji-Rong Wen. A survey on large language model based autonomous agents, 2025a. URL <https://arxiv.org/abs/2308.11432>.
- Weixuan Wang, Dongge Han, Daniel Madrigal Diaz, Jin Xu, Victor Rühle, and Saravan Rajmohan. Odysseybench: Evaluating llm agents on long-horizon complex office application workflows. *arXiv preprint arXiv:2508.09124*, 2025b. URL <https://arxiv.org/abs/2508.09124>.
- Zora Zhiruo Wang, Sanidhya Vijayvargiya, Aspen Chen, Hanmo Zhang, Venu Arvind Arangarajan, Jett Chen, Valerie Chen, Diyi Yang, Daniel Fried, and Graham Neubig. How well does agent development reflect real-world work? *arXiv preprint arXiv:2603.01203*, 2026. URL <https://arxiv.org/abs/2603.01203>.
- Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, et al. Browsecomp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516*, 2025. URL <https://arxiv.org/abs/2504.12516>.
- Richard Willis, Jianing Zhao, Yali Du, and Joel Z. Leibo. Evaluating collective behaviour of hundreds of llm agents. *arXiv preprint arXiv:2602.16662*, 2026. URL <https://arxiv.org/abs/2602.16662>.
- xbench Team. xbench: Tracking agents productivity scaling with profession-aligned real-world evaluations. *arXiv preprint arXiv:2506.13651*, 2025. URL <https://arxiv.org/abs/2506.13651>.
- Zhiheng Xi et al. The rise and potential of large language model based agents: A survey. *arXiv preprint arXiv:2309.07864*, 2023. URL <https://arxiv.org/abs/2309.07864>.
- Zikai Xiao, Jianhong Tu, Chuhan Zou, Yuxin Zuo, Zhi Li, et al. Webworld: A large-scale world model for web agent training. *arXiv preprint arXiv:2602.14721*, 2026. URL <https://arxiv.org/abs/2602.14721>.
- Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, et al. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *arXiv preprint arXiv:2404.07972*, 2024. URL <https://arxiv.org/abs/2404.07972>.
- Frank F Xu et al. Theagentcompany: Benchmarking llm agents on consequential real world tasks. *arXiv preprint arXiv:2412.14161*, 2024. URL <https://arxiv.org/abs/2412.14161>.
- John Yang, Carlos E Jimenez, et al. Swe-bench multimodal: Do ai systems generalize to visual software domains? *arXiv preprint arXiv:2410.03859*, 2024. URL <https://arxiv.org/abs/2410.03859>.
- Qianyu Yang, Yang Liu, Jiaqi Li, Jun Bai, Hao Chen, Kaiyuan Chen, et al. \$onemillion-bench: How far are language agents from human experts? *arXiv preprint arXiv:2603.07980*, 2026. URL <https://arxiv.org/abs/2603.07980>.
- Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ -bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024. URL <https://arxiv.org/abs/2406.12045>.

- Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. *arXiv preprint arXiv:2311.16502*, 2023. URL <https://arxiv.org/abs/2311.16502>.
- Daoguang Zan, Zhirong Huang, Wei Liu, Hanwu Chen, Linhao Zhang, et al. Multi-swebench: A multilingual benchmark for issue resolving. *arXiv preprint arXiv:2504.02605*, 2025. URL <https://arxiv.org/abs/2504.02605>.
- Andy Zhang et al. Cybench: A framework for evaluating cybersecurity capabilities and risks of language models. *arXiv preprint arXiv:2408.08926*, 2024. URL <https://arxiv.org/abs/2408.08926>.
- Andy K. Zhang, Joey Ji, Celeste Menders, Riya Dulepet, Thomas Qin, Ron Y. Wang, Junrong Wu, Kyleen Liao, et al. Bountybench: Dollar impact of ai agent attackers and defenders on real-world cybersecurity systems. *arXiv preprint arXiv:2505.15216*, 2025. URL <https://arxiv.org/abs/2505.15216>.
- Huanyao Zhang, Jiepeng Zhou, Bo Li, Bowen Zhou, Yanzhe Shan, et al. Browsecomp-v³: A visual, vertical, and verifiable benchmark for multimodal browsing agents. *arXiv preprint arXiv:2602.12876*, 2026. URL <https://arxiv.org/abs/2602.12876>.
- Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*, 2023. URL <https://arxiv.org/abs/2306.05685>. Published at NeurIPS 2023.
- Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, et al. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*, 2024. URL <https://arxiv.org/abs/2307.13854>.
- Yuxuan Zhu, Tengjun Jin, Yada Pruksachatkun, Andy Zhang, Shu Liu, Sasha Cui, Sayash Kapoor, Shayne Longpre, Kevin Meng, Rebecca Weiss, et al. Establishing best practices for building rigorous agentic benchmarks. In *Advances in Neural Information Processing Systems*, 2025. URL <https://openreview.net/forum?id=E58HNCqoaA>.
- Mingchen Zhuge, Changsheng Zhao, Dylan Ashley, Wenyi Wang, Dmitrii Khizbullin, Yunsang Xiong, Zechun Liu, Ernie Chang, Raghuraman Krishnamoorthi, Yuandong Tian, Yangyang Shi, Vikas Chandra, and Jürgen Schmidhuber. Agent-as-a-judge: Evaluate agents with agents, 2024. URL <https://arxiv.org/abs/2410.10934>.

A Comprehensive Benchmark Comparison

Table 5: Comparison across seven descriptive dimensions. **Prod.** = tasks sourced from and validated against an actual deployed or intended-for-deployment workflow at a partner company with paying customers (not merely simulated or workflow-inspired); **M.M.** = heterogeneous multi-modal inputs; **Under.** = preserved production-level ambiguity; **Div.E.** = three or more complementary evaluation paradigms; **Exp.V.** = criteria co-developed and validated with domain experts; **Dyn.** = designed for low-cost continuous extension through a reusable construction pipeline, not evidence of completed maintenance; **Cross.** = three or more professional domains.

