ByteDance | Seed

StartupBench: Benchmarking General-Purpose Agents on Market-Validated Workflows

[1]ByteDance Seed, [2]Peking University, [3]M-A-P, [4]TokenWave.AI

Full author list in Contributions

Abstract

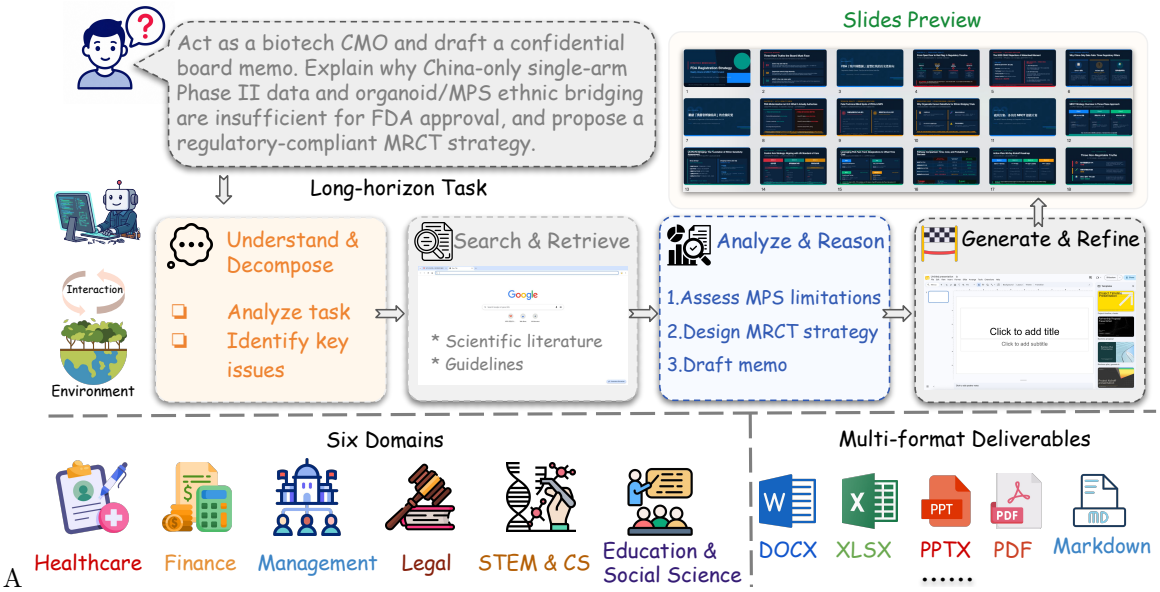
Recent advances in models and agent frameworks have significantly improved the ability of AI systems to perform complex tasks. However, existing benchmarks largely evaluate capabilities selected by researchers, leaving open a fundamental question: to what extent can current agents accomplish the end-to-end work that people actually wish to delegate to AI? We argue that successful startup-agent products provide a unique lens into this question, as they represent market-validated services built around real user demand. To study this gap, we introduce **StartupBench**, a benchmark of end-to-end workflows derived from extensive analysis of startup-agent products, product demonstrations, and user interviews across diverse professional domains. The resulting tasks capture real-world work that users actively adopt and pay for AI systems to perform, ranging from research and analysis to content creation and business operations. We evaluate frontier general-purpose agents on StartupBench and find that many market-validated workflows remain challenging despite recent advances in foundation models and agent frameworks. Through detailed task and failure-mode analyses, we identify recurring limitations in long-horizon execution, implicit requirement understanding, workflow orchestration, and deliverable quality. Our results reveal a substantial gap between the work users seek to delegate and the capabilities of current general-purpose agents, highlighting important directions for future agent training, evaluation, and development.

Project Page: https://startupbench.github.io/
Date: August 14, 2026

1 Introduction

Large language models (LLMs) are increasingly evolving from systems that primarily retrieve information or answer questions[4, 11, 22] into agents capable of executing complex real-world tasks. Beyond assisting users through isolated interactions, current AI systems are expected to complete end-to-end(E2E) workflows, coordinate multiple capabilities, and produce deliverables that satisfy practical requirements under realistic constraints. As AI becomes integrated into professional work, the ability to autonomously complete user-delegated tasks has become an increasingly important indicator of model capability. Evaluating whether AI systems can reliably transform their underlying capabilities into professionally usable deliverables is therefore becoming a central challenge for the next generation of benchmarks.

Recent benchmarks have made important progress toward evaluating E2E task execution under increasingly realistic settings. Nevertheless, several important limitations remain. First, benchmark tasks are still largely defined from researchers' perspectives rather than grounded in workflows that have demonstrated practical demand through real-world AI adoption [11, 15, 16], limiting their ability to reflect authentic user needs



A

Figure 1 Overview of StartupBench: real AI-product workflows, long-horizon agent execution, six-domain coverage, and multi-format deliverable generation.

and deliverable requirements. Second, although several benchmarks evaluate complete deliverables [20, 21], their evaluation protocols are often substantially coarser than the deliverables themselves. Real-world work products typically involve numerous functional, structural, formatting, and domain-specific requirements that holistic evaluations cannot faithfully assess. Consequently, existing benchmarks still provide only limited evidence of *how well current models and agents can satisfy complex real-world user requirements and reliably produce professionally usable deliverables*.

To address these limitations, we introduce **StartupBench**, a survey- and interview-driven benchmark for evaluating whether models and general-purpose agents can satisfy complex real-world user requirements through E2E task completion. StartupBench is built around following design principles:

- **Real-world task sourcing.** Rather than designing tasks from curated scenarios, StartupBench derives benchmark tasks from AI-native startups whose products have demonstrated practical demand and economic value, ensuring that benchmark workflows reflect authentic user needs.
- **E2E deliverables.** Each task requires producing the complete deliverables that users ultimately expect in realistic workflows, rather than intermediate outputs or simplified demonstrations.
- **Fine-grained rubric-based evaluation.** Each StartupBench task is evaluated using multiple independent rubrics covering distinct dimensions of the expected deliverable, including functionality, structure, formatting, and domain-specific quality, ensuring a comprehensive and faithful assessment of real-world task completion.

We evaluate a broad range of leading foundation models and general-purpose agents on StartupBench under a unified agent harness. Results show that even the strongest model successfully completes fewer than 30% of benchmark tasks, indicating that current systems remain far from reliably producing professionally usable deliverables. Performance varies substantially across domains, with STEM & Computer Science and Education & Humanities proving particularly challenging. Trajectory analysis further attributes these failures primarily to limitations in complex instruction following and domain-specific expertise, highlighting promising directions for future foundation models and agent systems.

To summarize, our contributions are as follows:

- We propose a survey- and interview-driven methodology for constructing agent benchmarks from real

AI adoption scenarios.

- We construct StartupBench, a benchmark built through AI-native startup research, enterprise-user interviews, user-oriented task abstraction, expert data construction, and multi-stage quality control.
- We present a comprehensive empirical study of current foundation models and general-purpose agents on startup-derived workflows, providing a practical evaluation signal for assessing whether AI agents are progressing from answering questions toward reliably completing real work.

2 Related Work

2.1 Commercial AI agents and Vertical Workflows

Recent advances in foundation models have enabled AI agents to gradually acquire planning, tool-use, file-processing, and multi-step execution capabilities [18, 26], pushing AI systems from general conversational assistants toward workflow-oriented products. In practice, commercial AI products increasingly package these capabilities into vertical workflows such as document processing, data analysis, financial research, enterprise operations, and knowledge work. This trend suggests that real-world AI value is not determined only by isolated model capabilities, but also by whether these capabilities can be productized into useful work products. However, existing academic benchmarks rarely use commercial adoption itself as a source of task discovery. StartupBench addresses this gap by deriving tasks from market-validated AI-native startups and their deep users, moving task construction from researcher-defined settings toward demand-driven workflow construction.

