

DAREBench: Deployment-Aware and Reliable Evaluation of Models as Agents

Yu Liu^{1,2,3*†}, Zhilin Liu^{3*}, Zhiwei Yang^{1,2,3*}, Shaojie Zhang^{3*}, Zheyuan Deng⁴,
Tingwei Huang³, Zhenbo Luo³, Lei Jiang^{1,2}, Yanbing Liu², Pei Fu^{3†}

¹Institute of Information Engineering, Chinese Academy of Sciences

²School of Cyber Security, University of Chinese Academy of Sciences

³MiLM Plus, Xiaomi Inc.

⁴Department of Computer Science, Brown University

Abstract

As large language models evolve from question-answering systems into general-purpose agents, evaluation must move beyond static answer correctness to assess multimodal perception, multi-step execution, tool use, and artifact delivery. However, existing benchmarks are often tied to specific task types, execution environments, or scoring protocols, limiting their comparability, interpretability, and reliability for deployment decisions. We introduce DAREBench (Deployment-Aware and Reliable Evaluation of Models as Agents), a benchmark designed to capture workload variation and support reliable agent evaluation. Built on a shared OpenClaw execution environment, DAREBench organizes 233 tasks selected and adapted from 22 source benchmarks into a 2×3 workload matrix defined by input modality and execution form, and evaluates them under a unified contract-based protocol with evidence-based score auditing. We evaluate 23 commercial API models and 12 locally deployed open-weight models over 7,587 model-task runs, reporting accuracy and token consumption alongside reference costs for API models. Results show that no single model dominates all workload groups, text and multimodal tasks exhibit distinct accuracy-cost trade-offs, and local open-weight models are competitive in several groups but still trail frontier commercial models overall. These findings suggest that agent deployment and model selection should consider workload profiles, deployment mode, and accuracy-cost trade-offs rather than rely on a single aggregate score. **The benchmark details can be found at <https://github.com/SeerRay-Lab/DAREBench>.**

1 Introduction

Frontier large language models (LLMs) are evolving from standalone question-answering systems into core decision-making components of general-purpose agent systems. These agents must perceive, reason, plan, use tools, and deliver artifacts in dynamic workspaces, rather than merely generate an answer string (Wei et al. 2022; Schick et al. 2023). Accordingly, model evaluation is shifting from static input-output performance to end-to-end execution under constraints on tools, context, time, and cost.

However, existing evaluation frameworks do not adequately support this transition. Traditional knowledge, reasoning, and long-context benchmarks primarily assess answer correctness under static inputs (Hendrycks et al. 2021;

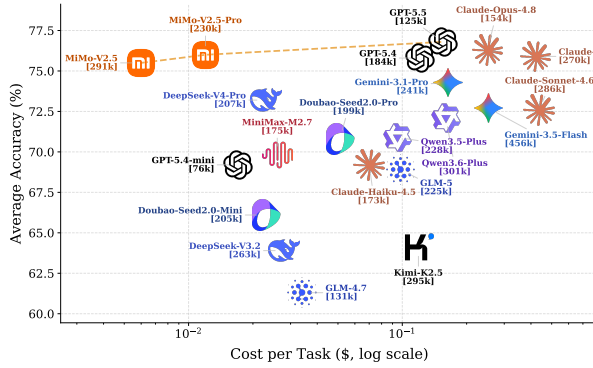
Rein et al. 2024; Bai et al. 2025c). Multimodal benchmarks incorporate visual inputs but still largely follow a static input-output paradigm (Yue et al. 2024; Wang et al. 2024). Tool-use and end-to-end agent benchmarks, by contrast, are often confined to specific command-line, web, or desktop environments (Merrill et al. 2026; Xie et al. 2024; Mialon et al. 2024; Liu et al. 2024; Zhou et al. 2024; Yao et al. 2025). Moreover, these benchmarks employ different runtimes, tool interfaces, task protocols, and scoring criteria, making their results difficult to compare directly. Recent studies have improved evaluation observability through execution trajectories, environment snapshots, and audit logs (Ye et al. 2026; Li et al. 2026a). Nevertheless, these studies still primarily report overall success rates within specific environments. They therefore provide limited insight into model behavior across modalities, execution processes, and degrees of tool dependence. As a result, they offer limited support for comparable, interpretable, and reliable model selection in practice.

We argue that agent evaluation can inform deployment decisions only by addressing three interrelated questions, rather than by simply increasing benchmark scale. First, **are the results interpretable?** Agent tasks differ substantially in input modality, execution process, and tool dependence. A single aggregate score can obscure model strengths and weaknesses across workloads. Second, **are comparisons attributable?** When tasks rely on different agent frameworks, tool interfaces, and execution protocols, observed differences conflate model capability with runtime compatibility. Third, **are the scores trustworthy?** Open-ended agent tasks often involve multi-step operations and artifact delivery. Matching only the final text cannot verify whether a task was actually completed. Moreover, an LLM judge may incorrectly award positive credit when tools were not executed, the task timed out, or required artifacts were missing. Together, these issues call for a deployment-oriented benchmark that unifies **workload organization, execution environment control, and outcome verification**.

