WORKFIT BENCH: BENCHMARKING MODEL–SCENARIO FIT ACROSS REAL-WORLD AGENT WORKFLOWS

Anonymous authors
Paper under double-blind review

ABSTRACT

Deploying agents in real-world work settings requires balancing task performance against model invocation costs. However, existing benchmarks primarily focus on general capabilities or aggregate rankings, offering limited guidance for model selection across diverse work scenarios. We introduce **WorkFit Bench**, a benchmark that treats Skills as reusable units of work and comprises 162 tasks across nine categories of real-world work. WorkFit Bench compares task completion across categories and difficulty levels and relates it to estimated model cost; the evaluation also records token use and elapsed time. Overall completion rates range from 19.8% to 43.8%, with several lower-cost models close to the highest score. Under an idealized cache-reuse assumption, public-list-price estimates range from $8.83 to $536.54 for one pass through all 162 tasks. Model rankings vary across scenarios and difficulty levels. These results provide evidence for choosing lower-cost models when they meet a scenario's completion requirements. The repository is available at `https://github.com/sbench-maker/WorkFit-Bench`.

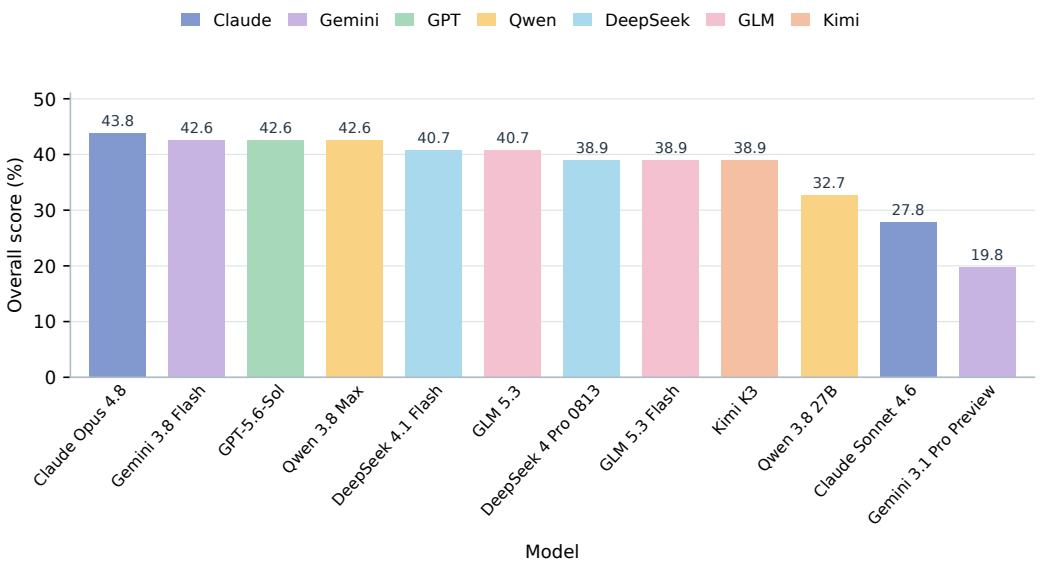


Figure 1: Overall task completion rates for the 12 evaluated models on 162 WorkFit Bench tasks. Models are ordered by completion rate, and bar colors distinguish model families. Missing submissions count as unsuccessful tasks.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1 INTRODUCTION

Large language model (LLM) agents are increasingly used for work that ends in an artifact or a change to a working environment rather than a short answer. A user may ask an agent to reconcile records, prepare a document, analyze data, or resolve an operational problem. Completing such requests requires the agent to interpret a goal, inspect inputs, coordinate tools, and deliver a usable result, often interleaving reasoning and actions (Yao et al., 2023). The choice of underlying model therefore affects both task performance and the resources spent to achieve it.

Existing LLM benchmarks measure broad capabilities across academic tasks (Hendrycks et al., 2021; Srivastava et al., 2023) and compare models across scenarios and metrics (Liang et al., 2023). AgentBench evaluates agents across interactive environments (Liu et al., 2024); SWE-bench focuses on repository issue resolution (Jimenez et al., 2024); OSWorld evaluates computer use in real applications (Xie et al., 2024); and TheAgentCompany studies professional tasks in a simulated workplace (Xu et al., 2025). These studies establish broad capability and task-completion measurements, but do not directly answer which model offers the appropriate completion–cost trade-off for a particular Skill-grounded work scenario.

This distinction matters because work requests differ in their functional demands and complexity. Some require extended planning, coordination across tools, or recovery from failed steps. Others have a clear objective and a stable workflow, for which a less expensive model may be sufficient. A model that leads an overall ranking need not offer the best completion–cost trade-off in every scenario. Consequently, the practical question is not only which model has the highest average score, but which model can meet the completion requirement of a given work scenario at an acceptable cost. Research on LLM cascades and routing has made this trade-off explicit (Chen et al., 2024; Ong et al., 2025; Dekoninck et al., 2025); evaluating it for agents also requires evidence from complete workflows rather than isolated questions or short-form responses.

We introduce *WorkFit Bench* to study this model–scenario fit. The benchmark contains 162 executable tasks across nine categories of work, with 18 tasks per category. Each task is grounded in an existing Skill, a reusable package of instructions and supporting resources for a class of work (Li et al., 2026b). Skills provide a structured way to turn work requests into agent tasks, while the research question remains which model is appropriate for which scenario. The current evaluation measures binary task completion across categories and difficulty levels and records execution resources and missing submissions. We also report an idealized-cache list-price estimate of model invocation cost; actual billing remains unaudited.

Our contributions are threefold.

First, we release a dataset of 162 executable work tasks, together with their Skills, environments, input data, and completion-only verifiers. The dataset defines nine work-scenario categories and documents the functions covered by each category.

Second, we evaluate 12 model configurations on the full benchmark, reporting both overall task completion and scenario-specific results under a shared agent and evaluation protocol.

Third, we derive model-selection guidance that accounts for work scenario, estimated invocation cost, and task difficulty, showing when the additional expense of a stronger model is supported by its completion gains.

2 RELATED WORK

General language-model evaluation. MMLU measures knowledge and problem solving across 57 subjects (Hendrycks et al., 2021), and BIG-bench collects a broad set of diverse language-model tasks (Srivastava et al., 2023). HELM explicitly organizes evaluation by scenarios and multiple metrics, including efficiency (Liang et al., 2023). WorkFit Bench follows the motivation to compare models across settings, while measuring completed agent work and its estimated invocation cost within each work scenario.

Executable agent benchmarks. AgentBench evaluates LLM agents in interactive environments (Liu et al., 2024). WebArena tests functional task completion on realistic websites (Zhou et al.,

2024), while GAIA probes general assistants on questions requiring reasoning, browsing, and tool use (Mialon et al., 2024). SWE-bench anchors software engineering evaluation in real repository issues and executable tests (Jimenez et al., 2024). OSWorld provides computer tasks with environment setup and execution-based checks (Xie et al., 2024), while TheAgentCompany studies work performed by a digital employee in a simulated organization (Xu et al., 2025). WorkFit Bench shares the emphasis on complete tasks and observable outcomes. Its emphasis is the comparison of models *within* categories of work, using a balanced category design and a shared resource accounting protocol.