2.2 End-to-end evaluation of AI agents

Agent benchmarks have evolved from capability evaluation toward end-to-end task completion. Early benchmarks mainly focus on tool use, multi-step reasoning, and general assistant abilities, as in AgentBench and GAIA [9, 11]. Later work extends evaluation to more realistic interactive environments, including web interaction and computer-use tasks [23, 30]. More recent benchmarks further move toward professional work and economically meaningful tasks, covering enterprise software, simulated workplaces, occupational tasks, and long-horizon workflows [3, 6, 10, 15, 24, 31]. These works reflect a clear transition from isolated capability tests to realistic task-completion evaluation. StartupBench follows this direction, but differs in its task source: rather than relying primarily on public environments, occupational taxonomies, or simulated workplaces, it derives tasks from AI-native startup products, ToB user needs, and expected business deliverables.

2.3 Agent-as-Judge for complex deliverable evaluation

As agent tasks shift from short answers to heterogeneous deliverables such as reports, spreadsheets, slides, code patches, and structured files, evaluation must move from capability-centric scoring to deliverable-centric assessment[27]. LLM-as-a-Judge has been widely used for open-ended response evaluation, with MT-Bench and Prometheus studying preference-based or rubric-conditioned judging [7, 29]. For agentic workflows, however, judging often requires evidence retrieval, file inspection, state verification, and constraint checking over final artifacts. Agent-as-a-Judge extends LLM judging by equipping evaluators with tools and environment interaction, while AJ-Bench further studies judge agents on information acquisition, state verification, and process verification [19, 32]. StartupBench adopts this deliverable-centric evaluation paradigm by combining expert rubrics with automated judge agents that inspect original artifacts and evidence views, evaluating whether agents can transform model capabilities into usable work products.

3 StartupBench

StartupBench is designed to evaluate whether models and agents can complete realistic workflows that users already delegate to AI products. This objective requires data that reflects authentic user demand while supporting controlled evaluation. We therefore require each task to be realistic, answerable, evaluable, and discriminative: it should originate from actual AI product usage, provide sufficient information for a valid solution, specify clear output requirements and assessment criteria, and remain challenging enough to distinguish current models and agents.

Table 1 Comparison of StartupBench with existing benchmarks. E2E denotes end-to-end workflow completion; Item-wise Eval. denotes criterion-level judging. ✓, ✗, and △indicate full, no, and partial support.

Benchmark	Task Source	Final Output	E2E	Item-wise Eval.	Process Logic	Open-ended	Evaluation Method
GAIA [11]	Manual	Text	△	✗	✗	✗	Rule
OSWorld [23]	Real+Manual	Env State	△	✗	✗	✗	Execution
DAComp [8]	Real+Manual	Workspace	✓	✗	✗	✓	Hybrid
GDPval [15]	Manual	Work products	✓	✗	✗	△	Manual
Workspace-Bench [21]	Sim.+Manual	Workspace	✓	✗	✗	✓	Agent-as-Judge
OneMillionBench [25]	Expert-authored	Text	△	✗	△	✓	LLM-judge
Agents' Last Exam [20]	Occupation-guided	Workspace	✓	✗	△	△	Hybrid
OfficeQA Pro [14]	Corpus-grounded	Text	✗	✗	✗	✗	Answer match
StartupBench (Ours)	**Market-vetted + Expert-built**	**Workspace**	✓	✓	✓	✓	**Agent-as-Judge**

Figure 2 StartupBench data construction pipeline. We select market-validated startup agents, collect real user workflows, construct task artifacts and instances, and calibrate quality and difficulty through multi-stage review.

Neither manual task design nor direct extraction from product demonstrations is sufficient to obtain such data. The former is controllable but may be detached from real demand, while the latter is realistic but often lacks context, success criteria, or reproducible specifications. We therefore adopt a survey- and interview-driven multi-stage pipeline shown in Figure 2. We first survey market-validated AI-native startups, interview deep users to identify usage contexts, task goals, and expected deliverables, recruit domain experts to convert validated scenarios into benchmark instances, and finally apply quality control for realism, answerability, evaluability, and discriminative difficulty.

3.1 Dataset Construction

3.1.1 Startup Survey for Market-Validated Agent Scenarios

We first survey AI-native startups that build agent products across diverse domains. To identify workflows with market validation, we retain startups with more than USD 1M in funding and require evidence of real adoption, either through paid usage or substantial user traction. Funding provides a coarse signal of market expectation, while payment or user-scale evidence indicates that the product is used beyond exploratory demonstrations. The output of this stage is a pool of candidate products and workflows for user interviews. During this stage we collect 20+ start-up agents as the basis for subsequent stage.

3.1.2 User Interviews for Scenario and Requirement Discovery

For each candidate Startup-product, we interview deep users of the corresponding agent from different domains to recover practical usage beyond public product descriptions. The interviews elicit the usage context, user goal, input information, expected deliverable, success criteria, and common constraints. We summarize these findings into scenario specifications and representative demo cases, which serve as anchors for later annotation stage. During this stage we interviewed over 30 deep users and collected at least one demo case for each candidate Agent.

3.1.3 Expert Construction of Domain Tasks

Given the interview-derived workflow specifications and representative user scenarios as a reference or seed task, we recruit domain experts to reconstruct benchmark tasks that preserve the original workflow objectives, practical constraints, and expected deliverables while adapting them into reproducible evaluation instances. Rather than designing synthetic problems, experts standardize authentic user requests into benchmark tasks that faithfully represent real professional workflows.

Each task is required to satisfy three principles: it must originate from a realistic work request that users would naturally delegate to AI agents, involve an end-to-end workflow rather than an isolated subtask, and produce concrete professional deliverables with clearly defined success criteria and fine-grained evaluation rubrics for objective evaluation.

To standardize these reconstructed workflows for benchmark evaluation, each task is represented as a triple

$$\mathcal{T} = (q, \mathcal{E}, \mathcal{R}),$$

where q is a natural-language user request describing the task, $\mathcal{E}$ denotes the workspace containing all input files and resources required to complete the task, and $\mathcal{R} = \{(p_i, w_i)\}_{i=1}^{n}$ is a set of weighted evaluation rubrics that assess complementary aspects of deliverable quality. Detailed task specifications and annotation guidelines are provided in Appendix A.1.

3.1.4 Quality Control

To ensure benchmark quality and consistency, every task undergoes the following multi-stage quality control process:

Expert Cross Validation. Every task is independently reviewed by at least one additional domain expert. Reviewers verify four aspects of task quality: (1) the authenticity and correctness of the task description and workspace, (2) the fidelity of the reconstructed workflow, (3) the validity of the evaluation rubrics, and (4) the consistency between the reference deliverable and the finalized evaluation protocol. Detailed review criteria are provided in Appendix A.2.

Difficulty Control. We calibrate task difficulty through pilot executions using several frontier models under the same evaluation harness adopted in the benchmark[1]. Tasks that are consistently completed with high quality are excluded due to limited discriminative value. Pilot results also serve as an additional quality assurance stage: when failures are attributed to ambiguous instructions, incomplete workspace information, or inadequately specified evaluation criteria rather than intrinsic task difficulty, the corresponding task is revised and re-validated before inclusion.

3.2 Statistics for StartupBench

StartupBench contains 97 real-world workflow tasks across 6 top-level domains: Medical & HealthCare, Finance, Legal, Business & Management, STEM & computer science, and education & humanities. These tasks are designed to reflect deliverable-oriented workflows in realistic office settings, requiring agents to understand task contexts, follow complex constraints, process heterogeneous files, and produce usable work products. For evaluation, each task is evaluated using an average of 25.3 fine-grained rubrics, enabling detailed

[1]We randomly select models from GPT-5.5, Seed2.1-pro and GLM-5.2 as the for for pilot executions.

Table 2 Statistics of StartupBench. The benchmark contains 97 real-world workflow tasks across six top-level domains and requires diverse deliverable formats.

Category	Count	Percentage (%)
Overall statistics		
Total tasks	97	100.0
Top-level domains	6	–
Domain distribution		
Medical&HealthCare	21	21.6
Finance	18	18.6
Legal	16	16.5
Business & Management	19	19.6
STEM & Computer Science	16	16.5
Education & Humanities	7	7.2

Output formats: DOCX, XLSX, PPTX, PDF, Markdown, images, and text-based deliverables.

assessment of complex, long-horizon workflows and diverse work deliverables. The summary of the domain coverage and main output formats of StartupBench is shown in Table 2.