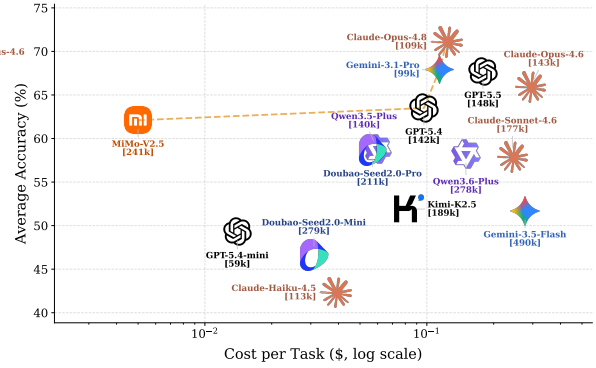
To address these requirements, we propose DAREBench (Deployment-Aware and Reliable Evaluation of Models as Agents), a benchmark for deployment-aware and reliable agent evaluation. To improve workload-level interpretability, we select and adapt 233 tasks from 22 existing benchmarks through a three-stage curation process based on measura-

*These authors contributed equally.

†Corresponding author.



(a) Text Modality: Performance vs. Cost



(b) Multi-Modal: Performance vs. Cost

Figure 1: Accuracy–cost trade-offs for models on (a) text and (b) multimodal tasks. Labels show average tokens per task; cost uses a logarithmic scale. Dashed lines mark the Pareto frontiers: MiMo-V2.5, MiMo-V2.5-Pro, and GPT-5.5 in (a), and MiMo-V2.5, GPT-5.4, Gemini-3.1-Pro, and Claude-Opus-4.8 in (b).

bility, difficulty, and representativeness. We organize these tasks into a 2×3 workload matrix defined by input modality and execution form, covering text and multimodal inputs as well as single-step, multi-step, and tool-mediated multi-step tasks. To improve the attribution of cross-benchmark comparisons, we build a shared execution environment on OpenClaw (Steinberger and OpenClaw Contributors 2026). All models operate under a common agent loop, tool interface, workspace abstraction, and logging protocol, thereby reducing the influence of heterogeneous agent scaffolds. For open-ended tasks, we define a task contract specifying the initial workspace, available tools, target artifacts, scorer, and time budget. Task outcomes are evaluated through automated checks, LLM-as-a-Judge, or hybrid scoring. We further introduce an evidence-based auditing mechanism that combines deterministic post-hoc rules with a meta-judge to review suspicious positive scores unsupported by execution trajectories or final artifacts. Using this framework, we evaluate 23 commercial API models and 12 locally deployed open-weight models over 7,587 applicable model–task runs. We report accuracy and token consumption, and use public pricing to estimate reference costs for API models and analyze their accuracy–cost trade-offs. Results show that model strengths are workload-dependent and that text and multimodal API workloads exhibit distinct accuracy–cost frontiers. Recent locally deployed open-weight models are competitive in several task groups, although they still trail frontier commercial models overall. The audit further identifies and removes 143 positive scores unsupported by execution evidence. Our main contributions are as follows:

- We introduce DAREBench, comprising 233 tasks adapted from 22 source benchmarks and organized under a unified runtime into a 2×3 workload matrix defined by modality and execution form.
- We evaluate 35 commercial API and locally deployed open-weight models over 7,587 model–task runs, revealing workload-dependent rankings, distinct accuracy–cost trade-offs across text and multimodal workloads, and the strengths and limitations of local models.

- We characterize unsupported positive scores in agent evaluation and introduce an evidence-based audit that verifies suspicious scores against execution trajectories and final artifacts.

2 Related Work

2.1 Agent Benchmarks and Capability Sources

Agent evaluation is shifting from static question answering to execution in operational environments. TerminalBench (Merrill et al. 2026), OSWorld (Xie et al. 2024), and MMSearch (Jiang et al. 2025) respectively assess command-line workflows, GUI interaction, and multimodal search, but remain tied to specific domains or environments. OpenClaw has further motivated general-agent benchmarks. ClawEval (Ye et al. 2026) emphasizes trace-based trustworthy evaluation; ClawBench (Zhang et al. 2026) targets real-time web interaction; ClawArena (Ji et al. 2026) studies conflict reasoning; WildClawBench (Ding et al. 2026) and ZClawBench (Z.AI 2026) introduce in-the-wild tool-use tasks; and ClawsBench (Li et al. 2026b) jointly evaluates capability and safety. However, differing runtimes, task taxonomies, and scoring protocols hinder systematic comparison under a unified framework.

2.2 LLM-as-a-Judge in Agent Evaluation

Following its validation on MT-Bench and Chatbot Arena (Zheng et al. 2023), LLM-as-a-Judge has been applied to code (Jimenez et al. 2024; Jiang et al. 2026), multimodal (Chen et al. 2024), dialogue (Dubois, Liang, and Hashimoto 2024), and security evaluation (Shao et al. 2026). Agent tasks are harder to judge because they involve long-horizon tool use, dynamic interactions, and artifact verification (Li et al. 2025). Prior work shows that judges may infer success from plausible outputs without verifying execution evidence, causing “hallucinated correctness” (You et al. 2026; Gurram 2026). ClawEval (Ye et al. 2026) likewise reports substantial misses without audit logs. Yet systematic characterization and mitigation of such hallucinations remain limited.