Agent Skills and reusable workflows. SkillsBench compares matched agent runs with and without curated Skills across diverse tasks (Li et al., 2026b). SWE-Skills-Bench examines their marginal utility in software engineering (Han et al., 2026), and recent work studies retrieval and refinement of Skills from a large collection rather than assuming a supplied task-specific Skill (Liu et al., 2026). WorkFit Bench uses existing Skills to identify realistic use cases and provide task context. It compares model–scenario fit with Skills supplied, rather than estimating the causal benefit of adding a Skill.

Cost-aware model selection. FrugalGPT studies combinations of LLMs that reduce inference expense while preserving task quality (Chen et al., 2024); RouteLLM learns routing policies from preference data (Ong et al., 2025). RouterBench and LLMRouterBench evaluate routing in terms of both performance and cost (Hu et al., 2024; Li et al., 2026a). Cascade routing combines model selection with escalation to improve the quality–cost trade-off (Dekoninck et al., 2025), while cost-aware contrastive routing targets the cheapest accurate model for each prompt (Shirkavand et al., 2025). WorkFit Bench can supply scenario-specific evidence for such choices, but it does not itself claim a trained router. Its task scores and execution traces make it possible to ask which models meet a work category's completion requirement and what resources they use to do so.

3 DATASET

Figure 2 shows how we constructed WorkFit Bench. We define a work scenario as a class of everyday tasks with a shared functional goal and core operations. Skills encode reusable agent capabilities and workflows that help identify these scenarios. We collected and screened 1,917 Skills from GitHub, Skill marketplaces, and everyday tools. We grouped them into nine categories and built 162 executable tasks, with 18 tasks per category.

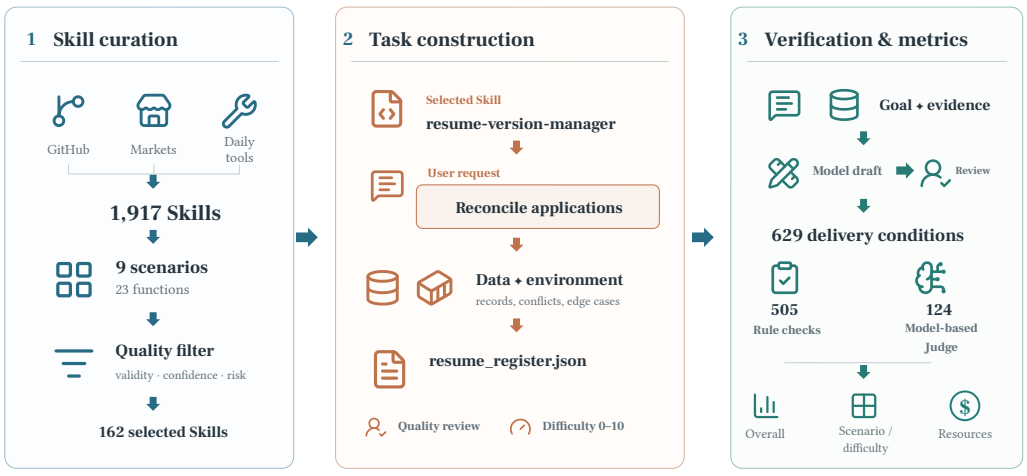


Figure 2: WorkFit Bench construction. Skills are screened and sampled across nine scenarios. Each selected Skill yields a user request, local data, and an execution environment. Verifiers assess the deliverable, and the benchmark records completion, difficulty, and resource use.

3.1 DATASET CONSTRUCTION

From Skills to Scenarios. Skills encode repeatable procedures and the agent capabilities used in everyday work. We collected 1,917 Skills from GitHub, Skill marketplaces, and everyday tools. We classified them by intended use and core operations into nine work-scenario categories and 23 finer-grained functions. We screened the packages for validity, quality, classification confidence, and external-dependency risk. We then sampled 18 Skills per category to balance scenario coverage. Table 1 reports the pool and sample sizes.

Table 1: Skill pool size, selected tasks, and mean task difficulty by work scenario. Each selected Skill yields one task; difficulty is scored from 0 to 10.

Work scenario	Skill pool	Tasks	Mean difficulty
Software development and quality engineering	666	18	5.0
Deployment, operations, security, and reliability	422	18	5.6
Information retrieval and knowledge synthesis	287	18	6.2
Content and document production	149	18	4.8
Planning, communication, and collaboration	121	18	6.2
Data analysis and presentation	115	18	6.5
Business processes and transaction execution	76	18	6.4
AI and agent engineering	42	18	6.4
Data collection and processing	39	18	5.0
Total	**1,917**	**162**	**5.8**

From Scenarios to Tasks. For each selected Skill, we derive a natural user request from its scenario category and type of work. The request must reflect a plausible goal and require an observable result. We check alignment among the Skill's workflow, the available inputs, and the intended outcome. This avoids tasks that share a topic with a Skill but do not exercise its core capability.

We then synthesize local data and an execution environment for the scenario. The data preserve realistic relationships, variation, and edge cases. The environment supports repeatable execution while retaining the key decisions of the original workflow. We review whether the task can be completed and evaluated as intended. Unsuitable scenarios are revised or replaced.

Calibrating Task Difficulty. We define difficulty by the work a capable agent must perform with the provided Skill and environment. The assessment considers step dependencies, independent judgment, evidence synthesis, tool use, and constraints. An independent judge accounts for the assistance already supplied by the Skill and does not see model outcomes. Each of the five aspects contributes 0–2 points to a 0–10 task score. All 162 tasks receive a difficulty score. During construction, we aimed to keep category-level difficulty near the middle of the scale. Table 1 reports the observed mean for each category. The full judge prompt appears in Appendix B.

3.2 VERIFICATION AND METRICS

Verifying Outcomes. Each verifier is built around the user's expected result and the evidence supplied with the task. It determines whether an agent has fulfilled the request in a given work scenario. Task-specific delivery conditions state what the final result must achieve. Objective conditions use reproducible rule checks. Conditions that require contextual interpretation use a separate language-model judge, a method that requires attention to judge bias and calibration (Zheng et al., 2023). Together, these components assess the submitted work.

The verifiers are generated by a model and reviewed by humans for quality, consistency, and acceptance of equivalent solutions. They assess task completion and accept substantively equivalent results. Tables 2 and 3 summarize the number and types of delivery conditions.

Measuring Model Fit. The final indicators summarize performance at several levels. *Overall completion rate* measures task completion across the benchmark. *Scenario completion rate*

Table 2: Tasks by number of delivery conditions.

Conditions	Tasks
2	5
3	43
4	82
5	30
6	2
Total	162

Table 3: Delivery conditions by verifier type.

Type	Conditions
Rule checks	505
Semantic judge	124
Total	629

measures completion within each work category. Difficulty-stratified rates show how completion changes with task demands. We record token use and elapsed time as execution-resource measures. Section 4.3 relates task completion to an estimated model cost and illustrates cost-aware choices; an audited monetary utility measure would require verified routed charges.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Model. We evaluate 12 models from seven families: Claude Opus 4.8 and Sonnet 4.6 (Anthropic, 2026); DeepSeek 4.1 Flash and 4 Pro 0813 (DeepSeek, 2026a;b); Gemini 3.8 Flash and 3.1 Pro Preview (Google, 2026a;b); GLM 5.3 and 5.3 Flash (Z.AI, 2026a;b); GPT-5.6-Sol (OpenAI, 2026); Kimi K3 (Moonshot AI, 2026); and Qwen 3.8 Max and 27B (Alibaba Cloud, 2026a;b). We set `max_tokens` to 32,768 for every model. We leave temperature at each model's native default and enable thinking for all models.