3.3 Automatic Evaluation of StartupBench Tasks

As stated above, every task in StartupBench is defined as a triple $\mathcal{T} = (q, \mathcal{E}, \mathcal{R})$. Given $(q, \mathcal{E})$, the evaluated model produces a set of deliverables $\mathcal{D}$. Before evaluation, we construct a lightweight evidence view $\mathcal{V}(\mathcal{D})$, consisting of extracted textual content and rendered page images, to facilitate navigation over heterogeneous artifacts while allowing the judge to access the original files whenever necessary.

Rather than relying on a single holistic judgment, StartupBench evaluates each rubric item independently through a dedicated AgentJudge session. Each task typically contains 20+ fine-grained rubric items covering diverse aspects of the expected deliverables, including functionality, completeness, formatting, and domain-specific requirements. Evaluating these criteria separately allows each rubric to receive dedicated reasoning and tool usage without interference from unrelated evaluation dimensions, while ensuring that every requirement is assessed explicitly. The final task score is then obtained by aggregating the outcomes of all rubric-level evaluations.

Following **Agent-as-a-Judge**, each rubric evaluation is carried out by a lightweight agent rather than a single forward pass. Let J denote the judging prompt and evaluation protocol used by the AgentJudge, given the task specification, deliverables, evidence view, and the target rubric item p_i, the judge produces both a binary decision $y_i \in \{0, 1\}$ indicating whether the rubric is satisfied and a textual justification e_i:

$$(y_i, e_i) = \text{AgentJudge}\big(J, q, \mathcal{D}, \mathcal{V}(\mathcal{D}), p_i\big), \tag{1}$$

The final task score is computed by aggregating the weighted decisions across all rubric items:

$$\text{score}(\mathcal{D}, \mathcal{T}) = \frac{\sum_{i=1}^{n} w_i y_i}{\sum_{i=1}^{n} w_i} \in [0, 1]. \tag{2}$$

4 Experiments

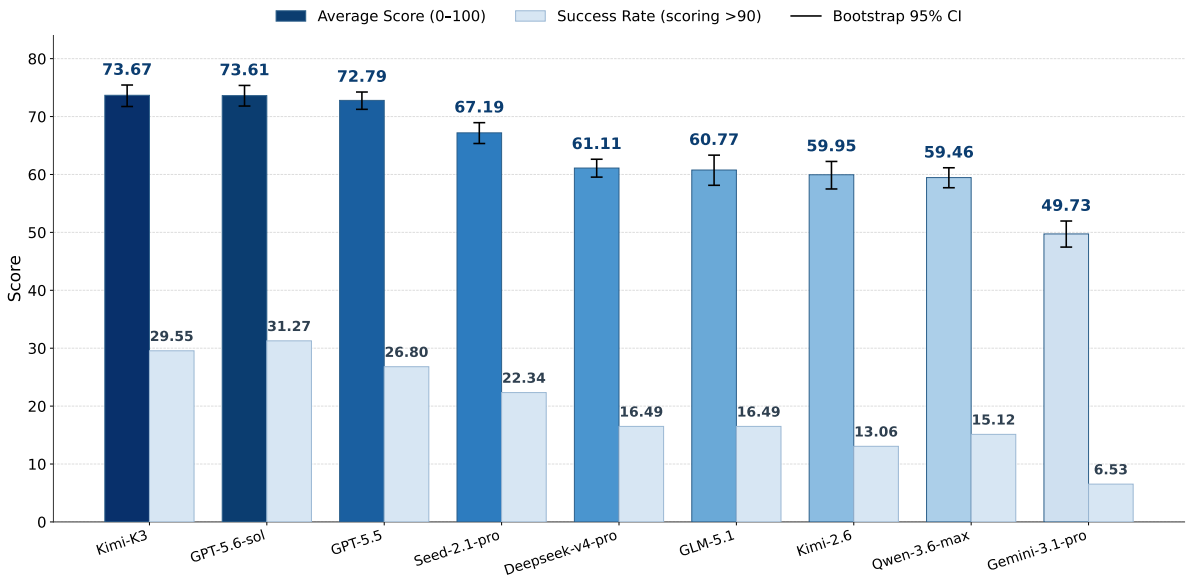


Figure 3 Leaderboard of StartupBench.

4.1 Experimental Setup

For every task, the agent is initialized in the task-specific workspace together with the corresponding input query. The end-to-end agent solves the task and then the generated deliverables under the designated output directory are collected as the task output. Following the evaluation protocol described in previous section, the output, together with the task specification, workspace context, and rubric set, is provided to the judge agent for automatic evaluation.

We evaluate 7 representative models spanning both closed- and open-source families:(i) **closed-source**: GPT-5.5 [13], Gemini-3.1-Pro [5], Seed-2.1-Pro [1],Qwen-3.6-Max [17];(ii) **open-source**: DeepSeek-V4-Pro [2], Kimi-K2.6 [12], GLM-5.1 [28]. All worker agents are executed under the same NANOBOT harness with an identical tool configuration. Each run is capped at a maximum of 200 interaction steps. For evaluation, the judge agent also runs under the same NANOBOT harness to ensure a consistent execution environment, while using GPT-5.5 as the underlying judge model. Apart from the continuous task score, we also report **pa ss rate**, the fraction of runs that are successfully completed, pooled over the three runs. A task is considered successful if its final score is at least 90; otherwise, it is counted as a failure. All reported metrics are averaged over three independent runs, with 95% bootstrap confidence intervals (10,000 resamples).

4.2 Main Results

Startup tasks remain far from saturated under a general-purpose agent harness. Although the strongest models, Kimi-k3 and GPT-5.6-sol, achieve average scores of 73.67% and 73.61%, respectively, no model successfully completes even one third of the benchmark under the strict acceptance criterion (score ≥ 90). More broadly, the average scores of most models lie between roughly 55 and 75, indicating that current agents are generally capable of making substantial progress toward solving realistic professional workflows. However, high average scores do not necessarily translate into high task completion rates. This is particularly evident for the Kimi series: Kimi-k3 achieves the highest average score overall but a lower pass rate than GPT-5.6-sol, while Kimi-K2.6 similarly attains a higher average score than Qwen-3.6-Max but a lower pass rate. These discrepancies suggest that some models can satisfy a substantial fraction of task requirements while still falling short of complete delivery. More generally, all models exhibit substantially lower pass rate than average score, suggesting that the primary bottleneck is no longer executing large portions of a workflow, but consistently producing artifacts that satisfy the standards required for direct professional use.

Table 3 Model performance of StartupBench, averaged over three independent runs. **Top block:** importance-weighted mean per-task score (0–100); **bottom block:** pass rate, the fraction of samples scoring ≥ 90 (%), pooled over the three runs. Per-column **best in bold**, second-best underlined.

Model	Overall	Med	Financial	Legal	Business	STEM	Education
Average score							
Kimi-k3	**73.67**	**80.03**	62.38	69.45	**83.90**	68.34	**77.71**
GPT-5.6-sol	73.61	79.32	61.66	**73.53**	77.61	**74.77**	73.94
GPT-5.5	72.79	78.59	62.09	71.75	79.59	72.14	68.32
Seed-2.1-Pro	67.19	73.15	57.26	61.35	77.83	65.61	62.91
Kimi-K2.6	59.95	67.70	51.17	54.62	71.91	51.38	58.60
GLM-5.1	60.79	65.66	51.22	50.20	78.53	52.18	66.50
Qwen-3.6-Max	59.46	66.51	**62.45**	50.33	75.03	48.83	59.26
DeepSeek-V4-Pro	61.11	69.42	51.11	54.53	73.69	54.89	57.03
Gemini-3.1-Pro	49.73	54.05	41.00	43.49	61.18	45.88	51.23
Pass rate (≥ 90)							
Kimi-k3	29.55	**34.92**	20.37	20.83	**52.63**	16.67	23.81
GPT-5.6-sol	**31.27**	33.33	**27.78**	22.92	38.60	**33.33**	**28.57**
GPT-5.5	26.80	30.16	22.22	**25.00**	38.60	22.92	9.52
Seed-2.1-Pro	22.34	19.05	16.67	14.58	45.61	20.83	4.76
Kimi-K2.6	13.06	15.87	14.81	6.25	24.56	4.17	4.76
GLM-5.1	16.49	12.70	16.67	8.33	36.84	8.33	9.52
Qwen-3.6-Max	15.12	11.11	16.67	10.42	33.33	6.25	4.76
DeepSeek-V4-Pro	16.49	20.63	16.67	8.33	29.82	10.42	0.00
Gemini-3.1-Pro	6.53	6.35	11.11	6.25	8.77	0.00	4.76

This gap has two important implications. First, it helps explain why specialized vertical agents continue to provide practical value in many market-validated workflows, where consistently meeting professional standards remains essential. Second, it suggests that the next frontier for foundation models is not merely performing professional tasks, but reliably producing work products that users can adopt without further verification or revision. We further investigate the underlying causes of this gap in Section 4.4.2.