Benchmark	Year	Domain	Eval Type	Prod.	M.M.	Under.	Div.E.	Exp.V.	Dyn.	Cross.
Software Engineering & Coding										
SWE-bench (Jimenez et al., 2024)	2024	Code	Rule	x	x	x	x	x	x	x
SWE-bench Multimodal (Yang et al., 2024)	2024	Code	Rule	x	✓	x	x	x	x	x
Multi-SWE-bench (Zan et al., 2025)	2025	Code	Rule	x	x	x	x	x	✓	x
SWE-bench Pro (Deng et al., 2025)	2025	Code	Rule	x	x	x	x	x	x	x

Continued on next page

Table 5 continued from previous page

Benchmark	Year	Domain	Eval Type	Prod.	M.M.	Under.	Div.E.	Exp.V.	Dyn.	Cross.
SWE-Lancer (Miserendino et al., 2025)	2025	Code	Rule	✓	✓	✓	✓	✓	✓	✓
SWT-Bench (Mundler et al., 2024)	2024	Code	Rule	✓	✓	✓	✓	✓	✓	✓
Terminal-bench (Laude Institute, 2025)	2025	Code	Rule	✓	✓	✓	✓	✓	✓	✓
FeatBench (Chen et al., 2025b)	2025	Code	Rule	✓	✓	✓	✓	✓	✓	✓
DevEval (Li et al., 2024)	2024	Code	R+M+H	✓	✓	✓	✓	✓	✓	✓
DevAI (Zhuge et al., 2024)	2024	Code	Agent Judge	✓	✓	✓	✓	✓	✓	✓
LongCLI-Bench (Feng et al., 2026)	2026	Code	Rule	✓	✓	✓	✓	✓	✓	✓
ProjDevBench (Lu et al., 2026)	2026	Code	Rule+Model	✓	✓	✓	✓	✓	✓	✓
Data Science & ML Engineering										
DSBench (Jing et al., 2024)	2024	Code	Model	✓	✓	✓	✓	✓	✓	✓
MLE-bench (Chan et al., 2024)	2024	Code	Rule	✓	✓	✓	✓	✓	✓	✓
KernelBench (Ouyang et al., 2025)	2025	Code	Rule	✓	✓	✓	✓	✓	✓	✓
DAComp (Lei et al., 2025)	2025	Code+Research	Rule+Model	✓	✓	✓	✓	✓	✓	✓
Code Competition & Security										
LiveCodeBench (Jain et al., 2024)	2024	Code	Rule	✓	✓	✓	✓	✓	✓	✓
CodeElo (Quan et al., 2025)	2025	Code	Rule	✓	✓	✓	✓	✓	✓	✓
Aider Polyglot (Gauthier, 2024)	2024	Code	Rule	✓	✓	✓	✓	✓	✓	✓
CyBench (Zhang et al., 2024)	2024	Code	Rule	✓	✓	✓	✓	✓	✓	✓
BountyBench (Zhang et al., 2025)	2025	Code	Rule	✓	✓	✓	✓	✓	✓	✓
VimGolf-Gym (Cybergod AGI Research, 2025)	2025	Code	Rule	✓	✓	✓	✓	✓	✓	✓
DPAI Arena (JetBrains, 2025)	2025	Code	Rule	✓	✓	✓	✓	✓	✓	✓
Spring AI Bench (Spring Community, 2025)	2025	Code	Rule	✓	✓	✓	✓	✓	✓	✓
AGENTS.md Eval (Gloaguen et al., 2026)	2026	Code	Rule	✓	✓	✓	✓	✓	✓	✓
Tool Use & Web Interaction										
WebArena (Zhou et al., 2024)	2024	Web	Rule+Model	✓	✓	✓	✓	✓	✓	✓
AgentBench (Liu et al., 2023)	2023	Tool	Rule	✓	✓	✓	✓	✓	✓	✓
AgentBoard (Ma et al., 2024)	2024	Tool	Rule	✓	✓	✓	✓	✓	✓	✓
τ-bench (Yao et al., 2024)	2024	Tool	Rule	✓	✓	✓	✓	✓	✓	✓
τ ² -Bench (Barres et al., 2025)	2025	Tool	Rule	✓	✓	✓	✓	✓	✓	✓
TheAgentCompany (Xu et al., 2024)	2024	Tool	Rule+Model	✓	✓	✓	✓	✓	✓	✓
Tool Decathlon (Li et al., 2025)	2025	Tool	Rule	✓	✓	✓	✓	✓	✓	✓
ACEBench (Chen et al., 2025a)	2025	Tool	Rule+Model	✓	✓	✓	✓	✓	✓	✓
MCP-Universe (Luo et al., 2025)	2025	Tool	Rule	✓	✓	✓	✓	✓	✓	✓
BFCL (Patil et al., 2025)	2025	Tool	Rule	✓	✓	✓	✓	✓	✓	✓
Context-Bench (Letta, 2025a)	2025	Tool	Rule	✓	✓	✓	✓	✓	✓	✓
Letta Evals (Letta, 2025b)	2025	Tool	Rule+Model	✓	✓	✓	✓	✓	✓	✓
EcomBench (Min et al., 2025)	2025	Tool	Model	✓	✓	✓	✓	✓	✓	✓
DeliveryBench (Mao et al., 2025)	2025	Tool	Rule	✓	✓	✓	✓	✓	✓	✓
WorFBench (Qiao et al., 2024)	2024	Tool	Rule	✓	✓	✓	✓	✓	✓	✓
BrowseComp (Wei et al., 2025)	2025	Search	Model	✓	✓	✓	✓	✓	✓	✓
AgencyBench (Li et al., 2026a)	2026	Code+Tool	Rule+Model	✓	✓	✓	✓	✓	✓	✓
Operating System & GUI										
GAIA (Mialon et al., 2023)	2023	OS	Rule	✓	✓	✓	✓	✓	✓	✓
OSWorld (Xie et al., 2024)	2024	OS	Rule	✓	✓	✓	✓	✓	✓	✓
AppWorld (Trivedi et al., 2024)	2024	OS	Rule	✓	✓	✓	✓	✓	✓	✓
WebSuite (Li & Waldo, 2024)	2024	OS	Rule	✓	✓	✓	✓	✓	✓	✓
HammerBench (Wang et al., 2024)	2024	OS	Model	✓	✓	✓	✓	✓	✓	✓
OSUniverse (Davydova et al., 2025)	2025	OS	R+M+H	✓	✓	✓	✓	✓	✓	✓
OdysseyBench (Wang et al., 2025b)	2025	OS	Rule	✓	✓	✓	✓	✓	✓	✓
OfficeQA (Databricks, 2025)	2025	OS	Rule	✓	✓	✓	✓	✓	✓	✓
Scientific Research										
EXP-Bench (Kon et al., 2025)	2025	Research	Model	✓	✓	✓	✓	✓	✓	✓
PaperBench (Starace et al., 2025)	2025	Research	Model	✓	✓	✓	✓	✓	✓	✓
CORE-Bench (Siegel et al., 2024)	2024	Research	Rule	✓	✓	✓	✓	✓	✓	✓
Auto-Bench (Chen et al., 2025c)	2025	Research	Rule	✓	✓	✓	✓	✓	✓	✓
ResearchCodeBench (Hua et al., 2025)	2025	Research	Rule	✓	✓	✓	✓	✓	✓	✓
AstaBench (Bragg et al., 2025)	2025	Research	Rule+Model	✓	✓	✓	✓	✓	✓	✓
AlInsteinBench (Duston et al., 2025)	2025	Code+Research	Rule	✓	✓	✓	✓	✓	✓	✓
ResearchGym (Garikaparathi et al., 2026)	2026	Research+Code	Rule	✓	✓	✓	✓	✓	✓	✓
Mathematics & Knowledge										
MMLU (Hendrycks et al., 2020)	2020	Knowledge	Rule	✓	✓	✓	✓	✓	✓	✓
GPQA Diamond (Rein et al., 2023)	2023	Knowledge	Rule	✓	✓	✓	✓	✓	✓	✓
MMMU (Yue et al., 2023)	2023	Knowledge	Rule	✓	✓	✓	✓	✓	✓	✓