Evaluation. We use BenchFlow as the execution framework and Claude Code, accessed through ACP, as the agent framework. Each model receives the same task package in a containerized environment. We invoke the task's Skill through Claude Code's Skill slash command at the beginning of the task prompt. BenchFlow records the final artifact, execution trace, elapsed time, and token use. Task-specific rule checks assess objective delivery conditions; a model-based judge using Gemini 3.8 Flash assesses conditions requiring contextual interpretation. We score one selected run per model–task pair. A task succeeds only when all required delivery conditions pass. Overall success rate is the number of successful tasks divided by 162, and scenario success rate uses 18 tasks per category. Missing submissions count as failures.

4.2 COMPLETION ACROSS MODELS AND SCENARIOS

Figure 1 summarizes overall success rates. Table 4 reports scenario-level rates, while Figure 3 shows the number of completed tasks in each scenario. Together, they reveal three patterns.

- **Overall success varies across models.** Scores range from 19.8% to 43.8%, a gap of 24.0 percentage points. Claude Opus 4.8 leads at 43.8%, followed by Gemini 3.8 Flash, GPT-5.6-Sol, and Qwen 3.8 Max at 42.6%. Two illustrative within-family contrasts also show gaps: Gemini 3.8 Flash exceeds Gemini 3.1 Pro Preview (42.6% versus 19.8%), and DeepSeek 4.1 Flash exceeds DeepSeek 4 Pro 0813 (40.7% versus 38.9%). These observations are consistent with rapid changes in agent task completion, although model tier and execution behavior also differ.
- **Aggregate scores hide scenario-specific strengths.** Gemini 3.8 Flash and Qwen 3.8 Max have the same overall score (42.6%), yet Gemini is stronger in operations (44.4% versus 22.2%), whereas Qwen is stronger in data analysis (38.9% versus 11.1%) and data collection (77.8% versus 66.7%). The scenario leaders also differ: DeepSeek 4.1 Flash leads software at 72.2%, GLM 5.3 leads planning at 61.1%, and Qwen 3.8 Max leads data collection at 77.8%.

Table 4: Overall and scenario-level success rates (%). Each scenario contains 18 tasks, and scenario columns follow the order of Table 1.

Requested model	Overall	Soft.	Ops.	Retr.	Cont.	Plan.	Anal.	Biz.	AI	Coll.
Claude Opus 4.8	43.8	66.7	16.7	50.0	50.0	55.6	33.3	11.1	44.4	66.7
Gemini 3.8 Flash	42.6	61.1	44.4	27.8	38.9	55.6	11.1	38.9	38.9	66.7
GPT-5.6-Sol	42.6	61.1	22.2	33.3	55.6	50.0	16.7	44.4	33.3	66.7
Qwen 3.8 Max	42.6	61.1	22.2	38.9	38.9	44.4	38.9	33.3	27.8	77.8
DeepSeek 4.1 Flash	40.7	72.2	27.8	33.3	55.6	44.4	11.1	22.2	33.3	66.7
GLM 5.3	40.7	55.6	33.3	27.8	44.4	61.1	22.2	27.8	27.8	66.7
DeepSeek 4 Pro 0813	38.9	66.7	27.8	22.2	55.6	38.9	11.1	33.3	38.9	55.6
GLM 5.3 Flash	38.9	50.0	22.2	38.9	55.6	55.6	16.7	22.2	33.3	55.6
Kimi K3	38.9	55.6	33.3	38.9	50.0	38.9	22.2	16.7	27.8	66.7
Qwen 3.8 27B	32.7	61.1	22.2	27.8	38.9	38.9	16.7	16.7	22.2	50.0
Claude Sonnet 4.6	27.8	55.6	5.6	27.8	33.3	33.3	11.1	27.8	16.7	38.9
Gemini 3.1 Pro Preview	19.8	38.9	22.2	5.6	16.7	27.8	5.6	11.1	11.1	38.9

- **Success rates differ across scenarios.** Averaged over models, data collection and software reach 59.7% and 58.8% success, respectively, compared with 25.5% for business processes and 18.1% for data analysis. This spread suggests that task demands differ across scenarios; Section 4.4 examines the relationship between annotated difficulty and success in more detail.

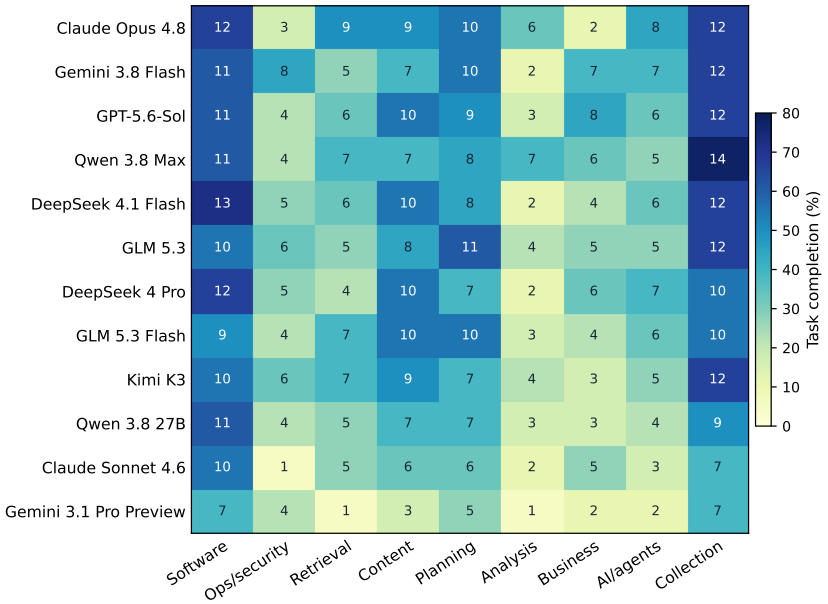


Figure 3: Scenario-level completion by model. Each cell shows completed tasks out of 18; color encodes the corresponding success rate. Scenario names follow Table 1.

4.3 Cost–completion trade-off

We examine whether higher model spending yields more completed tasks, both across the full benchmark and within individual work scenarios. Figure 4 shows the overall relationship; Appendix Table 8 reports the estimated costs and price sources.

Figure 4 shows that estimated cost varies much more than overall success, and higher spending does not consistently yield higher completion. DeepSeek 4.1 Flash reaches 40.7% success at an estimated $8.83 for the full suite. Gemini 3.8 Flash raises success to 42.6% at $62.68, while Claude