Performance varies substantially across professional domains. Table 3 reveals that current general-purpose agents do not exhibit uniform competence across real-world workflows. Among the six domains, Business is consistently the easiest: every model achieves an average score above 60, and the average pass@1 across models is also substantially higher than in any other domain. In contrast, Finance is the most challenging, with the lowest average score of 54.48%. The large gap between Business and Finance suggests that current models handle structured, relatively standardized tasks more reliably than workflows requiring sustained quantitative reasoning, cross-document consistency, and multi-step verification.

The results also reveal pronounced domain specialization. While Kimi-k3 and GPT-5.6-sol achieve nearly identical average scores overall (73.67 vs. 73.61), their strengths differ substantially across domains. Under pass rate, Kimi-k3 performs best in Medical and Business, whereas GPT-5.6-sol leads in Finance, STEM, and Education; GPT-5.5 remains strongest in Legal. A similar pattern emerges in average score, where Kimi-k3 leads in Medical, Business, and Education, GPT-5.6-sol in Legal and STEM, and Qwen-3.6-Max in Finance. Thus, even among the strongest models, no single agent consistently dominates across professional domains. These results suggest that current general-purpose agents still exhibit substantial domain-dependent variation, leaving considerable room for improving the robustness and transferability of end-to-end task execution across heterogeneous professional workflows.

Figure 4 Performance across rubric dimensions. Importance-weighted pass rates (%) of different models on the six rubric dimensions in StartupBench. Higher values indicate better performance under the corresponding evaluation dimension.

4.3 Effectiveness of Proposed Evaluation Framework

To validate the reliability of our evaluation framework, we compare its judgments against expert annotations. We randomly sample the outputs of two representative models for every task and ask domain experts to score each submission independently according to the predefined rubrics. We then compare the rubric-level decisions produced by our automatic evaluation pipeline with the corresponding human judgments. Across all evaluated rubrics, our framework achieves an average agreement rate of **92%**, indicating that the rubric decomposition and agent-based evaluation can accurately reproduce expert assessments while substantially reducing manual annotation effort.

We further investigate the necessity of evaluating each rubric independently. As an ablation, instead of assigning one rubric to one judge agent, we directly provide the complete model output together with the full rubric list to a single judge agent, requiring it to score all rubrics in one end-to-end inference. This alternative reduces the agreement with human experts to *83%*. More importantly, the holistic setting proves considerably less stable in practice. Since the judge agent must simultaneously perform long-context reasoning, maintain consistency across numerous evaluation criteria, and finally generate a structured output covering all rubric scores, failures such as malformed outputs or incomplete formatting frequently occur, forcing repeated executions. On average, each evaluation requires more than two runs before a valid result is obtained. In contrast, our rubric-wise evaluation decomposes the assessment into a collection of independent, lightweight judging tasks, yielding both substantially higher agreement with human experts and significantly better execution robustness.

4.4 Failure-Mode Analysis

Table 4 Output-format compliance on the 56 StartupBench tasks with explicit output-file requirement rubrics. Models are ranked by compliance rate.

Model	Compliance (%)
Kimi-k3	98.2
Seed-2.1-Pro	94.6
Qwen-3.6-Max	94.6
DeepSeek-V4-Pro	94.6
GPT-5.6-sol	94.6
GPT-5.5	92.9
Kimi-K2.6	91.1
Gemini-3.1-Pro	85.7
GLM-5.1	83.9

4.4.1 Outcome-level Failure Analysis

Capability differences across rubric dimensions. Figure 4 further breaks down model performance by rubric dimension. Overall, current models achieve consistently higher scores on Structure & Completeness, Information Integration, and especially Output & Presentation, indicating that they are generally capable of organizing heterogeneous information into well-structured and visually polished deliverables. In contrast, Domain-Specific Compliance emerges as the most challenging dimension across all evaluated models, while Accuracy also remains substantially weaker than the other dimensions. This suggests that although modern foundation models can produce artifacts that appear complete and professional, reliably satisfying domain-specific regulations, business conventions, and numerical correctness continues to be a major obstacle for real-world deployment.

The two strongest models, GPT-5.5 and Seed-2.1-Pro, outperform the remaining systems across nearly all rubric dimensions rather than relying on a single specialized capability. Their largest advantages are observed in Structure & Completeness, Calaulation Precision, Information Integration, and Output & Presentation, demonstrating stronger end-to-end workflow execution and artifact construction capabilities. Nevertheless, even these leading models obtain their lowest scores on Domain-Specific Compliance, indicating that adherence to professional standards and domain-specific requirements remains the primary bottleneck even for state-of-the-art systems.

Output format non-compliance. Some failures arise from models not producing deliverables in the file type explicitly required by the task. Unlike most failure modes discussed above, this requirement involves neither complex reasoning nor domain knowledge and can be verified deterministically from file extensions. Nevertheless, no model achieves perfect compliance on the 56 tasks with explicit output-format requirements (Table 4), with success rates ranging from 98.2% to 83.9%. Most violations fall into two recurring patterns: generating an incorrect file type (e.g., returning `.md`/`.txt` instead of the required `.pdf`) and omitting part of the requested deliverables (e.g., producing only one of the required `.xlsx` and `.docx` files). These errors suggest that even elementary deliverable-level constraints remain a non-negligible source of failures for current foundation models.

4.4.2 Behavior-level Failure Analysis

Complex Instruction Following. Complex real-world workflows typically impose numerous constraints on the final deliverable, including structural organization, computation logic, formatting, and executable correctness. Although these requirements are explicitly specified in the task description, current models frequently exhibit partial compliance: they satisfy the most visible requirements while overlooking other equally critical constraints. As a result, the generated artifact appears largely complete from a superficial inspection, yet fails to satisfy the complete acceptance criteria required for direct use. This behavior indicates that models often optimize for producing outputs that look correct, rather than ensuring that every instruction governing the final deliverable has been faithfully executed.

A representative example is shown in Figure 5. The task requires generating a complete HR payroll workbook from multiple source sheets under a diverse set of structural, computational, and formatting constraints.