Continued on next page

Table 5 continued from previous page

Benchmark	Year	Domain	Eval Type	Prod.	M.M.	Under.	Div.E.	Exp.V.	Dyn.	Cross.
MathVista (Lu et al., 2024)	2024	Math	Rule	✗	✓	✗	✗	✗	✗	✗
FrontierMath (Glazer et al., 2024)	2024	Math	Rule	✗	✗	✗	✗	✓	✗	✗
AIME (MAA, 2025)	2025	Math	Rule	✗	✗	✗	✗	✗	✓	✗
HMMT (Harvard-MIT, 2025)	2025	Math	Rule	✗	✗	✗	✗	✗	✓	✗
USAMO (Petrov et al., 2025)	2025	Math	Human	✗	✗	✗	✗	✓	✓	✗
MMMLU (OpenAI, 2024)	2024	Knowledge	Rule	✗	✗	✗	✗	✗	✗	✓
Video-MME (Fu et al., 2024)	2024	Knowledge	Rule	✗	✓	✗	✗	✗	✗	✓
OpenAI-MRCR (OpenAI, 2025)	2025	Knowledge	Rule	✗	✗	✗	✗	✗	✗	✓
HLE (Phan et al., 2025)	2025	Knowledge	Rule	✗	✓	✗	✗	✓	✗	✓
ARC-AGI-2 (Chollet & ARC Prize Team, 2025)	2025	Reasoning	Rule	✗	✗	✗	✗	✗	✗	✗
OODBench (Lin et al., 2026)	2026	Knowledge	Rule	✗	✓	✗	✗	✗	✓	✗
Agent Product Evaluation										
xbench (xbench Team, 2025)	2025	Recruit+Mkt	Model	✓	✗	✗	✗	✓	✓	✗
AgentIF-OneDay (AgentIF Team, 2026)	2026	Daily Tasks	Model	✗	✓	○	✗	✓	✗	✓
Emerging Benchmarks (2026)										
Persona2Web (Kim et al., 2026a)	2026	Search	Model	✗	✗	✗	✗	✗	✗	✗
AmbiBench (Sun et al., 2026)	2026	OS	Model	✗	✓	✗	✗	✗	✗	✗
PAHF (Liang et al., 2026)	2026	Tool	Rule+Model	✗	✗	✗	✗	✗	✗	✗
AgenticShop (Kim et al., 2026b)	2026	Search	Model	✗	✗	✗	✗	✗	✗	✗
GAP Benchmark (Cartagena & Teixeira, 2026)	2026	Tool	Rule	✗	✗	✗	✗	✗	✗	✗
AgentLAB (Jiang et al., 2026)	2026	Safety	Model	✗	✗	✗	✗	✗	✗	✓
STING (Talokar et al., 2026)	2026	Safety	Model	✗	✗	✗	✗	✗	✗	✗
GT-HarmBench (Cobben et al., 2026)	2026	Safety	Rule	✗	✗	✗	✗	✗	✗	✗
ForesightSafety (Tong et al., 2026)	2026	Safety	Model	✗	✗	✗	✗	✗	✓	✓
APST (Broadwater, 2026)	2026	Safety	Model	✗	✗	✗	✗	✗	✗	✗
MemoryArena (He et al., 2026)	2026	Search+Research	Rule	✗	✗	✗	✗	✗	✗	✓
WebWorld-Bench (Xiao et al., 2026)	2026	Search+Code	Rule	✗	✗	✗	✗	✗	✗	✓
Gaia2 (Froger et al., 2026)	2026	OS+Search	Rule	✗	✗	✗	✗	✗	✓	✗
SkillsBench (Li et al., 2026b)	2026	Code+Tool	Rule	✗	✗	✗	✗	✗	✗	✓
MATEO (Roccabruna et al., 2026)	2026	Reasoning	Rule	✗	✓	✗	✗	✗	✗	✗
SciAgentGym (Shen et al., 2026)	2026	Research+Tool	Rule+Model	✗	✗	✗	✗	✗	✗	✓
Drug Scouting (Vinogradova et al., 2026)	2026	Search+Research	Model	✗	✗	✗	✗	✗	✗	✓
AD-Bench (Hu et al., 2026)	2026	Tool	Rule	✗	✗	✗	✗	✗	✗	✗
GUI-GENESIS (Cao et al., 2026)	2026	OS	Rule	✗	✓	✗	✗	✗	✓	✗
BookingArena (Logeswaran et al., 2026)	2026	Search	Rule+Model	✗	✗	✗	✗	✗	✗	✗
BrowseComp-V ³ (Zhang et al., 2026)	2026	Search	Rule	✗	✓	✗	✗	✓	✗	✗
Collective Behavior (Willis et al., 2026)	2026	Research	Rule	✗	✗	✗	✗	✗	✓	✗
Unsafer (Li et al., 2026c)	2026	Tool	Model	✗	✗	✗	✗	✗	✗	✗
Proxy State Eval (Chuang et al., 2026)	2026	Tool	Model	✗	✗	✗	✗	✗	✗	✗
AssetOpsBench (Patel et al., 2026)	2026	Asset Ops	Rule+Model	✓	✓	✓	✗	✓	✓	✗
AlphaEval (Ours)	2026	6 Domains	Rule+Model	✓	✓	✓	✓	✓	✓	✓

B Evaluation Infrastructure Details

The evaluation pipeline proceeds in six stages: (1) **Task Loading**: the framework scans the benchmark directory and constructs a task queue filterable by domain, difficulty, or evaluation type; (2) **Environment Provisioning**: Docker containers are instantiated with task input files; (3) **Agent Invocation**: the agent receives query.md and interacts through a standardized interface; (4) **Output Collection**: artifacts are collected from the results directory; (5) **Evaluation Dispatch**: outputs are routed to the appropriate evaluator; (6) **Result Aggregation**: results are aggregated into domain-level and benchmark-level summaries.

All task inputs, evaluation scripts, and reference answers are version-controlled. Docker images are pinned to specific versions for reproducibility. Each run generates structured logs with full interaction traces, raw outputs, and scores in JSON format.

C Scoring and Aggregation Methodology

Per-Task Scoring. Each task’s evaluation rubric script (rubric.py) outputs a standardized score $s \in [0, 1]$, which is scaled to $[0, 100]$ for reporting. For tasks composing multiple evaluation paradigms, the rubric script internally handles the composition—typically as a

weighted average of sub-evaluations, where weights are co-designed with domain experts during the task formalization stage. The final per-task score thus captures multi-dimensional quality in a single number:

$$s_{\text{task}} = \sum_{k=1}^K w_k \cdot e_k(a, t), \quad \text{where } \sum_{k=1}^K w_k = 1 \quad (1)$$

where $e_k(a, t) \in [0, 1]$ is the evaluation result from paradigm k for agent a on task t , and w_k is the expert-assigned weight for paradigm k .