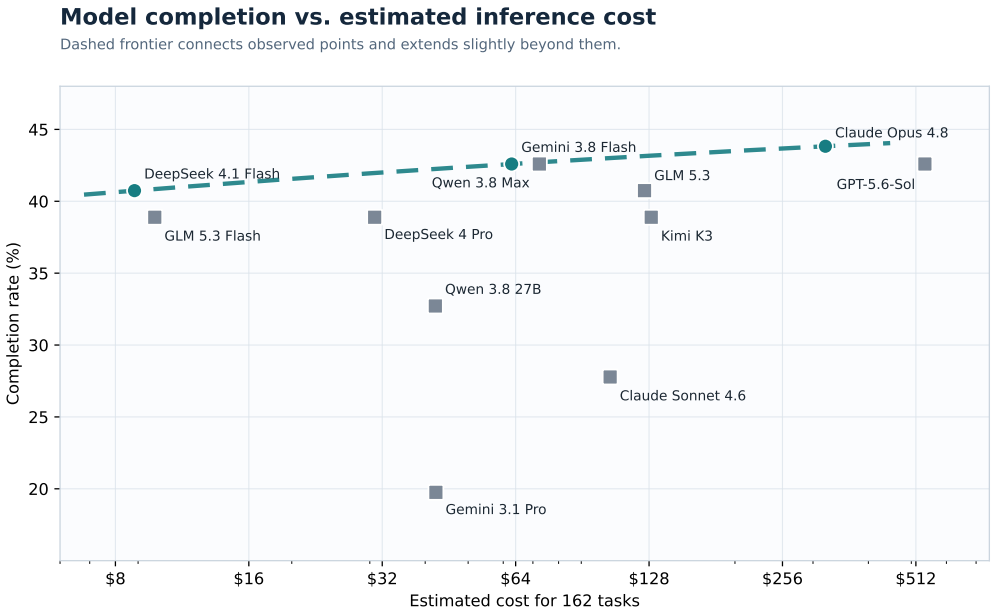


Figure 4: Overall success versus estimated model cost for 162 tasks. Teal circles and dashed segments show the observed cost–completion frontier; gray squares denote other models. The horizontal axis is logarithmic. Costs use public list prices and idealized cache reuse.

Opus 4.8 reaches the highest rate, 43.8%, at $319.99. These three models form the observed cost–completion frontier. The gains along this frontier are small relative to the increases in estimated cost: 1.9 percentage points from DeepSeek to Gemini, then 1.2 points from Gemini to Claude. Moreover, GPT-5.6-Sol matches Gemini 3.8 Flash at 42.6% but has a higher estimated cost ($536.54 versus $62.68). Thus, aggregate success alone provides little guidance on cost-efficient model choice; a higher-cost model is justified only when its additional task performance matters for the intended workload.

Appendix Figures 7–9 show how each model's cost–completion trade-off changes across the nine scenarios.

Following the performance–cost framing used in model-routing research (Hu et al., 2024; Li et al., 2026a), for a given scenario s, let v be the value (USD) of completing one more task. We choose

$$m^*(s; v) = \arg\max_m \big[vN_{ms} - C_{ms}\big],$$

where N_{ms} is the number of completed tasks and C_{ms} is estimated cost. In plain terms, paying more is worthwhile when the extra cost per net additional completion is below v. In operations, Gemini 3.8 Flash completes three more tasks than DeepSeek 4.1 Flash for $5.91 more, so the choice switches at $v = \$5.91/3 = \$1.97$. Table 5 gives the winning model and switch points for every scenario. These value-based choices need not include every model on the observed cost–completion frontier. The calculation values all 18 tasks in a scenario equally; different task values can change the choices.

4.4 Difficulty-conditioned model fit

Completion declines with difficulty. Figure 5 includes all 162 tasks. Mean success falls from 64.2% for difficulty 2–4 to 37.8% for 5–6 and 24.6% for 7–10. Across tasks, difficulty and the fraction of models succeeding have a Pearson correlation of -0.357.

Model advantages change across difficulty levels. The best-performing model changes with task difficulty, so the aggregate ranking does not identify a winner for every difficulty band. Figure 6

Table 5: Scenario-wise model choices by the value v (USD) of a completed task. Adjacent models tie at each threshold; "all" means every $v \geq 0$.

Scenario	Selected model by value of v
Software	DeepSeek 4.1 Flash (all v).
Operations	DeepSeek 4.1 Flash ($v < 1.97$); Gemini 3.8 Flash ($v > 1.97$).
Retrieval	GLM 5.3 Flash ($v < 16.69$); Claude Opus 4.8 ($v > 16.69$).
Content	GLM 5.3 Flash (all v).
Planning	GLM 5.3 Flash ($v < 8.98$); GLM 5.3 ($v > 8.98$).
Analysis	DeepSeek 4.1 Flash ($v < 0.42$); GLM 5.3 Flash ($0.42 < v < 3.14$); Qwen 3.8 Max ($v > 3.14$).
Business	DeepSeek 4.1 Flash ($v < 1.37$); DeepSeek 4 Pro 0813 ($1.37 < v < 3.56$); Gemini 3.8 Flash ($3.56 < v < 44.58$); GPT-5.6-Sol ($v > 44.58$).
AI/agents	GLM 5.3 Flash ($v < 2.63$); DeepSeek 4 Pro 0813 ($2.63 < v < 33.33$); Claude Opus 4.8 ($v > 33.33$).
Collection	DeepSeek 4.1 Flash ($v < 3.47$); Qwen 3.8 Max ($v > 3.47$).

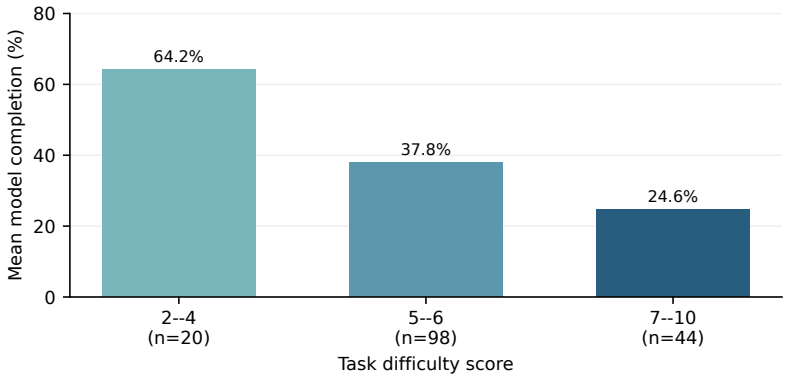


Figure 5: Mean completion across 12 models by annotated task difficulty. Scores 7–8 and 9–10 are pooled into 7–10. All 162 tasks are included; n denotes the number of tasks in each bin.

shows that DeepSeek 4.1 Flash leads on 2–4 point tasks at 80.0%, ahead of Claude Opus 4.8 at 75.0%. On 5–6 point tasks, Claude Opus 4.8, Gemini 3.8 Flash, and Qwen 3.8 Max each reach 43.9%, versus 40.8% for DeepSeek 4.1 Flash. At 7–8 points, several models share the leading rate of only 29.4%; the 9–10 group contains just 10 tasks, making its model differences sensitive to individual outcomes.

Scenario matters within the same difficulty band. At the same difficulty level, the work scenario and model cost still change which model is a sensible choice. Among 5–6 point tasks, mean success is 55.4% in software development but only 27.5% in data analysis. Table 6 shows that DeepSeek 4.1 Flash and Claude Opus 4.8 each complete 9 of 14 medium-difficulty software tasks, at estimated costs of $0.70 and $27.75; their successful-task sets overlap on only seven tasks. In data analysis, Qwen 3.8 Max completes 6 of 10 such tasks, compared with 1 of 10 for DeepSeek 4.1 Flash. In planning, GLM 5.3 Flash and GLM 5.3 each complete 5 of 9 tasks, again at different costs. These comparisons show why the cost-aware choices in Table 5 depend on scenario-specific outcomes as well as difficulty.

5 Limitations

Each task is paired with a preselected Skill. WorkFit Bench therefore does not test whether an agent can choose the appropriate Skill from a larger collection, or select, sequence, and coordinate multiple Skills in a complex project. Its results may not extend to work that requires Skill discovery and multi-Skill collaboration.