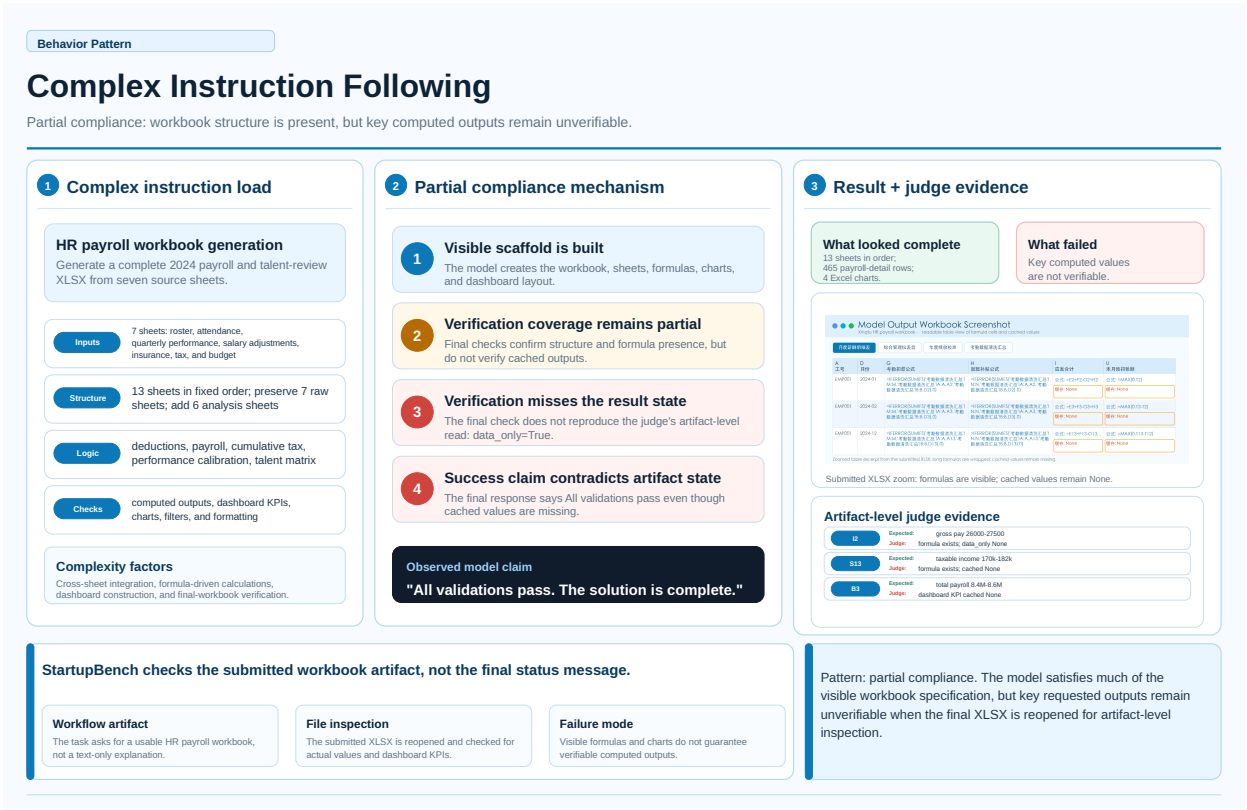


Figure 5 A representative case of complex instruction following failure. Although the generated workbook appears complete, artifact-level evaluation reveals that several critical acceptance conditions remain unsatisfied, illustrating a common pattern of partial compliance.

Although the model produces an artifact that appears largely complete—including all required worksheets, formulas, charts, and dashboard layouts—the artifact-level evaluation reveals that several critical acceptance conditions remain unsatisfied. Specifically, key cached values required for downstream verification are missing, rendering the workbook unverifiable despite its polished appearance. This case illustrates a common pattern of partial compliance, where the model satisfies the most visible requirements while overlooking less obvious but equally essential deliverable constraints.

Self-Verification Hallucination. We also observe that many models fail not because they cannot complete the required workflow, but because they incorrectly assume that successful execution implies successful delivery. After producing a plausible artifact, the model often summarizes the intended reasoning process or completed steps as evidence that the task has been verified, without independently inspecting the final deliverable against the original user requirements. Consequently, subtle yet critical errors—such as incorrect cell values, unmatched identifiers, missing constraints, or formatting inconsistencies—remain undetected despite an otherwise convincing output. This behavior reflects a form of self-verification hallucination, where the model mistakes confidence in its own reasoning process for evidence that the generated artifact satisfies the acceptance criteria. In real-world office workflows, however, successful completion requires validating the final deliverable itself rather than merely confirming the execution process, making this a common source of deployment-critical failures. A typical case is shown at Appendix C.1.

Domain-Specific Expertise. Another major source of failure arises from insufficient domain-specific expertise, corresponding to the relatively poor performance of the rubrics of dimension "Domain Knowledge". Unlike general instruction-following errors, these failures are driven by the unique operational requirements of different professional domains. Although models often generate outputs that appear fluent and professionally formatted, they frequently fail to satisfy the domain-specific constraints that determine whether a deliverable is actually

Table 5 Performance under different general-purpose agent frameworks. For each model, we report the average score under three general-purpose frameworks and the performance range (max–min) across frameworks.

Model	Hermes	Claude Code	nanobot	Max--Min
GPT-5.5	71.00	70.11	72.79	2.68
GLM-5.1	60.20	60.18	60.79	0.61
Qwen-3.6-Max	59.10	57.38	59.46	2.08
Average	63.43	62.56	64.35	1.79

usable. As a result, the dominant failure patterns vary substantially across domains, reflecting different notions of correctness, evidence, precision, and actionability required by real-world workflows.

From a domain perspective, different professional areas expose distinct failure characteristics. Business and management tasks primarily stress artifact correctness, particularly for spreadsheets, dashboards, formulas, and structured reports. Finance tasks emphasize source attribution, numerical precision, valuation assumptions, and temporal consistency, where errors commonly arise from insufficient reconciliation, inadequate numerical self-verification, or inconsistent accounting conventions. Legal tasks are less constrained by legal writing style than by the completeness and correctness of legal reasoning, including missing citations, incomplete factual coverage, insufficient statutory support, or incorrect mapping between facts and applicable rules. Medical tasks achieve relatively higher average scores but pose substantially greater safety risks, since mistakes in medication timing, continuation criteria, or discharge planning directly undermine clinical usability. STEM and computer science tasks instead require precise object-level grounding, demanding accurate relationships among code, engineering drawings, images, bills of materials, and system components. Finally, education and humanities tasks place greater emphasis on fine-grained rule following, such as curriculum requirements, credit allocation, document organization, and textual coherence. A typical example of failure caused by insufficient domain-specific expertise is shown at Appendix C.2.

4.5 Impact of Harness

4.5.1 Effect of General-Purpose Harness

In the main experiment, all models are evaluated under nanobot Agent framework. To examine whether the observed performance is primarily determined by the choice of agent framework rather than the underlying model, we re-evaluate the benchmark using two more representative general-purpose agent frameworks: Hermes and Claude Code. All frameworks are evaluated under the same task set and identical evaluation protocol, with only the execution framework replaced.

As shown in Table 5, replacing the general-purpose framework leads to only minor performance differences. Across all evaluated models, the average variation between the best and worst framework is only 1.66 points. GLM-5.1 is almost completely unaffected (0.40 points), while even the largest difference observed on GPT-5.5 is only 2.85 points. More importantly, the relative ordering of models remains unchanged across all three frameworks. These results suggest that StartupBench is largely robust to the choice of general-purpose agent framework. Although different frameworks adopt different prompting strategies, tool interfaces, and interaction mechanisms, simply replacing one general-purpose framework does not substantially alter end-to-end task performance. Consequently, the performance gap observed on StartupBench primarily reflects the capability of the underlying foundation models rather than implementation-specific characteristics of a particular framework.

4.5.2 General-Purpose Agents versus Specialized Agents

To further understand the role of agent frameworks, we compare general-purpose agent frameworks with the specialized startup agents from which StartupBench tasks are derived. Unlike the previous experiment, this comparison evaluates each task using its corresponding production startup agent, which has been

Table 6 General-purpose versus specialized agent systems. Oracle General denotes the highest score achieved by any evaluated general-purpose agent for each task.

Agent System	Avg Score	Pass Rate
General (All Runs)	64.26	19.74
General (Oracle)	71.75	28.06
Specialized Agent	**83.50**	**39.18**

specifically optimized for its target workflow through domain-specific model selection, prompting strategies, tool orchestration, and workflow design.

Table 6 summarizes the comparison between general-purpose and specialized agent systems. Across all executions, general-purpose agents achieve an average score of 64.26 and a Pass rate of 19.74%, substantially below the specialized agents at 83.50 and 39.18%, respectively. Since specialized agents are explicitly designed and optimized for their target domains, we further consider a stronger oracle setting for the general-purpose agents: for each model–task pair, we retain the best result among three independent trials. This oracle selection raises the average score to 71.75 and Pass rate to 28.06%, but still falls well short of the specialized agents, with gaps of 11.75 points in average score and 11.12 percentage points in Pass@1. Thus, the performance gap cannot be explained simply by stochastic variation or occasional execution failures: even when general-purpose agents are allowed multiple attempts and evaluated by their best observed outcome, specialized agent systems remain substantially more effective on these tasks. Combined with the failure-mode analysis presented earlier, these findings provide a more complete picture of the remaining gap between general-purpose foundation models and specialized startup agents. While today's foundation models still cannot fully replace domain-specialized agents under a unified general-purpose framework, their shortcomings are concentrated in a relatively clear set of capabilities, including complex instruction following, domain-specific knowledge, professional operational conventions, and long-horizon workflow execution. This suggests that the observed advantage of specialized systems does not necessarily require fundamentally different agent architectures, but may instead reflect capabilities that current general-purpose models have yet to acquire reliably. As these underlying capabilities continue to improve, general-purpose agents may increasingly accomplish valuable professional end-to-end tasks without relying on domain-specific agent harness designs.