Domain-Level Aggregation. Domain scores are computed as unweighted arithmetic means across all tasks within a domain:

$$S_{\text{domain}}(a) = \frac{1}{|T_d|} \sum_{t \in T_d} s_{\text{task}}(a, t) \times 100 \quad (2)$$

where T_d is the set of tasks in domain d .

Overall Aggregation. The overall benchmark score is the unweighted arithmetic mean across the six domain scores, giving each domain equal weight regardless of task count:

$$S_{\text{overall}}(a) = \frac{1}{6} \sum_{d=1}^6 S_{\text{domain}}(a) \quad (3)$$

This equal-weighting ensures that domains with fewer tasks (e.g., Human Resources with 11 tasks) receive the same influence as domains with more tasks (e.g., Procurement & Operations with 23 tasks), reflecting the principle that production readiness requires competence across all occupational domains.

D Agent System Configuration

All agent systems are invoked via their respective command-line interfaces (CLIs) within Docker-sandboxed environments. Table 6 summarizes the configuration details.

Agent Product	Version	Interface	Trajectory
Claude Code	2.1.70	claude CLI	Stream JSON
Codex	0.80.0/0.111.0	codex CLI	JSON log
GitHub Copilot	1.0.10	copilot-cli	Structured log
Cursor	2026.03.11	cursor CLI	Agent log

Table 6: Pinned agent-system versions used in the reported runs. Codex version varies by model because of API compatibility.

Trajectory Recording. Each evaluation run produces an observable execution trajectory capturing: (1) tool calls and their arguments, (2) intermediate reasoning where exposed by the scaffold, (3) file read/write operations, and (4) final output artifacts. These trajectories enable post-hoc failure analysis—for instance, distinguishing search failure from incorrect use of retrieved evidence. Closed products do not expose every internal state, and product or retrieval-service updates can prevent bit-for-bit reruns; our claim is therefore auditability of recorded behavior and reproducibility of scoring against versioned rubrics, not deterministic replay. Complete trajectories are retained internally because partner data cannot all be released; the public repository provides the framework, templates, and representative examples.

Model Configuration. All models are evaluated using their default API configurations (temperature, max tokens, etc.) as provided by each scaffold. We do not modify model parameters, as the goal is to evaluate agent *products* as deployed, not to optimize individual model settings. This mirrors how production users interact with these systems.

E Evaluation Taxonomy Coverage

Paradigm	Leaf-node Type	Cov.	#Tasks	Domains
Reference Verif.	Fuzzy matching	✓	42	HR, F&I, P&O
	Exact matching	✓	33	HR, F&I, P&O
	Word-embed. sim.	—	0	—
	LLM semantic	✓	49	F&I, H&LS, TR
Test Case Verif.	Code unit testing	✓	20	P&O
	Repo integ. test	—	0	—
	Env. state verif.	✓	13	F&I, SE
Formal Verif.	Math proof	✓	25	H&LS, F&I, P&O
	Code/logic	✓	2	H&LS
Rubric Assess.	Human-authored	✓	49	F&I, H&LS, TR
	Auto-generated	—	0	—
	Human-AI collab.	—	0	—
Competition	Ranking	—	0	—

Table 7: Fine-grained evaluation type coverage across the paradigms implemented by *AlphaEval*.

F Benchmark Statistics

Statistic	Value
Total tasks	94
Domains covered	6
Partner companies	7
<i>Input modality distribution</i>	
PDF-primary	~42% (39 tasks)
Excel/CSV-primary	~21% (20 tasks)
Markdown/Text	~25% (24 tasks)
Code/YAML	~12% (11 tasks)
Avg. agent timeout	857 seconds
Avg. max interaction turns	107
<i>Observed on Claude Code + Opus 4.6</i>	
Avg. execution time	14 minutes
Avg. interaction turns	46

Table 8: Benchmark statistics. Time and observed-turn statistics are measured on Claude Code + Opus 4.6; timeout and maximum-turn rows summarize task limits.

G Multi-label Evaluation Composition

Domain	#Tasks	Avg.	Types Composed
Human Resources	11	2.0	Fuzzy + Exact match
Finance & Investment	22	2.5	Rubric + LLM semantic + Structural verif. + Match
Procurement & Operations	23	3.8	Fuzzy + Exact + Unit test + Math + LLM
Software Engineering	11	2.5	Env. state verif. + Functional verif. + LLM
Healthcare & Life Sci.	16	2.8	Rubric + LLM + Math + Code + Match
Technology Research	11	2.5	Rubric + LLM semantic + Factual verif.
Overall	94	2.8	8 types, 100% ≥ 2

Table 9: Multi-label evaluation composition by domain. Each domain composes multiple evaluation types within individual task rubrics.

H Economic Value Estimation Methodology

We annotate each task with an assumption-dependent labor-value proxy through a two-stage pipeline: automated time/cost estimation followed by domain-expert calibration. It contextualizes task scale and must not be interpreted as realized economic value or the cost of replacing a worker.

H.1 Stage 1: AI Estimation

For each task, an LLM analyzes the task specification (`query.md`) and produces: (1) a list of required professional roles, (2) estimated hours per role, and (3) a complexity rating. The core estimation principle is: “estimate the time a qualified human professional would need to complete *this specific task*”—not the cost of building a replacement system.

Wage Data Sources. U.S. wages are queried from the Bureau of Labor Statistics (BLS) Occupational Employment and Wage Statistics via SOC codes. Chinese wage inputs are compiled from heterogeneous public Beijing salary pages and reviewed during domain calibration; unlike the BLS series, they are market estimates rather than an official uniform source. Each role maps to a standardized SOC code (e.g., Software Developer $\rightarrow$ 15-1252, Recruiter $\rightarrow$ 13-1071, Data Analyst $\rightarrow$ 15-2051).

Cost Calculation.

$$\text{Value} = \text{Hourly Rate} \times \text{Benefit Multiplier} \times \text{Hours} \quad (4)$$

where the benefit multiplier is $1.3\times$ for U.S. (healthcare, retirement, PTO), following BLS Employer Costs for Employee Compensation data and the methodology of Yang et al. (2026). We use $1.45\times$ for China as an explicit modeling assumption covering statutory contributions and annualized bonuses, not as an externally validated constant. Hourly rates use the 25th–75th percentile range to produce value intervals.

H.2 Stage 2: Expert Calibration

Domain practitioners review AI estimates and apply correction methods tailored to each domain:

- **Individualized assessment** (Software Engineering, factor 0.38): Experts evaluated each task considering code reuse across tasks, reducing total hours from 2,458 to 946. The largest reduction (78%) was for the task marketplace app where transaction components were reusable.