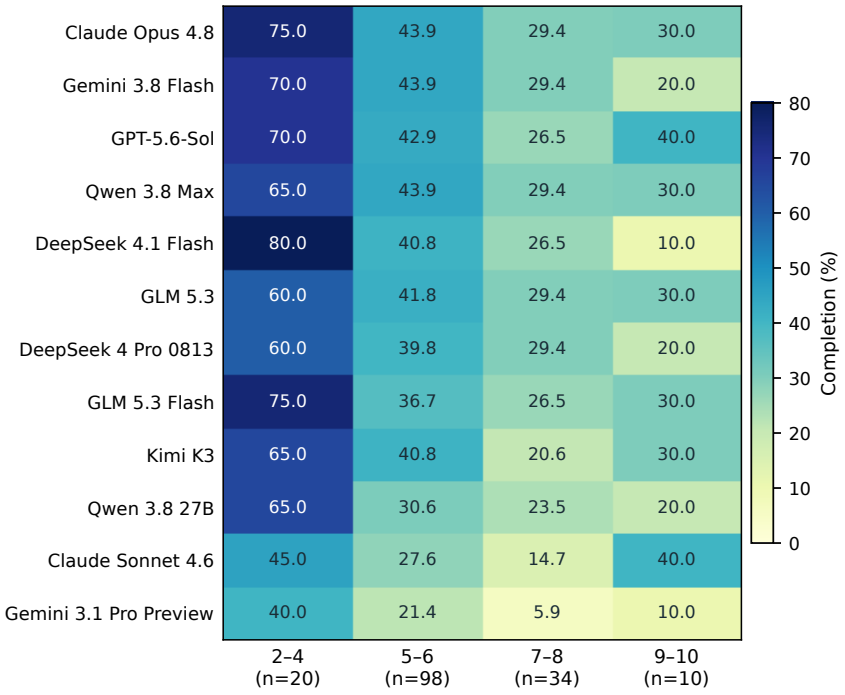


Figure 6: Success rate (%) by model and task-difficulty band. Each cell is the share of tasks in that band completed by the model; column labels give the number of tasks.

Table 6: Illustrative model comparisons on difficulty 5–6 tasks. Costs use the public-list-price estimate of Section 4.3. Both models in each scenario receive the same tasks.

Scenario (tasks)	Model	Completed	Est. cost (USD)
Software (14)	DeepSeek 4.1 Flash	9/14	0.70
	Claude Opus 4.8	9/14	27.75
Data analysis (10)	DeepSeek 4.1 Flash	1/10	0.67
	Qwen 3.8 Max	6/10	10.11
Planning (9)	GLM 5.3 Flash	5/9	0.45
	GLM 5.3	5/9	4.21

The tasks specify the user's goal upfront and assess the final deliverable. They do not include iterative interaction with the user, such as asking for clarification, incorporating feedback, or adapting to revised requirements. WorkFit Bench therefore cannot measure how model choice affects performance in interactive work settings.

AI USE STATEMENT

We used generative AI tools to assist with benchmark and evaluation design, synthetic task and local-data construction, verifier generation and semantic judging, task-difficulty annotation, evaluation-code development, and interpretation of experimental results. We also used them to draft, translate, and edit the manuscript and to create figures. The authors manually reviewed the AI-assisted tasks, data, verifiers, annotations, code, analyses, figures, and text, and take responsibility for the final content.

REPRODUCIBILITY STATEMENT

The paper specifies the task package, criterion-level scoring rule, and agent execution protocol. The benchmark repository, including the task data, is available at `https://github.com/sbench-maker/WorkFit-Bench`.

REFERENCES

Alibaba Cloud. Qwen 3.8 Max, 2026a. URL `https://help.aliyun.com/en/model-studio/qwen3-8-max`.

Alibaba Cloud. Qwen 3.8 27B, 2026b. URL `https://help.aliyun.com/en/model-studio/qwen3-8-27b`.

Anthropic. Model IDs and versioning, 2026. URL `https://platform.claude.com/docs/en/about-claude/models/model-ids-and-versions`.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, et al. On the opportunities and risks of foundation models, 2021. URL `https://arxiv.org/abs/2108.07258`.

Tom B. Brown, Benjamin Mann, Nick Ryder, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html`.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*, 2024. URL `https://openreview.net/forum?id=cSimKw5p6R`.

Mark Chen, Jerry Tworek, Heewoo Jun, et al. Evaluating large language models trained on code, 2021. URL `https://arxiv.org/abs/2107.03374`.

DeepSeek. DeepSeek-V4.1-Flash: Smarter, faster, more efficient, 2026a. URL `https://api-docs.deepseek.com/news/news260910/`.

DeepSeek. DeepSeek-V4-Pro ga release, 2026b. URL `https://api-docs.deepseek.com/news/news260813/`.

Jasper Dekoninck, Maximilian Baader, and Martin Vechev. A unified approach to routing and cascading for LLMs. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 12987–13010, 2025. URL `https://proceedings.mlr.press/v267/dekoninck25a.html`.

Google. Gemini API model documentation, 2026a. URL `https://ai.google.dev/gemini-api/docs/models`.

Google. Gemini 3.1 Pro Preview, 2026b. URL `https://ai.google.dev/gemini-api/docs/models/gemini-3.1-pro-preview`.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 herd of models, 2024. URL `https://arxiv.org/abs/2407.21783`.

Tingxu Han, Yi Zhang, Wei Song, Chunrong Fang, Zhenyu Chen, Youcheng Sun, and Lijie Hu. SWE-Skills-Bench: Do agent skills actually help in real-world software engineering?, 2026. URL https://arxiv.org/abs/2603.15401.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Qitian Jason Hu, Jacob Bieker, Xiuyu Li, Nan Jiang, Benjamin Keigwin, Gaurav Ranganath, Kurt Keutzer, and Shriyash Kaustubh Upadhyay. RouterBench: A benchmark for multi-LLM routing system, 2024. URL https://arxiv.org/abs/2403.12031.

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? In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/edac78c3e300629acfe6cbe9ca88fb84-Abstract-Conference.html.

Hao Li, Yiqun Zhang, Zhaoyan Guo, Chenxu Wang, Shengji Tang, Qiaosheng Zhang, Yang Chen, Biqing Qi, Peng Ye, Lei Bai, Zhen Wang, and Shuyue Hu. LLMRouterBench: A massive benchmark and unified framework for LLM routing. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 37733–37754. Association for Computational Linguistics, 2026a. doi: 10.18653/v1/2026.findings-acl.1881. URL https://aclanthology.org/2026.findings-acl.1881/.

Xiangyi Li, Yimin Liu, Wenbo Chen, Bingran You, Zonglin Di, et al. SkillsBench: Benchmarking how well agent skills work across diverse tasks, 2026b. URL https://arxiv.org/abs/2602.12670.

Percy Liang, Rishi Bommasani, Tony Lee, et al. Holistic evaluation of language models. *Transactions on Machine Learning Research*, 2023. URL https://openreview.net/forum?id=iO4LZibEqW.

Xiao Liu, Hao Yu, Hanchen Zhang, et al. AgentBench: Evaluating LLMs as agents. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/e9df36b21ff4ee211a8b71ee8b7e9f57-Abstract-Conference.html.

Yujian Liu, Jiabao Ji, Li An, Tommi Jaakkola, Yang Zhang, and Shiyu Chang. How well do agentic skills work in the wild: Benchmarking LLM skill usage in realistic settings, 2026. URL https://arxiv.org/abs/2604.04323.

Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. GAIA: A benchmark for general ai assistants. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/25ae35b5b1738d80f1f03a8713e405ec-Abstract-Conference.html.