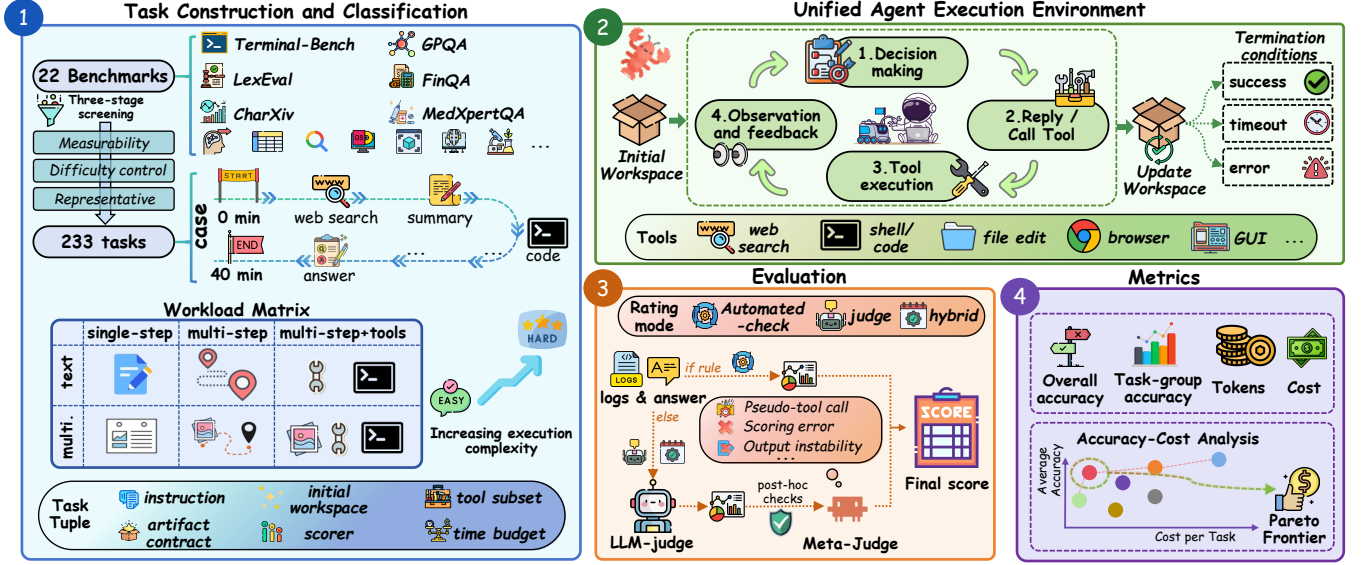


Figure 2: Overview of DAREBench. The benchmark comprises four stages: (1) constructing 233 tasks from 22 source benchmarks and organizing them by modality and execution form; (2) executing all models in a shared agent environment; (3) applying automated, LLM-based, or hybrid scoring with evidence-based auditing; and (4) reporting overall and workload-level accuracy, token usage, reference cost, and accuracy–cost frontiers.

3 DAREBench

As illustrated in Figure 2, DAREBench consists of four components: workload construction, shared agent execution, reliable scoring, and deployment-oriented metrics. It organizes 233 tasks from 22 source benchmarks into a 2×3 workload matrix and evaluates all models under a unified execution protocol. Performance is assessed using automated, LLM-based, or hybrid scoring with evidence-based auditing, and reported alongside token usage and reference cost.

3.1 Problem Definition

Unlike conventional benchmarks that evaluate only final responses, agentic tasks require models to interact with an execution environment and produce verifiable outputs. We formalize each task as

$$\tau_i = (p_i, \mathcal{W}_i^0, \mathcal{T}_i, \mathcal{C}_i, G_i, t_i^{\max}), \quad (1)$$

where p_i is the task instruction, $\mathcal{W}_i^0$ the initial workspace, $\mathcal{T}_i$ the available tool set, $\mathcal{C}_i$ the artifact contract, G_i the task-specific scorer, and $t_i^{\max}$ the execution-time budget. The artifact contract specifies the required output paths and their validity predicates:

$$\mathcal{C}_i = \{(a_{i,j}, \phi_{i,j})\}_{j=1}^{|\mathcal{C}_i|}, \quad \phi_{i,j} : \text{file bytes} \rightarrow \{0, 1\}. \quad (2)$$

Let $\mathcal{W}^*[a]$ denote the content at path a in the final workspace. Complete contract satisfaction is defined as

$$\text{Sat}(\mathcal{W}^*, \mathcal{C}_i) = \prod_{j=1}^{|\mathcal{C}_i|} \mathbf{1}[\exists \mathcal{W}^*[a_{i,j}] \phi_{i,j}(\mathcal{W}^*[a_{i,j}])]. \quad (3)$$

Under this formulation, a model operates within a shared execution environment to produce the required artifacts, which are evaluated by G_i . Task construction and scoring are described in §3.2 and §3.3, respectively.