5 Conclusion

Current foundation models have made substantial progress toward performing realistic professional workflows, yet a significant gap remains before they can consistently deliver work products that are ready for direct professional use. Our results suggest that this gap lies not primarily in whether models can perform the required work, but in whether they can produce deliverables that satisfy the full set of practical acceptance criteria without further human correction. Across diverse domains, the remaining failures consistently arise from a relatively small set of capabilities—including complex instruction following, domain-specific knowledge, professional operational conventions, and long-horizon workflow execution—ultimately preventing otherwise strong solutions from becoming fully reliable deliverables. By grounding evaluation in market-validated AI workflows and systematically characterizing these bottlenecks, StartupBench provides not only a realistic benchmark for measuring practical agent capabilities, but also a concrete roadmap for future foundation-model development. We hope StartupBench will facilitate the development of increasingly capable general-purpose agents that are able not only to perform valuable professional work, but also to deliver production-ready results that users can trust without relying on task-specific agent designs.

6 Contributions

Project Leads
Liya Zhu, Xin Ma, Tao Liu, Haodong Wang, Ge Zhang

Core Contributors
Jingzhe Ding*, Qingshui Gu*, Yongjie Zhong*, Jinxiang Meng*, Yuan Gao*, Yunqiu Zhou*, Hao Zhu*, Jifeng He*, Yongzhi Liao*, Xinyi Zhang*, Chaoxin Li*, Yi Zhu*, Xi Lin*, Duju Zeng*, Xiang Gao*, Wen Zhang*, Yunyang Wang*, Duo Wang*

Contributors
Huan Zhou*, Zuo Wang*, Jin Chen*, Kaiyuan Zhang*, Chuqian Yu*, Tianhao Yu*, Longxiang Liu*, Jianbo Xue*, Huimin Che*, Jiahao Wang

Sponsor Committee
Yujia Qin*, Jiaheng Liu

Corresponding Authors
Ge Zhang (zhangge.eli@bytedance.com)
Shen Yan (sheny@bytedance.com)
Xiaolong Chang (changxiaolong@bytedance.com)
Wenhao Huang (huang.wenhao@bytedance.com)

References

[1] Bytedance Seed. Seed2.0 model card: Towards intelligence frontier for real-world complexity. Technical report, Bytedance, 2025. URL https://lf3-static.bytednsdoc.com/obj/eden-cn/lapzild-tss/ljhwZthlaukjlkulzlp/seed2/0214/Seed2.0%20Model%20Card.pdf.

[2] DeepSeek-AI. Deepseek-v4: Towards highly efficient million-token context intelligence, 2026. URL https://huggingface.co/collections/deepseek-ai/deepseek-v4. Technical Report.

[3] Alexandre Drouin, Maxime Gasse, Massimo Caccia, Issam H Laradji, Manuel Del Verme, Tom Marty, Léo Boisvert, Megh Thakkar, Quentin Cappart, David Vazquez, et al. Workarena: How capable are web agents at solving common knowledge work tasks?, 2024. URL https://arxiv. org/abs/2403.07718.

[4] Xeron Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, Minghao Liu, Yiming Liang, Xiaolong Jin, Zhenlin Wei, Chujie Zheng, et al. Supergpqa: Scaling llm evaluation across 285 graduate disciplines. Advances in Neural Information Processing Systems, 38, 2026.

[5] Google Cloud. Gemini 3.1 Pro Model Card, 4 2026. URL https://docs.cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/3-1-pro?hl=zh-cn.

[6] Xiaomeng Hu, Yinger Zhang, Fei Huang, Jianhong Tu, Yang Su, Lianghao Deng, Yuxuan Liu, Yantao Liu, Dayiheng Liu, and Tsung-Yi Ho. Occubench: Evaluating ai agents on real-world professional tasks via language environment simulation, 2026. URL https://arxiv.org/abs/2604.10866.

[7] Seungone Kim, Jay Shin, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Ryan Shin, Sungdong Kim, James Thorne, Minjoon Seo, et al. Prometheus: Inducing fine-grained evaluation capability in language models. In International Conference on Learning Representations, 2024.

[8] Fangyu Lei, Jinxiang Meng, Yiming Huang, Junjie Zhao, Yitong Zhang, Jianwen Luo, Xin Zou, Ruiyi Yang, Wenbo Shi, Yan Gao, et al. Dacomp: Benchmarking data agents across the full data intelligence lifecycle. arXiv preprint arXiv:2512.04324, 2025.

[9] Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. Agentbench: Evaluating llms as agents. In International Conference on Learning Representations, 2024.

[10] Jinxiang Meng, Shaoping Huang, Fangyu Lei, Jingyu Guo, Haoxiang Liu, Jiahao Su, Sihan Wang, Yao Wang, Enrui Wang, Ye Yang, et al. Dv-world: Benchmarking data visualization agents in real-world scenarios. arXiv preprint arXiv:2604.25914, 2026.

[11] Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants. In International Conference on Learning Representations, 2024.

[12] Moonshot AI. Kimi K2.6: Advancing open-source coding. https://www.kimi.com/blog/kimi-k2-6, 2026. Accessed: 2026-06-02.

[13] OpenAI. Introducing gpt-5.5, 4 2026. URL https://openai.com/index/introducing-gpt-5-5/. System Card: https://deploymentsafety.openai.com/gpt-5-5/gpt-5-5.pdf.

[14] Krista Opsahl-Ong, Arnav Singhvi, Jasmine Collins, Ivan Zhou, Cindy Wang, Ashutosh Baheti, Owen Oertell, Jacob Portes, Sam Havens, Erich Elsen, et al. Officeqa pro: An enterprise benchmark for end-to-end grounded reasoning. arXiv preprint arXiv:2603.08655, 2026.

[15] Tejal Patwardhan, Rachel Dias, Elizabeth Proehl, Grace Kim, Michele Wang, Olivia Watkins, Simón Posada Fishman, Marwan Aljubeh, Phoebe Thacker, Laurance Fauconnet, et al. Gdpval: Evaluating ai model performance on real-world economically valuable tasks. arXiv preprint arXiv:2510.04374, 2025.

[16] Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, et al. Humanity's last exam. arXiv preprint arXiv:2501.14249, 2025.

[17] Qwen Team. Qwen3.6-Max-Preview: Smarter, sharper, still evolving, April 2026. URL https://qwen.ai/blog?id=qwen3.6-max-preview.

[18] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. Advances in neural information processing systems, 36:68539–68551, 2023.

[19] Wentao Shi, Yu Wang, Yuyang Zhao, Yuxin Chen, Fuli Feng, Xueyuan Hao, Xi Su, Qi Gu, Hui Su, Xunliang Cai, et al. Aj-bench: Benchmarking agent-as-a-judge for environment-aware evaluation. arXiv preprint arXiv:2604.18240, 2026.

[20] Yiyou Sun, Xinyang Han, Weichen Zhang, Yuanbo Pang, Tianyu Wang, Yuhan Cao, Yixiao Huang, Chris Duroiu, Haoyun Zhang, Jeffrey Lin, et al. Agents' last exam. arXiv preprint arXiv:2606.05405, 2026.

[21] Zirui Tang, Xuanhe Zhou, Yumou Liu, Linchun Li, Yukai Wu, Weizheng Wang, Hongzhang Huang, Wei Zhou, Jun Zhou, Jiachen Song, et al. Workspace-bench 1.0: Benchmarking ai agents on workspace tasks with large-scale file dependencies. arXiv preprint arXiv:2605.03596, 2026.

[22] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. Advances in Neural Information Processing Systems, 37:95266–95290, 2024.