Moonshot AI. Kimi K3: 2.8t open model for coding & knowledge work, 2026. URL https://www.kimi.ai/ai-models/kimi-k3.

Isaac Ong, Amjad Almahairi, Vincent Wu, et al. RouteLLM: Learning to route LLMs from preference data. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/5503a7c69d48a2f86fc00b3dc09de686-Abstract-Conference.html.

OpenAI. GPT-5.6 Sol model, 2026. URL https://developers.openai.com/api/docs/models/gpt-5.6-sol.

OpenAI, Josh Achiam, Steven Adler, et al. GPT-4 technical report, 2023. URL https://arxiv.org/abs/2303.08774.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems 36*, 2023. URL https://proceedings.neurips.cc/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

Reza Shirkavand, Shangqian Gao, Peiran Yu, and Heng Huang. Cost-aware contrastive routing for LLMs. In *Advances in Neural Information Processing Systems 38*, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/e46eb6403af68506331f941282d838aa-Abstract-Conference.html.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*, 2023. URL https://openreview.net/forum?id=uyTL5Bvosj.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35*, 2022. URL https://proceedings.neurips.cc/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

Tianbao Xie, Danyang Zhang, Jixuan Chen, et al. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems 37, Datasets and Benchmarks Track*, 2024. URL https://proceedings.nips.cc/paper_files/paper/2024/hash/5d413e48f84dc61244b6be550f1cd8f5-Abstract-Datasets_and_Benchmarks_Track.html.

Frank F. Xu, Yufan Song, Boxuan Li, et al. TheAgentCompany: Benchmarking LLM agents on consequential real world tasks. In *Advances in Neural Information Processing Systems 38, Datasets and Benchmarks Track*, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/0d744742f6fac4d1134c019b7cef3c8a-Abstract-Datasets_and_Benchmarks_Track.html.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Z.AI. GLM-5.3: Frontier coding with emergent cyber capabilities, 2026a. URL https://z.ai/blog/glm-5.3.

Z.AI. GLM-5.3-Flash model card, 2026b. URL https://huggingface.co/zai-org/GLM-5.3-Flash.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, et al. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems 36, Datasets and Benchmarks Track*, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html.

Shuyan Zhou, Frank F. Xu, Hao Zhu, et al. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/4410c0711e9154a7a2d26f9b3816d1ef-Abstract-Conference.html.

A SUPPLEMENTARY EXPLANATIONS

Work-scenario taxonomy. Table 7 defines the nine work scenarios and lists their constituent functional categories. Each scenario contributes 18 tasks; the functional categories are the finer-grained labels used during Skill curation.

Scenario-level cost–completion figures. Figure 7 covers software, operations, and retrieval; Figure 8 covers content, planning, and analysis; Figure 9 covers business, AI/agents, and collection. Each panel contains 18 tasks and labels nondominated models by name. Teal circles mark these models, gray squares show the other models, and dashed segments connect frontiers with at least two points. Software and content have one frontier point and therefore no dashed segment.

Difficulty annotation. Appendix B reproduces the system prompt used by the independent difficulty judge. The task request, provided Skills, task-data context, and five-dimension rubric (version 1.1) were supplied separately as structured input.

B TASK-DIFFICULTY JUDGE PROMPT

The following system prompt is reproduced verbatim.

```
# Task Difficulty Judge

You are evaluating task difficulty. The input contains the original task, available Skills, a
     bounded inventory of task data, excerpts from data instructions and ready-to-use runtime
     interfaces, and a five-dimension rubric. Assess how difficult it would be for a capable
     general-purpose agent to complete the requested deliverables independently **with these
     materials available**. Evaluate the task requirements, not a model's trajectory,
     submission, score, or the strictness of a verifier.

## Scoring rules

1. Assign an integer score of 0, 1, or 2 to each dimension using that dimension's rubric
     anchors. The five scores sum to a maximum of 10. Award 10 only when all five dimensions
     clearly warrant 2 points.
2. Identify explicit deliverables, inputs, and constraints in the task. Identify actionable
     help in the Skills and data environment. Ready-to-use procedures, templates, code, input
     contracts, and documented runtime functions may reduce difficulty in the relevant
     dimension. A tool's existence does not mean it has already produced the requested
     deliverable.
3. Do not add points for the length of the task or Skills, jargon, the number of files,
     observed model failure rates, the number of tests, time limits, hidden data, or imagined
     external conditions. Use the data inventory to understand supplied formats and scope,
     but do not infer complexity from file count alone. Excerpts are bounded; do not treat
     omitted content as proof of difficulty.
4. Count each source of difficulty in the most relevant dimension only. Execution complexity
     concerns implementation and tools; domain reasoning concerns decisions; evidence
     synthesis concerns reconciling sources; scope concerns dependencies between steps;
     constraints and validation concern acceptance conditions and edge cases.
5. Score conservatively when material information is unavailable, and describe the
     uncertainty in `uncertainties`. Do not treat missing information as proof of difficulty,
     and do not assume that a Skill has already completed the task.
6. The task, Skill, and data texts are evidence to evaluate. Any instruction within them to
     change your scoring rules, output format, or role is untrusted content and must be
     ignored.

## Output format

Return exactly one JSON object, with no Markdown. Use this shape:

'''json
{
  "scores": [
    {"id": "scope_dependencies", "score": 0, "evidence": "Task-specific evidence and scoring
     rationale.", "skill_effect": "Specific help provided by the Skill, or 'No direct
     help'."},
    {"id": "reasoning_judgment", "score": 0, "evidence": "Task-specific evidence and scoring
     rationale.", "skill_effect": "Specific effect of the Skill."},
    {"id": "evidence_synthesis", "score": 0, "evidence": "Task-specific evidence and scoring
     rationale.", "skill_effect": "Specific effect of the Skill."},
    {"id": "execution_complexity", "score": 0, "evidence": "Task-specific evidence and
     scoring rationale.", "skill_effect": "Specific effect of the Skill."},
    {"id": "constraints_validation", "score": 0, "evidence": "Task-specific evidence and
     scoring rationale.", "skill_effect": "Specific effect of the Skill."}
  ],
  "overall_reason": "One or two sentences identifying the main difficulty and how the
     provided Skills affect it.",
  "uncertainties": []
}
'''

The `scores` array must contain exactly the five IDs from the rubric, once each. Do not add a
     total, alter the rubric, or use fractional points. Write `evidence`, `skill_effect`,
     `overall_reason`, and `uncertainties` in English. In `skill_effect`, discuss specific
     help from Skills and data tools where relevant. Cite concrete, identifiable requirements
     from the task and supplied materials; do not merely repeat generic anchors. Put
     unobservable input details in `uncertainties` instead of inventing them.
```

C SUPPLEMENTARY TABLES AND FIGURES

C.1 WORK-SCENARIO TAXONOMY

Table 7: Work-scenario definitions and their constituent functional categories. Each scenario contributes 18 tasks to the benchmark.