- **Uniform scaling** (Finance & Investment, factor 0.50): Expert judged AI systematically overestimated analytical tasks; all hours halved.
- **Quantitative standardization** (Human Resources, factor 0.64): Expert provided an empirical rate (20 resumes = 1 hour) to recalculate screening tasks.
- **Fixed-rate override** (Procurement & Operations): Expert specified a domain hourly rate (¥213/hr). This changes the cost valuation rather than task hours and is therefore not directly comparable with the time-scaling factors.
- **Selective adjustment** (Finance & Investment, factor 1.04): Expert increased two complex tasks to 150% of baseline; others unchanged.
- **Module re-estimation** (Healthcare & Life Sciences, factor 1.54): Expert reorganized 10 tasks into 3 functional modules, *increasing* hours from 86 to 132, reflecting AI’s underestimation of clinical protocol complexity.
- **No adjustment** (Finance & Investment, factor 1.00): Expert confirmed AI estimates as reasonable.
- **Uniform scaling** (Technology Research, approximately 0.80): Expert judged AI estimates slightly high; task-level adjusted values sum to 242 hours after rounding, from an unadjusted total of 298.

Key Observations. All six domains have completed expert calibration. Time-scaling adjustments vary in magnitude and direction: AI overestimates some routine tasks (notably software engineering with reusable components) but underestimates some domain-specialized tasks (clinical research). The procurement rate override is a different kind of adjustment and is not included in this comparison.

H.3 Sensitivity Analysis

To assess the robustness of the economic value estimates, we conduct a sensitivity analysis varying the three key parameters: benefit multipliers, hourly rate percentiles, and expert correction factors.

Benefit Multiplier Sensitivity. Varying the U.S. multiplier from 1.2 to 1.4 (baseline: 1.3) and the China multiplier from 1.3 to 1.6 (baseline: 1.45) produces total value ranges of \$141K–\$253K (USD) and ¥358K–¥627K (CNY), representing $\pm 9\%$ variation from the baseline estimates. The estimates are most sensitive to the China multiplier due to the larger proportion of China-based tasks.

Correction Factor Sensitivity. To test the impact of expert calibration, we compute valuations under three scenarios: (1) AI-only estimation (no expert correction), (2) baseline expert-calibrated values, and (3) uniform 20% increase in all correction factors. Results: AI-only yields 3,240 hours (\$217K–\$326K), baseline yields 2,420 hours (\$154K–\$231K), and the inflated scenario yields 2,904 hours (\$185K–\$277K). Expert calibration reduces the AI estimate by 25%, primarily driven by the Software Engineering domain where AI overestimated development hours by $2.6\times$.

Hourly Rate Sensitivity. Shifting from the 25th–75th percentile range to the 10th–90th percentile range widens the value interval from \$154K–\$231K (width \$77K) to \$128K–\$289K (width \$161K), approximately 109% wider. The median estimate remains approximately \$192K under this scenario.

These scenarios show that absolute dollar values are sensitive to wage, multiplier, and calibration assumptions. We therefore use the estimates only to contextualize task scale and do not treat them as evidence of realized economic value or worker replacement.

I Detailed Error Analysis

We qualitatively inspect failed task runs across configurations and domains, categorizing recurring patterns by their cognitive root causes rather than surface symptoms. We exclude external failures such as network errors and API quota exhaustion. The categories below organize illustrative cases; without a preregistered sampling and coding protocol, they should not be interpreted as prevalence estimates.

I.1 Information Retrieval Cognitive Failures

We identify five failure modes from Technology Research tasks requiring agents to produce comprehensive reports on recent AI developments.

Mode A: Factual Hallucination via Stale Training Data. When search tools fail to retrieve current information, models can substitute outdated training data *without acknowledging uncertainty*. Table 10 shows a representative case: Claude Opus 4.6 reports Harvey AI’s Series D funding figures when the task asks about the recent F-round, with every data point incorrect by a large margin. The same error appears under both Claude Code and Cursor, a pattern consistent with a model-level contribution rather than a single-scaffold artifact. The model produces no hedging language (e.g., “this information may be outdated”), presenting stale data with full confidence.

Dimension	Ground Truth	Model Output	Error
Funding round	F-round	Series D	Wrong round
Amount	\$160M	~\$300M	1.9×
Valuation	\$8B	~\$3B	2.7× off
Lawyer users	74,000	Not mentioned	Missing

Table 10: Factual hallucination example: Claude Opus 4.6 on Harvey AI funding data. The model substitutes outdated Series D data for the requested F-round.

Mode B: Imprecise Retrieval. Models correctly identify the research *area* a question targets and conduct extensive searches, but may not locate the *specific* papers required by the rubric. For example, on a task about Tool-Integrated Reasoning (TIR), none of the examined outputs found all three required papers (ToolRL, ToRL, ZeroTIR—all from early 2025). The best-performing agent (Cursor + Opus) produced a 16KB TIR survey that demonstrated genuine understanding of the field, but substituted related papers (ReTool, THOR, Tool-Zero) for the specific ones the rubric expected. Models also create their own conceptual frameworks rather than using original terminology—e.g., inventing “precision ceiling” and “generalization ceiling” instead of the papers’ actual terms “empirical support set” and “feasible support set.”

Mode C: Rigid Search Strategy. GPT-5.2 exhibits variable search persistence: some runs make extensive searches, whereas others declare “tool limitations” and output placeholder templates. Under the same Docker setup, Claude Opus 4.6 more consistently cycles through the cutoff-bounded interfaces to arXiv, Semantic Scholar, and OpenAlex before writing. This behavior is also scaffold-associated: GPT-5.2 via Codex scores 0.4 on a task where it scores 0.0 via Claude Code, consistent with search persistence being elicitable rather than a fixed model property.

Mode D: Attribution Confusion. Models find correct papers but extract incorrect meta-data by mixing information across search results. Claude Opus 4.6 correctly identifies the ToRL paper but attributes it to “Tsinghua University & ByteDance” (likely from a concurrent paper) instead of the correct “Shanghai Jiao Tong University,” and reports an accuracy of 43.3% instead of 62.1%. GLM-5 *inverts* the core finding of a paper, writing that SFT preserves old capabilities and RL enables new tasks when the opposite is true.

Mode E: Positive-Information Bias. The examined configurations systematically miss negative industry events. In one task requiring coverage of AI-agent startup failures, none reported Robin AI’s distressed sale (Series C failure, \$10M revenue against \$14M losses). Their queries and returned results were dominated by successful funding announcements, and they did not proactively search for startup failures or bankruptcies. The same outputs also missed niche success stories (e.g., Gamma achieving \$100M ARR with only 50 employees) outside the mainstream “agent startup” narrative.

Model Capability Profiles. Table 11 summarizes the distinctive cognitive profiles of each model, revealing that models have complementary rather than uniformly ordered capabilities.