[23] Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track, 2024.

[24] Frank Fangzheng Xu, Yufan Song, Boxuan Li, Yuxuan Tang, Kritanjali Jain, Mengxue Bao, Zora Wang, Xuhui Zhou, Zhitong Guo, Murong Cao, et al. Theagentcompany: benchmarking llm agents on consequential real world tasks. Advances in Neural Information Processing Systems, 38, 2026.

[25] Qianyu Yang, Yang Liu, Jiaqi Li, Jun Bai, Hao Chen, Kaiyuan Chen, Tiliang Duan, Jiayun Dong, Xiaobo Hu, Zixia Jia, et al. Onemillion-bench: How far are language agents from human experts? arXiv preprint arXiv:2603.07980, 2026.

[26] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In 11th International Conference on Learning Representations, ICLR 2023, 2023.

[27] Runyang You, Hongru Cai, Caiqi Zhang, Qiancheng Xu, Meng Liu, Tiezheng Yu, Yongqi Li, and Wenjie Li. Agent-as-a-judge. arXiv preprint arXiv:2601.05111, 2026.

[28] Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, et al. Glm-5: from vibe coding to agentic engineering. arXiv preprint arXiv:2602.15763, 2026.

[29] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. Advances in neural information processing systems, 36:46595–46623, 2023.

[30] Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. In International Conference on Learning Representations, 2024.

[31] Liya Zhu, Jingzhe Ding, Jian Zhang, Jianbo Xue, Shihao Liang, Ge Zhang, Yi Zhu, Duju Zeng, Xiang Gao, Qingshui Gu, Mailun Gao, Huimin Che, Yan Zhao, Peiheng Zhou, Haojun Wang, Chaobo Xian, Lili Le, Chi Wu, Yiwei Liu, Shengda Long, Jiale Yang, Fangzhi Xu, Sijin Wu, Haodong Duan, Chao He, Zhaojian Li, Minchao Wang, Huan Zhou, Jiani Hou, Chuqian Yu, Weiran Shi, Hongwan Gao, Jiamin Chen, Guanhong Chen, Tingqin Luo, Kaiyuan Zhang, Zhixin Yao, Qing Hua, Yuhao Jiang, Jin Chen, Pu Chen, Zhenyu Hu, Xingyu Li, Zhengxuan Jiang, Meng Cao, Tianfeng Long, Haozhe Wang, Mingzhang Wang, Yichen Zhang, Yiming Dai, Chenchen Zhang, Jiaying Wang, Xinying Liu, Xingzu Liu, Lingling Zhang, Xinjie Chen, Yujia Qin, Wangchunshu Zhou, Zhiyong Wu, Yang Liu, Jiaheng Liu, Lei Zhang, Shen Yan, Wenhao Huang, Zaiyuan Wang, and Xiaolong Chang. Workflow-gym: Towards long-horizon evaluation of computer-use agentic tasks in real-world professional fields, 2026. URL https://arxiv.org/abs/2606.11042.

[32] Mingchen Zhuge, Changsheng Zhao, Dylan R. Ashley, Wenyi Wang, Dmitrii Khizbullin, Yunyang Xiong, Zechun Liu, Ernie Chang, Raghuraman Krishnamoorthi, Yuandong Tian, Yangyang Shi, Vikas Chandra, and Jürgen Schmidhuber. Agent-as-a-judge: Evaluate agents with agents. In Forty-second International Conference on Machine Learning, 2025.

Appendix

A Tutorial of StartupBench Task Annotation

A.1 Task Construction

A STARTUP task can be defined as a triple $\mathcal{T} = (q, \mathcal{E}, \mathcal{R})$. The experts are required to provide all the elements of a task:

- **Input query**: The query q is a natural-language statement of the task, specifying the user instruction that initiates the task. Derived from authentic scenarios of the work of annotators, the input query is supposed to reflect realistic user intent, including task background, implicit constraints and expected deliverables.
- **Task workspace**: The environment $\mathcal{E}$ denotes the complete workspace provided to the agent, including the multimodal source files together with any additional resources and interaction setting required to complete the task. The task workspace provides a self-contained execution environment for the agent. Each workspace includes all necessary input artifacts (e.g., files, datasets, or structured resources) required to complete the task, and ensures that the agent has access to sufficient input information to solve the problem. This design enables reproducible evaluation under controlled conditions while preserving realistic workflow structure.
- **Evaluation rubrics**: The rubric $\mathcal{R} = \{(p_i, w_i)\}_{i=1}^{n}$ is a checklist within which every item consists of a natural-language scoring point p_i and a positive importance weight w_i. The evaluation rubrics define a set of fine-grained criteria used to assess task completion quality. Due to the complexity and multi-faceted nature of real-world workflows, each task is evaluated using multiple rubric items covering different dimensions. We categorize all rubrics in StartupBench to 6 dimensions, which is detailed in Appendix B.

Together, these components preserve the structure of real-world agent workflows while enabling systematic and objective evaluation. Tasks may require information synthesis, constraint following, structured generation, or domain-specific reasoning and judgment.

A.2 Cross Validation

To ensure reliability of the quality of data, in our construction process each task is independently reviewed by at least one additional domain expert with relevant professional experience. Rather than focusing only on annotation consistency, reviewers validate the realism and correctness of the task from multiple complementary perspectives.

- **Task authenticity.** Reviewers verify the authenticity of both the task description and the accompanying workspace. This includes checking that the input query reflects realistic user requests, that all provided materials are factually correct, and that no domain-specific knowledge errors exist.
- **Workflow fidelity.** Reviewers examine whether the task faithfully represents real-world workflows in the corresponding profession. Tasks that deviate from practical working procedures or fail to reflect genuine job responsibilities are revised or discarded.
- **Rubric validity.** Reviewers inspect the evaluation rubrics to ensure that the assessed capabilities are meaningful for the target workflow, that rubric items are clear and unambiguous, and that their relative weights reasonably reflect task priorities.
- **Ground-truth verification.** Reviewers validate the reference solution (ground truth) by evaluating it against the finalized rubrics. The ground-truth deliverable is required to achieve a full score. If inconsistencies are identified between the reference solution and the evaluation criteria, both the solution and the rubrics are revised until they are mutually consistent.

B Rubrics of StartupBench

To facilitate consistent evaluation across heterogeneous real-world tasks, we organize all evaluation rubrics into six high-level categories according to the primary capability they assess. Rather than being tied to specific application domains, these categories capture common dimensions of deliverable quality that recur across office tasks, including correctness, completeness, data processing, professional reasoning, engineering quality, and presentation. This taxonomy enables more interpretable analysis of model strengths and weaknesses while maintaining a unified evaluation framework across different task types.

- **Structure & Completeness.** Evaluates whether the required deliverables are fully produced with the correct organization, file structure, required components, and overall completeness.
- **Accuracy.** Evaluates the correctness of numerical results, logical reasoning, formula execution, factual consistency, and other objective computations.
- **Information Integration.** Evaluates data cleaning, transformation, aggregation, statistical analysis, feature engineering, modeling, and other structured data manipulation workflows.
- **Domain-Specific Compliance.** Evaluates whether outputs satisfy domain-specific professional requirements, including business logic, financial principles, legal reasoning, medical practice, educational standards, and other expert knowledge.
- **Engineering & Format.** Evaluates implementation quality, coding conventions, spreadsheet engineering, document formatting, reproducibility, robustness, naming conventions, and compliance with technical specifications.
- **Output & Presentation.** Evaluates the quality of visual presentation and communication, including charts, layouts, formatting, readability, report organization, and overall usability of the final deliverables.

Table 7 summarizes the distribution of all 2,453 rubrics in StartupBench. Accuracy-related rubrics constitute the largest category (38.9%), reflecting the importance of producing objectively correct work products. Structure & Completeness and Domain Knowledge together account for another 42.8%, highlighting that successful completion of realistic office workflows requires not only correct computations but also complete deliverables and professional domain reasoning.

Table 7 Distribution of rubric categories in StartupBench.