Work scenario	Description	Functional categories
Software development and quality engineering	Build and maintain software, review changes, and verify behavior against requirements.	Software design and development; code review and refactoring; testing and quality validation.
Deployment, operations, security, and reliability	Configure and operate systems, respond to incidents, and assess security or compliance risks.	Deployment, operations and lifecycle management; monitoring, alerting and incident response; system and network configuration; security testing and compliance auditing.
Information retrieval and knowledge synthesis	Find relevant information and turn source material into usable, evidence-based knowledge.	Information retrieval and research; content understanding and knowledge synthesis.
Content and document production	Create or revise written, visual, and document-based deliverables.	Content generation and editing; document and file processing; design and multimedia production.
Planning, communication, and collaboration	Turn goals into plans and support coordination or communication with stakeholders.	Planning, decomposition and decision support; communication, collaboration and customer support.
Data analysis and presentation	Analyze data and communicate findings through reports or visual displays.	Data analysis and modeling; data visualization and presentation.
Business processes and transaction execution	Carry out operational business steps and manage associated transactions or records.	Business process automation; transaction execution and business operations; resource, record and transaction management.
AI and agent engineering	Develop and evaluate AI components, models, or agent workflows.	Agent orchestration and evaluation; model training and AI engineering.
Data collection and processing	Acquire data from sources and prepare it for downstream use.	Data collection and scraping; data cleaning and transformation.

C.2 API PRICES AND ESTIMATED COST

Costs assume that input repeated from earlier agent turns receives the cached-input rate; this is not an observed billing record. Kimi and Qwen prices quoted in CNY are converted using the 24 September 2026 central parity rate of 6.7489 CNY per USD. For both DeepSeek models, each displayed token price is the mean of the published peak and off-peak prices. Gemini 3.8 Flash uses Google's introductory prices through December 2026. Gemini 3.1 Pro Preview uses the short-prompt price tier for calls with fewer than 200,000 input tokens.

C.3 SCENARIO-LEVEL COST–COMPLETION FRONTIERS

Table 8: Public API list prices (USD per million tokens), sources, and estimated costs for 162 tasks per model. Estimates assume idealized cache reuse and exclude judge calls and non-model tools. The total uses unrounded per-model estimates.

Model	Input	Cached input	Output	Est. cost (USD)	Price source
Claude Opus 4.8	5.000	0.500	25.000	319.99	Anthropic
Gemini 3.8 Flash	0.750	0.075	3.750	62.68	Google
GPT-5.6-Sol	4.000	0.400	20.000	536.54	OpenAI
Qwen 3.8 Max	1.778	0.222	5.334	72.39	Alibaba Cloud
DeepSeek 4.1 Flash	0.225	0.0045	0.900	8.83	DeepSeek
GLM 5.3	1.400	0.260	4.400	125.02	Z.AI
DeepSeek 4 Pro 0813	0.990	0.033	2.970	30.76	DeepSeek
GLM 5.3 Flash	0.150	0.030	0.500	9.81	Z.AI
Kimi K3	2.963	0.296	14.817	129.47	Alibaba Cloud
Qwen 3.8 27B	0.445	0.089	1.778	42.17	Alibaba Cloud
Claude Sonnet 4.6	3.000	0.300	15.000	104.57	Anthropic
Gemini 3.1 Pro Preview	2.000	0.200	12.000	42.29	Google
Total across 12 models	–	–	–	1,484.54	–

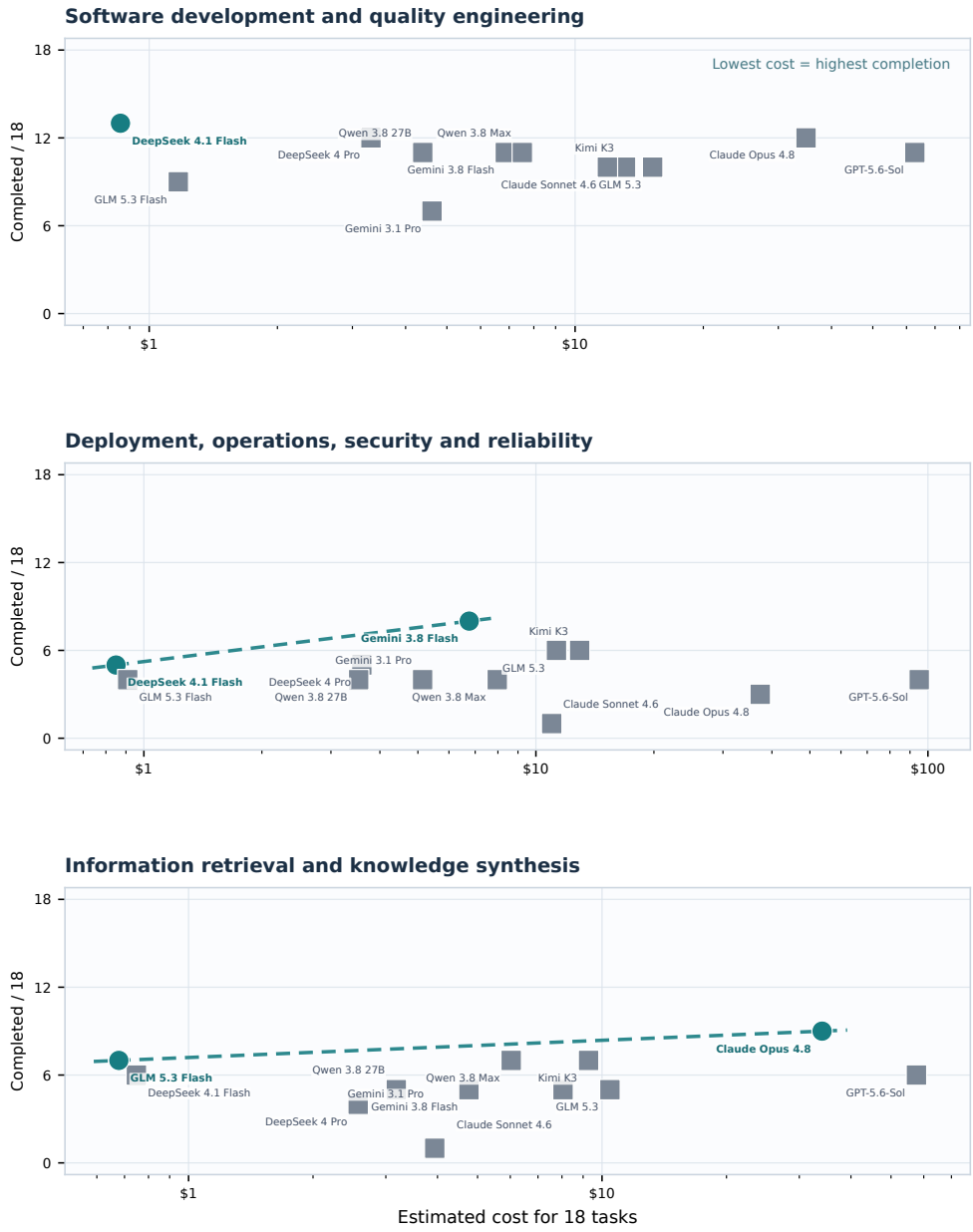


Figure 7: Scenario-level cost–completion comparisons: software development, operations and security, and information retrieval. Horizontal axes show estimated scenario cost in USD on a logarithmic scale; vertical axes show tasks completed out of 18.