Model	Search	Halluc.	Signature Pattern
Opus 4.6	30–50	High	Fills gaps with stale data confidently
GPT-5.2	5–75	Low	Variable persistence, sometimes outputs templates
Gemini 3 Pro	10–20	Low	Conservative, acknowledges uncertainty
GLM-5	5–10	Med	Conceptual inversion errors
MiniMax M2.5	3–5	Med	Correct direction, wrong details
Kimi K2.5	5–15	Med	Moderate persistence, domain-specific strengths

Table 11: Qualitative model profiles observed on technology-research tasks. Search ranges summarize examined logs rather than population estimates.

I.2 Cross-Domain Cognitive Failures

Beyond information retrieval, we identify cognitive failure modes across multiple domains that are poorly captured by coding-centric benchmarks because they require domain knowledge, subjective judgment, cross-document reasoning, and constraint satisfaction under ambiguity.

I.2.1 Human Resources: Subjective Judgment Failures

Human Resources tasks require agents to evaluate resumes against multi-dimensional hiring criteria, exposing subjective-assessment failures outside the scope of most coding benchmarks.

Criterion Explicitness Dependence. Agent performance inversely correlates with criterion subjectivity. Tasks with quantifiable requirements (“5+ years Python experience,” “AWS certification required”) achieve scores 2–3× higher than tasks requiring holistic judgment (“strong leadership potential,” “culture fit”). This suggests models have not learned to *infer* soft-skill indicators from indirect evidence—e.g., inferring leadership from descriptions of project scope, team size, and outcome ownership.

Inconsistent Criterion Weighting. When rubrics specify multiple evaluation dimensions without explicit weights, agents apply inconsistent priority orderings across candidates. The same agent may prioritize “technical depth” for one candidate and “breadth of experience” for another within the same evaluation batch, producing rankings that a human recruiter would flag as internally inconsistent.

Cross-Candidate Calibration Failure. Agents evaluate each resume independently rather than comparatively. When a strong candidate appears early in a batch, subsequent candidates receive inflated scores because the agent has no persistent calibration anchor. This “memoryless evaluation” pattern means that ranking quality degrades with batch size.

1.2.2 Healthcare & Life Sciences: Protocol Reasoning Failures

Healthcare & Life Sciences tasks require precise adherence to study protocols with cascading temporal dependencies, exposing failures in structured reasoning that superficially resemble—but fundamentally differ from—coding logic errors.

State Machine Precondition Semantics. Clinical protocols define visit windows, dose modifications, and adverse event escalation as state machines with preconditions. Agents correctly implement individual state transitions but fail on precondition chains—e.g., correctly computing that a dose reduction is needed but not recognizing that the reduction requires a preceding safety assessment that itself has a 48-hour observation window.

Protocol Terminology Confusion. Clinical research distinguishes between “Protocol Deviation” (minor, reportable) and “Protocol Violation” (major, may require participant withdrawal). Models conflate these terms or apply incorrect severity classifications, with downstream effects on recommended actions. Such domain-specific semantic distinctions are poorly captured by the general-purpose benchmarks considered here.

Temporal Escalation Logic. Multi-visit study protocols specify escalating responses to repeated events: first occurrence triggers documentation, second triggers dose modification, third triggers study withdrawal. Models correctly handle individual events but fail to maintain *cross-visit state*—treating each visit as independent and never triggering the escalation sequence.

1.2.3 Finance & Investment: Cross-Section Coherence Failures

Investment analysis tasks require producing multi-section reports (market sizing, competitive analysis, risk assessment, financial projections) that must be internally consistent, exposing a failure mode unique to long-form analytical writing.

TAM-SAM-SOM Logical Inconsistency. Agents produce market sizing sections where the Total Addressable Market (TAM), Serviceable Addressable Market (SAM), and Serviceable Obtainable Market (SOM) contain contradictory figures—e.g., a TAM of \$50B in the executive summary but \$80B in the market analysis, or a SOM that exceeds the SAM. Each paragraph is locally plausible; the inconsistency only surfaces when reading the full document.

Context Collapse: Generic vs. Company-Specific Analysis. Models produce analysis that reads as a *sector* report rather than a *company* report—discussing general industry trends without grounding them in the specific company’s financials, competitive position, or strategic choices. The most common manifestation: risk factors that are industry-generic (“regulatory uncertainty,” “competitive pressure”) without explaining how they specifically affect *this* company’s business model.

Hallucinated Market Data. When specific market data is not provided in the input documents, models fabricate plausible-sounding statistics (“the global SaaS market is projected to reach \$X billion by 2027”) without citing sources. Unlike information retrieval hallucination (Mode A above), this occurs even when the model is not explicitly asked to search—it fills analytical gaps with invented quantitative claims.

1.2.4 Procurement & Operations: Optimization and Constraint Failures

Procurement tasks require multi-attribute optimization under technical constraints, exposing failures in mathematical reasoning and constraint satisfaction that coding benchmarks test only superficially.

Synergy Blindness in Multi-Attribute Optimization. When optimizing across multiple correlated attributes (cost, performance, power consumption, thermal rating), agents sometimes optimize attributes independently rather than jointly, producing higher-cost solutions

than the jointly optimal reference because they fail to exploit component synergies (e.g., a slightly more expensive component that reduces cooling requirements and total system cost).

Constraint Specification Misinterpretation. Technical specifications use domain-specific notation that models misread: $\pm 10V$ tolerance interpreted as $\pm 5V$, “Type II” classification confused with “Class II,” or “continuous rating” conflated with “peak rating.” These are not hallucinations—the model reads the specification but applies incorrect domain knowledge to interpret it.

Infeasibility Recognition Bias. When a procurement problem has no feasible solution because of conflicting constraints, examined agents often fabricate a “best effort” solution rather than declaring infeasibility. This completion bias silently relaxes constraints instead of reporting that the requirements cannot be simultaneously satisfied, a risky behavior when outputs may be acted upon.

1.2.5 Finance & Investment: Grounded Critique Failures

Pitch coaching tasks require agents to provide actionable, data-grounded feedback on startup pitches, exposing the difference between structurally correct and substantively useful output.

Template Critique vs. Grounded Analysis. All models produce feedback that follows the correct structure (strengths, weaknesses, recommendations) but fails to reference the specific data provided. A critique stating “your market size claims need better support” is generic; a useful critique states “your Slide 7 claims a \$2B TAM but your bottom-up calculation on Slide 12 implies \$800M—reconcile these.” Models consistently produce the former. This failure is invisible to any benchmark that evaluates structural correctness.

User Category Confusion. When pitch decks present multiple user metrics (total registered users, monthly active users, core power users), models confuse categories—citing “500K users” without specifying which category, or conflating growth rates across categories. This produces feedback that sounds data-informed but references the wrong numbers.

Implicit Question Extraction. Meeting transcripts contain implicit investor concerns (e.g., a question about “team background” that is really probing founder-market fit). No model reliably identifies these implicit questions or addresses the underlying concern rather than the surface question.