Category	Percentage
Structure & Completeness	27.31%
Accuracy	38.89%
Data Processing	9.46%
Domain Knowledge	15.49%
Engineering & Format	6.20%
Output & Presentation	2.65%
Total	**100.00%**

C Examples of Behavior-level Failure Modes

C.1 Self-Verification Hallucination

A representative example is shown in Fig. 6. The task requires the model to filter all level-11 members, simulate route check-ins until each member reaches the level-12 threshold, and generate an Excel workbook satisfying a set of precise value and formatting constraints. The produced workbook appears largely complete, containing the required worksheet, member records, and output columns, leading the model to conclude that the task has been successfully finished. However, artifact-level inspection reveals multiple acceptance-critical errors, including incorrect effective-point values and mismatched member names. The underlying issue is not that the model fails to perform the required computation, but that it never verifies the generated artifact itself. Instead,

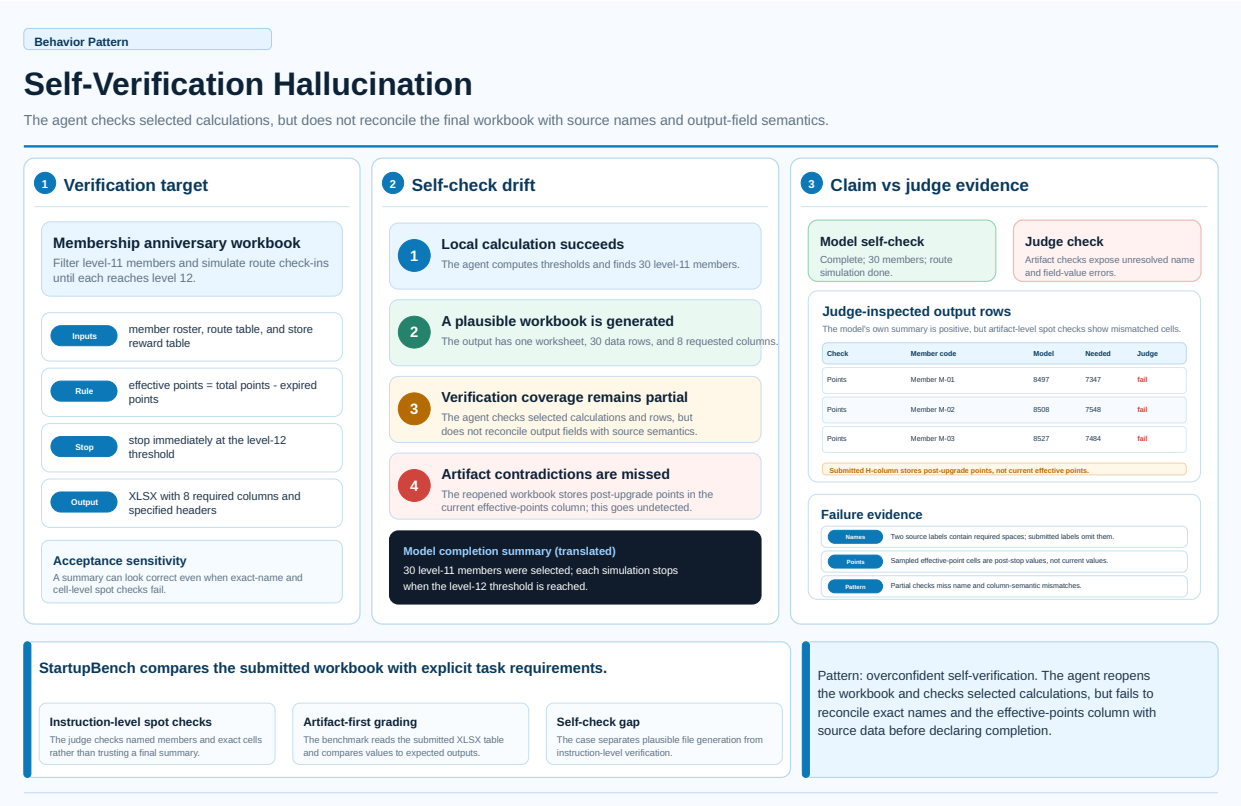


Figure 6 A representative example of self-verification hallucination. The model generates a plausible workbook and concludes that the task has been completed based on its execution process. However, artifact-level spot checks expose acceptance-critical errors in the final deliverable, showing that the model mistakes a progress summary for actual verification instead of validating the generated artifact against the original task requirements.

it treats a successful execution summary as evidence that the final deliverable has been validated. Consequently, localized yet user-critical errors remain unnoticed, illustrating a form of self-verification hallucination, where confidence is derived from the reasoning process rather than from independently checking the completed artifact against the original task requirements.

C.2 Domain-Specific Failure

A representative example of failure caused by insufficient domain-specific clinical actionability is shown at Figure 7. The model generates a well-formatted multidisciplinary treatment plan that appears clinically professional, yet violates multiple safety-critical treatment constraints. The failure stems not from poor writing quality, but from an inability to convert medical knowledge into actionable and clinically reliable decisions.

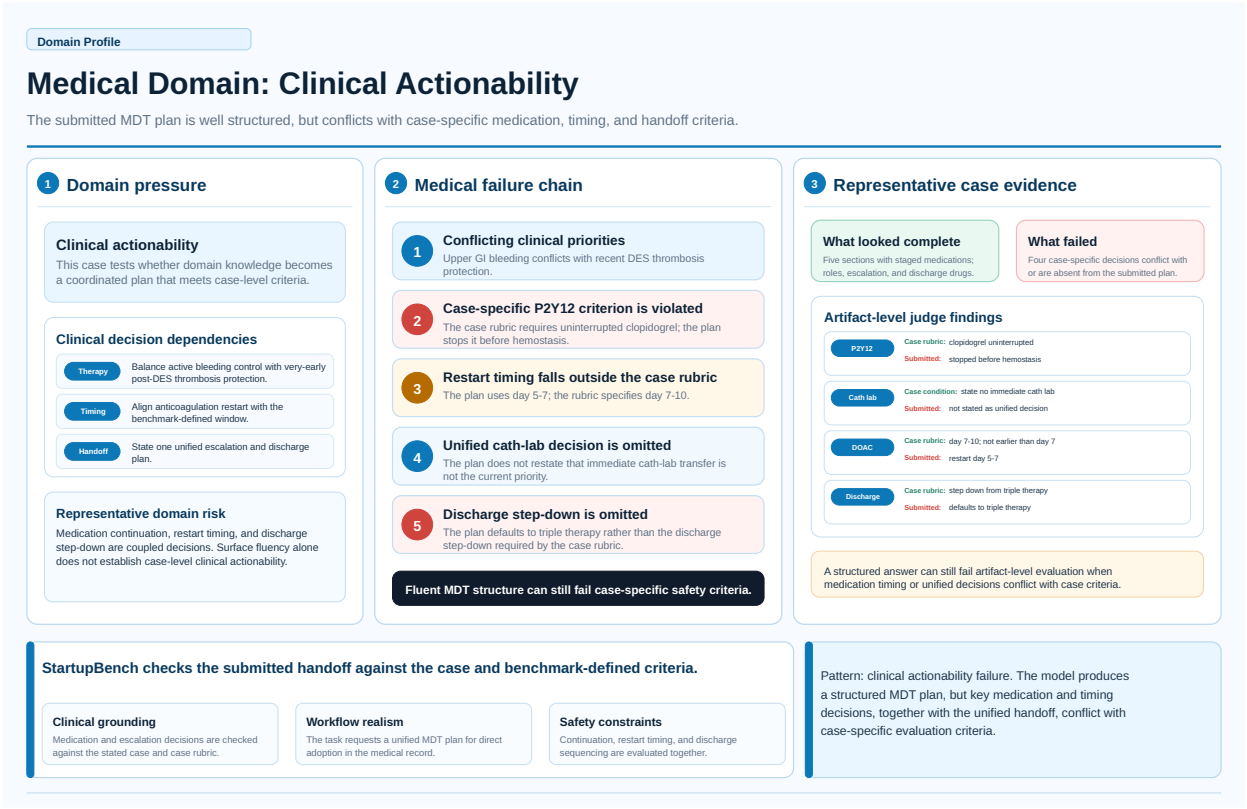


Figure 7 A representative example of failure caused by insufficient domain-specific clinical actionability.