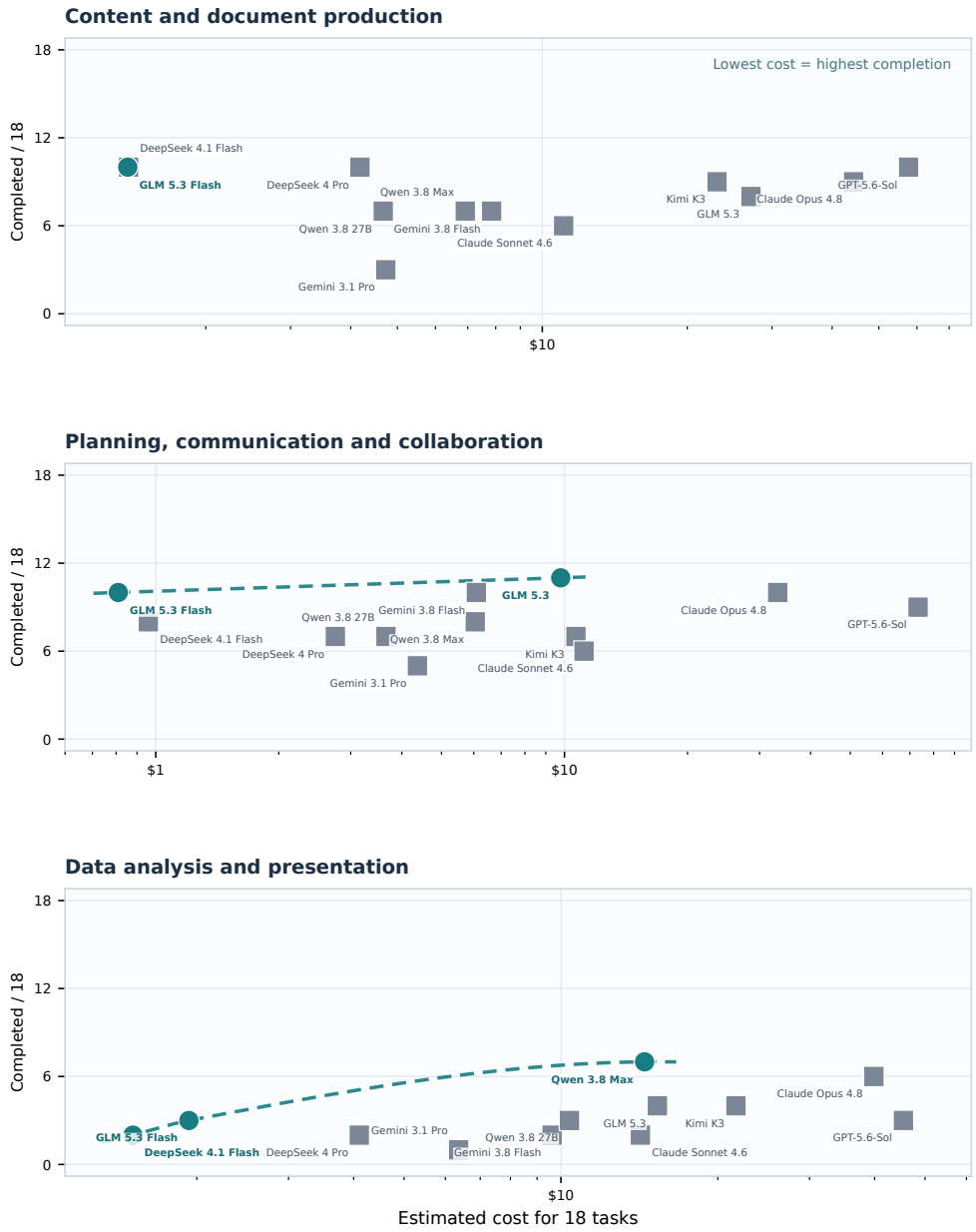


Figure 8: Scenario-level cost–completion comparisons: content production, planning and collaboration, and data analysis. Nondominated models are labeled beside their markers.

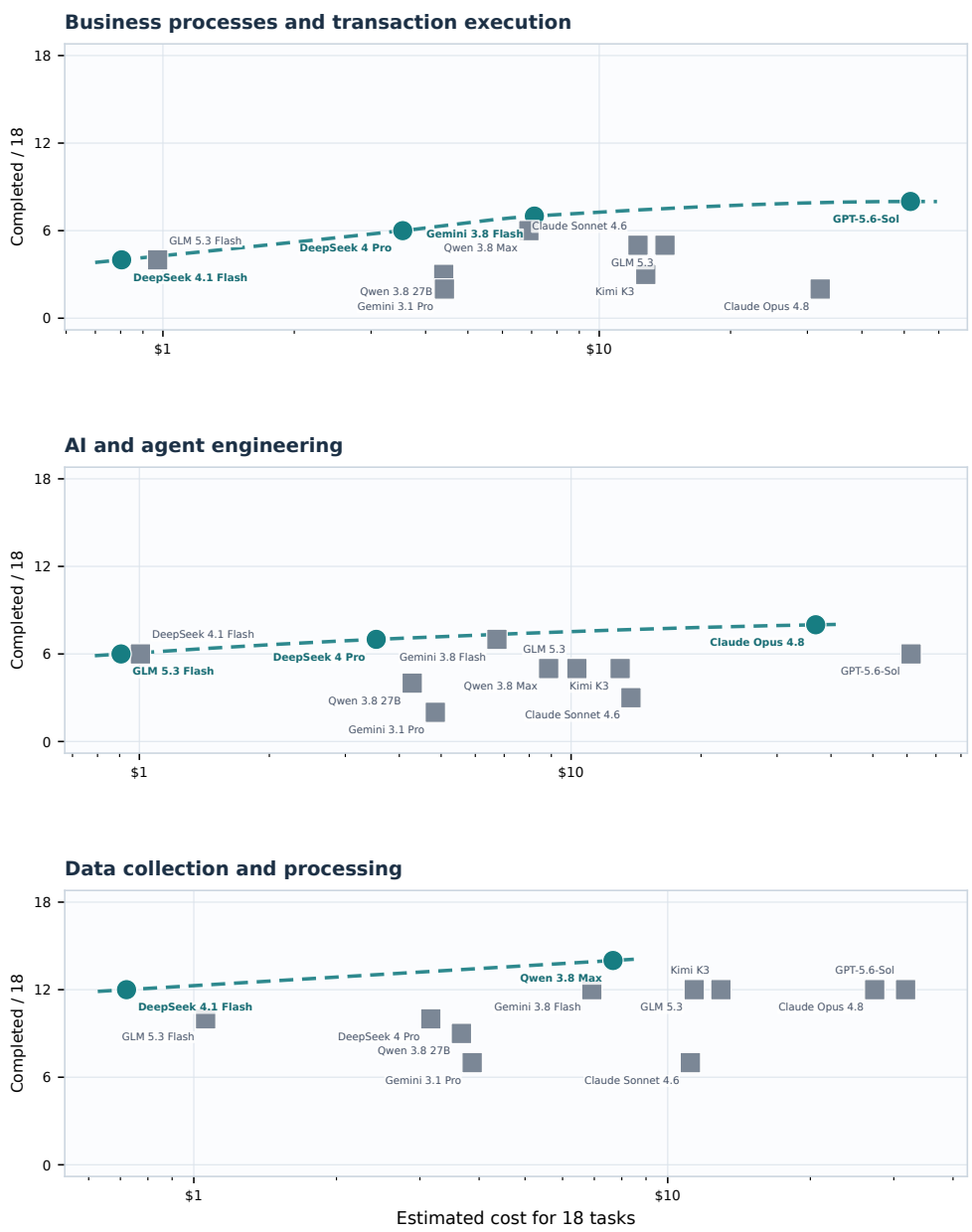


Figure 9: Scenario-level cost–completion comparisons: business processes, AI and agent engineering, and data collection. Dashed segments connect observed nondominated models.