1.2.6 Finance & Investment: Scale and Mapping Failures

Beyond information retrieval (Section I), Finance & Investment tasks requiring data synthesis from multiple documents expose additional cognitive failures.

Numerical Data Mapping Errors. When synthesizing data from tables across multiple documents, agents can extract correct numbers but assign them to incorrect columns or rows. For example, an output may correctly read that Company A has revenue of \$50M and Company B has revenue of \$30M, but swap them in the final table. Such mapping errors appear more frequently in the examined tasks with larger numbers of data points, though our qualitative sample does not support a precise prevalence estimate.

Authority Hierarchy Misunderstanding. When documents contain conflicting information (e.g., a press release states one figure while an SEC filing states another), models do not consistently prioritize the more authoritative source. In production, domain experts have clear authority hierarchies (regulatory filings > press releases > news articles); models treat all sources as equally credible.

Selective Document Reading. When given a large document set, agents read documents in the order provided and exhibit recency bias—information from later-read documents is weighted more heavily. If critical information appears in an early document that the agent scans quickly, it may be omitted from the final output even though the agent demonstrably “saw” it (evidenced by interaction logs).

1.2.7 Software Engineering: Specification-Implementation Gap

Software Engineering tasks expose a distinctive failure mode where agents produce functional code that satisfies superficial requirements but misses deeper specification intent. For example, agents implement correct navigation structure but omit specified interaction details (e.g., swipe gestures, animation timing). The gap between “code that compiles and runs” and “code that matches the product specification” mirrors the broader production challenge: research benchmarks test functional correctness, while production demands specification fidelity.

J Representative Task Examples

We present one representative task from each of the six O*NET domains to illustrate the production complexity *AlphaEval* captures.

Human Resources: Resume Screening (hunter-ai-1). The agent is given a job description for an AI Scientist Mapping Intern role together with department-specific hiring criteria (e.g., minimum internship duration, university tier, prior AI experience) and must screen 24 candidate resumes provided as PDF and JPEG files. The agent must select exactly six candidates to advance to a first-round interview and output the chosen candidate IDs as a Python list in a designated answer file. Evaluation compares the agent’s selections against the ground-truth shortlist using precision, recall, and F1 score, with a passing threshold of $F1 \geq 0.6$.

```
name: "AI Resume Screening - Mapping Dept"
category: hr
difficulty: hard
evaluation:
  type: code_exec
  rubric_script: .eval/rubric.py
agent:
  timeout_seconds: 600
  max_turns: 100
```

Finance & Investment: Segment Research Report (segment-research-1). The agent acts as a senior industry research analyst and must produce an investment-grade Segment Research report from a startup’s business plan PDF. The report must follow a prescribed eight-section template covering industry definition, TAM–SAM–SOM market sizing (with both top-down and bottom-up methods), product landscape, customer typology, key success factors with quantitative thresholds, three to four competitive case studies with GTM strategy analysis, and a technology deep-dive including patent barriers and commercialization feasibility. A reference template is provided. Evaluation uses LLM-as-Judge to assess structural completeness, analytical depth, data accuracy, and adherence to the template format.

```
name: "Segment Research Report"
category: research
difficulty: hard
evaluation:
  type: code_exec
  rubric_script: .eval/rubric.py
agent:
  timeout_seconds: 1200
  max_turns: 150
```

Procurement & Operations: BOM Cost Optimization (yuhe-1). The agent receives a catalog of 2,000 board cards with detailed specifications and pricing in an Excel file, a field-structure CSV defining column semantics, and a natural-language procurement requirements document specifying functional constraints. It must select an optimal combination of board cards that satisfies all functional requirements while minimizing total procurement cost, breaking ties by fewest cards and then by ascending card ID. The output is a structured cost breakdown listing each selected card's ID, quantity, unit price, and subtotal. Evaluation uses a rubric script that verifies constraint satisfaction and compares total cost against a known optimal solution.

```
name: "BOM Board Card Procurement Optimization"
category: optimization
difficulty: hard
evaluation:
  type: code_exec
  rubric_script: .eval/rubric.py
agent:
  timeout_seconds: 600
  max_turns: 100
```

Software Engineering: Poetry Mini-Program (miniprogram_poetry). The agent must implement a complete WeChat Mini-Program for classical Chinese poetry appreciation, following a detailed 200-line product requirements document that specifies page layout (gallery-style hand-drawn illustration aesthetic), navigation architecture (four-tab bottom bar), core features (AI text-to-speech recitation with speed and voice controls, waterfall-style poem browsing, a community forum for user posts, and a personal collection manager), and interaction patterns (modal dialogs for content creation, clipboard-based sharing). The deliverable is a fully functional codebase. Evaluation combines automated UI testing to verify page navigation and component rendering with rubric-based checks on code structure and feature completeness.

```
name: "Poetry Appreciation Mini-Program"
category: miniprogram
difficulty: hard
evaluation:
  type: miniprogram
  rubrics: [10 weighted UI test points]
agent:
  timeout_seconds: 1800
  max_turns: 100
```

Healthcare & Life Sciences: eCRF Visit Window Calculation (linchuang-1). The agent operates within a simulated eCRF (electronic Case Report Form) system for a Phase III non-small-cell lung cancer clinical trial. Given a JSON-based visit-window rule configuration and a patient's screening visit date, the agent must compute target dates and permissible windows for subsequent treatment visits, determine which visits can be calculated given available data, and perform cascade-impact analysis when an actual visit date deviates from the target. The output must show explicit calculation formulas and all dates in YYYY-MM-DD format. Evaluation uses a rubric script that checks date arithmetic correctness, proper application of window offsets, and accurate cascade reasoning.

```
name: "eCRF Visit Window Calculation"
scenario: linchuang
evaluation:
  type: code_exec
  rubric_script: .eval/rubric.py
files:
  - files/JSON_eCRF_CORE.md
  - files/CRF_visit_window_spec.pdf
```

Technology Research: AI Agent Startup Landscape (jiqizhixin-1). The agent acts as a professional AI industry analyst and must produce a comprehensive research report on the 2025 landscape of AI agent startups. The report must cover four dimensions: agent startups that raised over \$100M (with vertical categories such as legal, coding, and search), startups that failed or were acquired along with root-cause analysis, large-tech acquisition activity targeting agent companies, and shifts in Anthropic’s Claude adoption within the YC ecosystem. All claims must be supported by specific figures such as funding amounts and market sizes, with references to reliable sources. Evaluation uses LLM-as-Judge to assess factual accuracy, analytical depth, data substantiation, and structural completeness.

```
name: "AI Agent Startup 2025 Landscape"
category: research
difficulty: hard
evaluation:
  type: code_exec
  rubric_script: .eval/rubric.py
agent:
  timeout_seconds: 1200
  max_turns: 150
```

K Statistical Reliability of Reported Scores

We quantify uncertainty from task sampling using a domain-stratified non-parametric bootstrap ($B = 10,000$) over the 94 observed per-task scores. Within each replicate, tasks are sampled with replacement inside each domain; the overall score is the unweighted mean of the six resampled domain means, and 95% intervals are percentile intervals. We illustrate the analysis for Claude Code + Opus 4.6. We do not claim repeated whole-benchmark runs.