3.2 Task Construction

Query Selection. To construct a diverse and deployment-relevant evaluation suite, we draw candidate tasks from 22 publicly available benchmarks spanning reasoning, long-context understanding, question answering, instruction following, domain-specific analysis, code and shell workflows, desktop control, and multimodal understanding and search.

We apply a three-stage curation procedure. First, the *feasibility filter* removes tasks for which an explicit artifact contract $\mathcal{C}_i$ or a reliable scorer G_i cannot be defined. Second, the *difficulty filter* excludes overly simple tasks. For single-step tasks, we run Qwen3.5-27B three times and remove tasks solved correctly in all three trials. Where source difficulty annotations are available, we retain only tasks labeled *medium* or *hard*. Finally, the *representativeness filter* manually selects tasks according to the category structure of each source benchmark, preserving broad coverage while avoiding over-representation of any single category. This procedure yields 233 tasks from the 22 source benchmarks.

Workload Taxonomy. Rather than organizing results by source benchmark, we group tasks along two workload dimensions: input modality and execution form. The modality dimension $\mathcal{M}$ distinguishes text-only and multimodal inputs, while the execution-form dimension $\mathcal{K}$ distinguishes single-step, multi-step, and multi-step-plus-tools tasks. Their Cartesian product defines six workload groups:

$$\mathcal{G} = \mathcal{M} \times \mathcal{K}, \quad (4)$$

$$\mathcal{M} = \{\text{text}, \text{multimodal}\}, \quad (5)$$

$$\mathcal{K} = \{\text{single-step}, \text{multi-step}, \text{multi-step+tools}\}. \quad (6)$$

Operationally, we define a deterministic mapping $\varphi : \mathcal{B} \rightarrow \mathcal{G}$ that assigns each source benchmark to a single workload group according to its dominant modality and required execution form. A task i originating from benchmark $b(i) \in \mathcal{B}$ inherits the corresponding group label:

$$g_i = \varphi(b(i)) = (m_i, k_i) \in \mathcal{G}, \quad (7)$$

where m_i denotes the input modality and k_i the required execution form. Let $T_g = \{i : g_i = g\}$ denote the set of tasks assigned to group g . These sets partition the benchmark, satisfying

$$\sum_{g \in \mathcal{G}} |T_g| = N, \quad (8)$$

where $N = 233$. The benchmark-to-group assignments and task counts are shown in the Supplementary Material.

This taxonomy stratifies evaluation results along workload dimensions that are obscured by aggregate leaderboards. For example, a model may perform well on single-step text tasks while struggling with multi-step tool-mediated execution. Reporting performance across $\mathcal{G}$ therefore reveals workload-specific variation that would otherwise be hidden by a single aggregate score.

3.3 Evaluation Protocol

Task Scoring. Because DAREBench contains heterogeneous task types, each task τ_i is assigned one of three scoring modes during benchmark construction:

$$\eta_i \in \{\text{auto}, \text{judge}, \text{hybrid}\},$$

corresponding to automated, LLM-based, and hybrid scoring.

- **Automated Scoring.** Automated scoring is used for tasks with deterministic answers or artifacts that can be verified through rule-based procedures. For single-step tasks, we check both the presence of required artifacts and the consistency of their contents with the ground truth. For multi-step tasks, we define task-specific scoring criteria over independently verifiable intermediate states and final outputs, weighted by their operational importance. This produces an automated score $s_{m,i}^{\text{auto}}$ for model m on task i .

- **LLM-Based Scoring.** For tasks involving open-ended outputs or multiple semantic constraints that cannot be reliably evaluated using deterministic rules, we employ an LLM-as-a-Judge. A fixed judge model receives the task prompt p_i , a summary of the execution trajectory, the final artifacts in $\mathcal{W}_i^*$, and a manually specified task-level rubric. It returns criterion-level scores in a constrained format, which are aggregated using predefined rubric weights to obtain $s_{m,i}^{\text{judge}}$.

- **Hybrid Scoring.** Tasks containing both rule-verifiable and semantically judged criteria combine the two scoring modes. The weights $w_i^{\text{auto}}, w_i^{\text{judge}} \in [0, 1]$ are fixed during task construction and satisfy $w_i^{\text{auto}} + w_i^{\text{judge}} = 1$. The resulting task score is

$$s_{m,i} = \begin{cases} s_{m,i}^{\text{auto}}, & \eta_i = \text{auto}, \\ s_{m,i}^{\text{judge}}, & \eta_i = \text{judge}, \\ w_i^{\text{auto}} s_{m,i}^{\text{auto}} + w_i^{\text{judge}} s_{m,i}^{\text{judge}}, & \eta_i = \text{hybrid}. \end{cases} \quad (9)$$

All task scores are normalized to $[0, 1]$.