Domain	Mean	95% CI	n
Human Resources	35.91	[20.91, 50.36]	11
Finance & Investment	70.35	[56.08, 82.75]	22
Procurement & Ops.	83.35	[66.70, 96.39]	23
Software Engineering	70.95	[51.33, 88.45]	11
Healthcare & Life Sci.	50.06	[33.50, 67.19]	16
Technology Research	75.82	[70.45, 82.27]	11
Overall (6 domains)	64.41	[58.25, 70.45]	94

Table 12: Task-level bootstrap 95% confidence intervals for Claude Code + Opus 4.6. The overall mean exactly matches Table 3.

These intervals express uncertainty conditional on the observed benchmark tasks and show why small ranking gaps should not be over-interpreted. The Finance & Investment comparison in the main text uses a paired task bootstrap and a two-sided Monte Carlo paired-permutation test with 10,000 sign flips. Because related task variants (including BOM and eCRF families) are resampled as individual tasks, these intervals do not account for company- or task-family-level clustering and may be narrower than a cluster-aware analysis.

L Generalist-Human Baseline

CS PhD students completed all 94 tasks, distributed one student per task, at a normal working pace and without the agents’ timeouts. The original task rubrics scored their first attempts. Because these participants were not the partner experts who authored the criteria, this comparison avoids using the rubric authors as their own baseline, but it should be interpreted only as a *generalist-human* anchor.

The aggregate masks strong heterogeneity: humans lead on Software Engineering and Healthcare & Life Sciences and are near parity on Human Resources, whereas the agent leads

	HR	F&I	P&O	SE	H&LS	TR	Avg.
Claude Code + Opus 4.6	35.91	70.35	83.35	70.95	50.06	75.82	64.41
Generalist human (CS PhD)	39.00	44.31	30.48	78.27	58.75	68.18	53.17

Table 13: Full-benchmark generalist-human baseline under the same task rubrics. This is not a domain-expert or incumbent-workflow baseline.

most on programmatic Procurement & Operations. Neither row establishes deployment readiness; partner acceptance depends on domain-specific criteria and workflow context.

M Judge-Bias Sanity Check

We sampled 50 rubric points from three LLM-judged domains and compared each judge with a human expert. “Opus differential” is the judge’s excess coverage relative to the human on Opus-produced answers minus the corresponding excess on non-Opus answers.

Judge	Agreement	Cohen’s κ	Opus differential
Claude Opus 4.6	84%	.58	+16 pp
DeepSeek-V3.2	82%	.52	+28 pp
o3	78%	.47	+28 pp
Gemini-2.5-Pro	78%	.41	+20 pp
GPT-4o	76%	.34	+24 pp
GPT-5.2	74%	.40	+12 pp

Table 14: Small-sample judge-bias sanity check. The human expert covered 80% of rubric points for Opus answers and 68% for non-Opus answers.

The deployed Opus judge is the most human-aligned in this sample, and its differential is not larger than four of the five independent judges shown. This is convergent evidence against a large self-preference effect in the sample, not a definitive test.

N Practitioner Survey: AI Product Deployment Challenges

To ground *AlphaEval* in real practitioner needs, we conducted a mixed-methods survey of 27 AI product companies affiliated with a startup accelerator (response rate: 54% of 50 invited). The survey combined structured questionnaires (16 items) with open-ended text analysis, conducted from December 2025 to January 2026.

Respondent Profile. Companies span four development stages: concept verification (14.8%), internal pilot (14.8%), early commercialization (48.1%), and scaled deployment (22.2%). The majority target enterprise clients (59.3%), with consumer-facing products at 29.6%, with the remainder targeting internal employees or niche verticals. Most companies support text input (85.2%), with growing adoption of image (55.6%), structured data (48.1%), audio/video (44.4%), compound documents (40.7%), and code (37.0%).

Evaluation Infrastructure Gap. Current evaluation practices reveal significant infrastructure deficits:

- 25.9% of companies have *no explicit evaluation criteria*
- 33.3% rely solely on small sets of “golden samples” with manual inspection
- Only 11.1% have established structured test datasets with automated evaluation pipelines
- 22.2% have closed-loop A/B testing feedback (primarily among scaled companies)

Core Technical Challenges. The most frequently cited challenges (multiple selection, N=27):

- Output instability / inconsistency: 59.3%
- Instruction-following failures in complex scenarios: 51.9%
- Hallucination / factual errors: 40.7%
- Low inference efficiency / high token costs: 40.7%
- Long-context processing / memory loss: 33.3%
- Multi-turn dialogue degradation: 22.2%
- Safety / compliance risks: 14.8%

Confidence in Model Updates. When asked “After each model update or prompt modification, are you confident the new version is better?”:

- 25.9% reported high confidence (with established evaluation systems)
- **63.0%** reported low confidence (lacking reliable evaluation mechanisms)
- 7.4% reported no confidence at all
- 3.7% reported the question was not applicable (not yet at frequent update stage)

Testing Resource Allocation. 70.4% of companies rely on *developers performing testing as a side task*, consuming development time. Only 18.5% have dedicated testing personnel with substantial human review investment. 11.1% depend entirely on user feedback post-deployment.

Top Evaluation Needs. Open-ended text analysis (keyword extraction and co-occurrence network) identified three demand themes:

1. **Automated evaluation platform construction** (weight: 0.47): automated problem localization, iteration recommendations, and priority ranking
2. **Objective evaluation standards** (weight: 0.24): reliable product quality verification and performance benchmarking
3. **Cost and efficiency optimization** (weight: 0.18): reducing testing costs, improving inference efficiency, and token cost control

Survey Instrument. The questionnaire comprised 16 items covering product information (Q1–Q5), technical status (Q6–Q11), and evaluation needs (Q12–Q16). Key questions included: development stage (single choice), target customer type (single choice), input modalities supported (multiple choice), evaluation infrastructure maturity (single choice, 5-level scale from “no criteria” to “closed-loop A/B testing”), core technical challenges (multiple choice, 7 categories), confidence in model updates (single choice, 4-level scale), testing resource allocation (single choice, 3-level scale), preferred evaluation methodologies (multiple choice), required execution environments (multiple choice), and willingness to share anonymized test data.

These findings directly motivated the design of *AlphaEval*: the 63% low-confidence rate in model updates underscores the need for reliable automated evaluation, the diversity of input modalities (42% PDF, 21% structured data) informed our task composition, and the demand for automated evaluation platforms (weight 0.47) validates our requirement-to-benchmark construction framework.