Evidence-Based Audit. Our pilot experiments revealed cases in which the primary judge awarded positive credit that was not supported by the execution trajectory or final artifacts. We refer to these cases as *unsupported positive judgments*. Typical examples include nonzero scores assigned to empty outputs, executions without substantive operations, or runs missing required artifacts.

To identify and correct such cases, we apply deterministic post-hoc rules to all scores involving the primary judge. Cases satisfying one or more trigger conditions are routed to a separate meta-judge from a different model family. The meta-judge receives the complete task prompt, execution trajectory, final artifacts, scoring rubric, and the primary judge’s score and rationale. It then determines whether the awarded positive score is supported by the available execution evidence.

If the meta-judge determines that the score is unsupported as a whole, the task score is set to zero; otherwise, the original score is retained. The audited score is defined as

$$s_{m,i}^{\text{audit}} = (1 - h_{m,i}) s_{m,i}, \quad (10)$$

where $h_{m,i} \in \{0, 1\}$ indicates whether the primary score is confirmed to be unsupported. Unflagged cases and flagged cases upheld by the meta-judge are assigned $h_{m,i} = 0$. Detailed trigger rules, meta-judge prompts, case studies, and human–meta-judge agreement results are provided in the appendix.

Metrics and Efficiency Accounting. Let $\mathcal{I}_m$ denote the set of tasks applicable to model m , and let $\mathcal{I}_{m,g} = \mathcal{I}_m \cap T_g$ denote its applicable tasks in workload group g . We report overall and group-level audited accuracy as

$$\text{Acc}(m) = \frac{1}{|\mathcal{I}_m|} \sum_{i \in \mathcal{I}_m} s_{m,i}^{\text{audit}}, \quad (11)$$

$$\text{Acc}_g(m) = \frac{1}{|\mathcal{I}_{m,g}|} \sum_{i \in \mathcal{I}_{m,g}} s_{m,i}^{\text{audit}}. \quad (12)$$

We additionally report average token usage and estimated dollar cost per task, averaged over $\mathcal{I}_m$. Writing $u_{m,i}^k$ for the token count of kind $k \in \{\text{in}, \text{out}, \text{cr}, \text{cw}\}$ (uncached input, output, cache-read, and cache-write), the two efficiency metrics are

$$\text{Token}(m) = \frac{1}{|\mathcal{I}_m|} \sum_{i \in \mathcal{I}_m} \sum_k u_{m,i}^k, \quad (13)$$

$$\text{Cost}(m) = \frac{1}{|\mathcal{I}_m|} \sum_{i \in \mathcal{I}_m} \frac{1}{10^6} \sum_k \alpha_m^k u_{m,i}^k, \quad (14)$$

where α_m^k is the per-million-token price of kind k , taken from model-specific OpenRouter reference rates. When a provider exposes no separate cache-read or cache-write rate, those tokens are billed at the input price ($\alpha_m^{\text{cr}} = \alpha_m^{\text{cw}} = \alpha_m^{\text{in}}$). For locally deployed models, any hosted-inference price is only a third-party reference rather than the measured cost of local vLLM serving; accuracy–cost frontier analyses are therefore based on API reference costs. Full per-model rates are listed in the supplementary material.

Model	Text				Multi-Modal			
	Single	Multi	Multi+Tools	Avg	Single	Multi	Multi+Tools	Avg
API Deployment								
Claude-Opus-4.8 (Anthropic 2026b)	98.8	78.1	65.3	76.3	85.0	69.2	61.8	71.1
GPT-5.5 (OpenAI 2026c)	94.1	83.6	59.8	76.8	90.0	75.0	42.4	67.7
Claude-Opus-4.6 (Anthropic 2026a)	91.5	83.4	58.3	75.9	75.0	76.2	48.1	65.9
Gemini-3.1-Pro (Google DeepMind 2026a)	96.4	80.7	56.1	74.3	85.0	88.5	32.7	67.9
GPT-5.4 (OpenAI 2026a)	92.1	82.4	59.6	75.8	82.5	71.2	40.3	63.5
MiMo-V2.5 (Xiaomi MiMo Team 2026c)	94.7	82.5	57.5	75.5	80.0	66.7	43.0	62.1
Claude-Sonnet-4.6 (Anthropic 2026c)	89.2	76.8	59.9	72.6	75.0	59.6	42.3	57.9
Qwen3.6-Plus (Qwen Team 2026d)	95.5	77.7	54.8	72.1	75.0	61.5	41.2	58.2
Qwen3.5-Plus (Qwen Team 2026a)	89.5	77.3	53.9	70.9	70.0	69.2	39.1	58.8
Doubao-Seed2.0-Pro (Bytedance Seed 2026)	95.9	77.4	51.1	70.8	75.0	61.5	42.6	58.7
Gemini-3.5-Flash (Google DeepMind 2026b)	94.9	80.9	51.7	72.7	85.0	50.0	26.7	51.7
GPT-5.4-mini (OpenAI 2026b)	77.8	79.1	50.3	69.2	70.0	46.2	35.9	49.3
Claude-Haiku-4.5 (Anthropic 2025)	89.1	76.7	50.1	69.2	50.5	46.2	31.6	42.3
MiMo-V2-Omni (Xiaomi MiMo Team 2026a)	70.4	75.4	51.5	66.7	75.0	34.6	34.7	46.0
Kimi-K2.5 (Kimi Team 2026)	89.2	70.0	45.6	64.1	65.0	54.4	38.7	51.9
Doubao-Seed2.0-Mini (Bytedance Seed 2026)	75.7	78.9	42.5	66.1	55.0	51.9	34.3	46.6
MiMo-V2.5-Pro (Xiaomi MiMo Team 2026d)	91.6	81.7	61.2	76.0	—	—	—	—
DeepSeek-V4-Pro (DeepSeek-AI 2026)	91.6	80.9	54.3	73.2	—	—	—	—
MiMo-V2-Pro (Xiaomi MiMo Team 2026b)	91.7	75.4	58.4	71.6	—	—	—	—
MiniMax-M2.7 (MiniMax 2026)	80.9	77.8	53.0	69.8	—	—	—	—
GLM-5 (GLM-5-Team et al. 2026)	84.9	75.0	53.3	68.9	—	—	—	—
DeepSeek-V3.2 (DeepSeek-AI 2025)	80.0	71.1	46.8	63.9	—	—	—	—
GLM-4.7 (Z.AI 2025)	73.1	76.2	33.6	61.3	—	—	—	—
Local Deployment								
Qwen3.6-27B (Qwen Team 2026b)	84.9	72.1	37.2	61.8	75.0	55.8	38.5	55.1
Qwen3.6-35B-A3B (Qwen Team 2026c)	79.3	70.2	33.9	59.0	65.0	57.7	32.7	51.0
Qwen3.5-27B (Qwen Team 2026a)	85.8	59.9	27.0	51.9	55.0	53.8	24.5	43.9
Qwen3.5-35B-A3B (Qwen Team 2026a)	66.2	57.9	34.0	50.8	60.0	38.5	22.9	39.0
Qwen3-VL-8B-Instruct (Bai et al. 2025a)	35.7	58.2	28.9	45.5	40.0	25.0	22.3	28.3
Qwen2.5-VL-32B-Instruct (Bai et al. 2025b)	53.3	61.8	20.1	46.6	15.0	34.6	13.0	21.5
Qwen3-VL-32B-Instruct (Bai et al. 2025a)	48.3	48.7	25.2	40.6	55.0	34.6	17.4	34.3
Gemma-4-26B-A4B-it (Gemma Team 2026)	83.0	40.6	19.3	38.6	14.0	55.8	20.7	31.6
Qwen3.5-9B (Qwen Team 2026a)	56.4	32.9	27.1	33.8	62.5	28.8	16.2	33.9
Qwen2.5-VL-7B-Instruct (Bai et al. 2025b)	28.0	35.2	19.7	29.0	10.0	7.7	4.0	7.0
Gemma-3-27B-it (Gemma Team et al. 2025)	42.4	29.5	14.5	26.0	25.0	13.5	0.0	12.0
Qwen3-Coder-30B-A3B-Instruct (Yang et al. 2025)	73.9	56.3	37.7	52.1	—	—	—	—

Table 1: Per-model accuracy (%) across the six DAREBench task groups. Averages are task-count-weighted (Text: 20/87/55; Multi-Modal: 20/26/25). **Bold**, underlined, and *italicized* values denote first, second, and third place within each deployment mode (competition ranking); “—” marks text-only models not evaluated on multimodal tasks.

4 Experiments

4.1 Experimental Setup

Models Evaluated. We evaluate 35 models: 23 commercial API models from 10 families accessed through OpenRouter and 12 open-weight models served locally with vLLM on a single NVIDIA RTX PRO 6000 GPU (96 GB VRAM). To represent practical on-premise deployment, local models use a common FP16, 32K-context configuration—the largest stable setting shared by all models on one GPU—without quantization or multi-GPU serving. API-local comparisons therefore reflect end-to-end performance under their respective deployment profiles rather than context-matched intrinsic capability. All runs use the same OpenClaw runtime and default inference settings, with isolated sessions, workspaces, artifacts, and logs for each task.

Infrastructure. All models use the default OpenClaw tool suite, supplemented with `web_search`, `web_fetch`, and nine Clawhub skills covering reasoning, automation, and

general utilities. The complete skill list is provided in the supplementary material. Tool and skill visibility, permissions, and interfaces are kept identical across models.

Evaluation Metrics. We use accuracy as the primary capability metric, reporting both overall accuracy and accuracy for six workload groups: Text-SS, Text-MS, Text-MT, MM-SS, MM-MS, and MM-MT. Efficiency is measured by average token usage and estimated USD cost per task, computed from measured input/output tokens using OpenRouter reference rates; pricing details are provided in the supplementary material. For LLM-based scoring, we use Qwen3.5-VL-Plus as the primary judge and Claude-Opus-4.6 as the meta-judge.

4.2 Main Results

Table 1 reports Accuracy across the six DAREBench task groups. Models are grouped by deployment mode (API and Local Deployment). For text-only models, we report their performance on text-only tasks exclusively. We also report token consumption and USD cost for all API models in Ta-

Model	Tokens/task								Cost\$/task (OpenRouter rates)							
	Text				MM				Text				MM			
	SS	MS	MT	Avg	SS	MS	MT	Avg	SS	MS	MT	Avg	SS	MS	MT	Avg
Claude-Opus-4.8	52k	85k	301k	154k	77k	98k	147k	109k	0.075	0.184	0.417	0.250	0.060	0.132	0.164	0.123
GPT-5.5	41k	104k	190k	125k	41k	69k	317k	148k	0.051	0.139	0.217	0.155	0.045	0.103	0.365	0.179
Claude-Opus-4.6	35k	112k	605k	270k	49k	110k	251k	143k	0.175	0.245	0.784	0.419	0.062	0.362	0.406	0.293
Gemini-3.1-Pro	33k	123k	505k	241k	43k	115k	128k	99k	0.083	0.107	0.283	0.164	0.055	0.138	0.136	0.114
GPT-5.4	50k	81k	396k	184k	50k	69k	291k	142k	0.056	0.058	0.244	0.121	0.036	0.074	0.170	0.097
MiMo-V2.5	75k	143k	603k	291k	79k	95k	523k	241k	0.003	0.003	0.011	0.006	0.001	0.004	0.007	0.005
Claude-Sonnet-4.6	49k	157k	575k	286k	52k	125k	330k	177k	0.154	0.205	0.902	0.436	0.063	0.290	0.337	0.243
Qwen3.6-Plus	57k	163k	607k	301k	50k	181k	561k	278k	0.039	0.086	0.321	0.160	0.027	0.108	0.292	0.150
Qwen3.5-Plus	40k	80k	529k	228k	40k	111k	248k	140k	0.021	0.035	0.218	0.095	0.018	0.050	0.105	0.060
Doubao-Seed2.0-Pro	46k	75k	452k	199k	164k	173k	289k	211k	0.019	0.023	0.105	0.051	0.031	0.056	0.079	0.057
Gemini-3.5-Flash	101k	338k	773k	456k	191k	590k	625k	490k	0.111	0.193	0.401	0.253	0.155	0.305	0.346	0.277
GPT-5.4-mini	31k	52k	131k	76k	38k	40k	95k	59k	0.008	0.010	0.033	0.017	0.007	0.012	0.022	0.014
Claude-Haiku-4.5	46k	75k	375k	173k	54k	108k	164k	113k	0.022	0.035	0.143	0.070	0.018	0.042	0.052	0.039
MiMo-V2-Omni	49k	90k	618k	264k	51k	54k	284k	134k	0.003	0.003	0.017	0.008	<0.001	0.003	0.005	0.003
Kimi-K2.5	73k	184k	550k	295k	41k	226k	268k	189k	0.042	0.072	0.221	0.119	0.025	0.088	0.122	0.082
Doubao-Seed2.0-Mini	55k	79k	460k	205k	318k	148k	384k	279k	0.008	0.009	0.049	0.023	0.034	0.018	0.041	0.031
MiMo-V2.5-Pro	72k	134k	441k	230k	—	—	—	—	0.010	0.008	0.020	0.012	—	—	—	—
DeepSeek-V4-Pro	61k	126k	390k	207k	—	—	—	—	0.013	0.015	0.040	0.023	—	—	—	—
MiMo-V2-Pro	42k	98k	520k	234k	—	—	—	—	0.005	0.009	0.026	0.014	—	—	—	—
MiniMax-M2.7	50k	75k	377k	175k	—	—	—	—	0.013	0.010	0.056	0.026	—	—	—	—
GLM-5	48k	125k	449k	225k	—	—	—	—	0.037	0.057	0.185	0.098	—	—	—	—
DeepSeek-V3.2	122k	176k	454k	263k	—	—	—	—	0.016	0.019	0.046	0.028	—	—	—	—
GLM-4.7	35k	92k	229k	131k	—	—	—	—	0.016	0.026	0.052	0.034	—	—	—	—

Table 2: Per-model token consumption and dollar cost per task on DAREBench. Averages are task-count-weighted. SS, MS, and MT denote Single-Step, Multi-Step, and Multi-Step+Tools, respectively.

ble 2 (corresponding details for local models are provided in the Supplementary), and visualize the API accuracy–cost trade-off in Figure 1. We summarize the main findings below:

- **Top performers diverge across workload types, and model selection should be workload-aware.** Overall, although stronger models tend to exhibit higher aggregate performance, relative differences persist across workloads: On Text-SS (98.8), Text-MT (65.3), and MM-MT (61.8), Claude-Opus-4.8 is the strongest model; GPT-5.5 has the best performance on Text-MS (83.6) and MM-SS (90.0); while on MM-MS (88.5), the SOTA model is Gemini-3.1-Pro. This indicates that top performers diverge across workload types, and model selection should be guided by target workload characteristics to achieve the best performance.

- **Recent open-weight models offer competitive on-premise performance, but a notable gap to frontier models remains.** For workloads requiring local deployment, Qwen3.6-27B emerges as the strongest default (Overall 59.8), with aggregate performance approaching that of some earlier commercial models. However, it still has a substantial gap of approximately 15 percentage points relative to frontier commercial models. Qwen3.6-35B-A3B follows closely (Overall 56.6), serving as a viable MoE alternative when lower activated-parameter cost or latency is desired. Although smaller 8B/9B local models are easy to deploy, they exhibit marked instability within the full agent loop.

- **Token consumption and USD cost form two decoupled extremal chains.** Table 2 shows token consumption and USD cost for all API models. On the token-consumption axis, models differ substantially in the tokens required to complete tasks: GPT-5.4-mini is the lowest across the board;

Pattern	Local	API	Total
Deadlock loop	86	0	86
Rubric leakage	12	14	26
Pseudo-tool-call credulity	20	0	20
Output instability	5	2	7
Timeout-with-credit	1	3	4
Total	124	19	143

Table 3: Audit type of LLM Judge Hallucination Patterns

conversely, Gemini-3.5-Flash highest on both axes (Text Avg 456k, MM Avg 490k), consuming roughly six and eight times as many tokens as GPT-5.4-mini, respectively. Yet when converted to USD cost, the ordering shifts dramatically: MiMo-V2.5 is the cheapest model on text tasks (\$0.006), while MiMo-V2-Omni is the cheapest on multimodal tasks (\$0.003); at the other extreme, Claude-Sonnet-4.6 tops the text column at \$0.436 and Claude-Opus-4.6 tops the multimodal column at \$0.293. Notably, the extremes of the token-consumption and cost rankings do not coincide: MiMo-V2.5 incurs high token usage (Overall 276k) yet yields a low Overall cost of \$0.005, whereas Claude-Opus-4.6 consumes fewer tokens overall (231k) at roughly $71\times$ that cost (\$0.381).

- **Cost-efficiency-aware model selection.** Figure 1 presents Pareto fronts separately for text and multimodal panels. In the text panel, MiMo-V2.5 (\$0.006, 75.5), MiMo-V2.5-Pro (\$0.012, 76.0), and GPT-5.5 (\$0.155, 76.8) form an optimal sequence ordered by increasing cost. The step from MiMo-V2.5 to MiMo-V2.5-Pro raises cost by approximately $2\times$ while accuracy improves by 0.5 percentage points, and the subsequent step to GPT-5.5 raises cost by a further $13\times$

for only 0.8 percentage points, indicating that the sub- $\$0.02$ range is already near the accuracy–cost saturation point for text tasks. In the multimodal panel, the frontier slope is markedly steeper: after MiMo-V2-Omni ($\$0.003$, 46.0) and MiMo-V2.5 ($\0.005, 62.1), GPT-5.4 ($\$0.097$, 63.5), Gemini-3.1-Pro ($\$0.114$, 67.9) and Claude-Opus-4.8 ($\$0.123$, 71.1) successively raise the bar, and compared with the text panel, the cost-to-performance gains are notably more pronounced.

4.3 LLM Judge Reliability

Table 3 reports the hallucination patterns identified by the meta-judge and the corresponding audit results. These include rubric leakage (the judge mistakes answer information embedded in the grading rubrics for the agent’s execution output), deadlock loop (the agent repeatedly cycles through materially equivalent reasoning or actions without making workspace progress, eventually exhausting the 32K context window and leaving the task incomplete, while the judge still awards positive credit), pseudo-tool-call (the model outputs tool-like text rather than an actually executable tool call), timeout-with-credit (timed-out trajectories still receiving positive scores), and output instability (incomplete or malformed output). The meta-judge audit removes 143 unsupported positive scores and reveals the dominant hallucination patterns. We also observe a pronounced divergence in hallucination patterns by deployment mode. Locally deployed models predominantly exhibit deadlock loops and pseudo-tool-call credulity (86 and 20 cases, respectively), while these two patterns never appear in API-based models. The same 32K local deployment profile is applied to all local models, yet only a subset repeatedly enters non-progressing loops before reaching the limit. We therefore treat these cases as model-dependent agent failures exposed by the deployment environment, rather than as context exhaustion caused by task length alone. Conversely, timeout-with-credit is more common in API invocation scenarios. We attribute this disparity to differences in error modes: stronger trajectory control helps a model avoid non-progressing loops, whereas API execution remains more exposed to network and serving latency. These results characterize reliability under the specified single-GPU deployment profile and do not assume that the same failure rate would hold under a larger-context serving configuration.

4.4 Case Study

Figure 3 illustrates a representative multimodal agent failure. The task requires filtering foods by two visual constraints before retrieving their glycemic-index values. Qwen3.5-9B misreads the chart and retains *Avocado* despite the CO_2 threshold; this early error changes the candidate set and leads to the wrong answer. It also relies on unverified sources. In contrast, Claude Opus 4.8 applies both constraints and checks retrieved values against authoritative sources. With identical tools and runtime, the contrast highlights differences in visual grounding, constraint tracking, and evidence verification, as well as how early errors propagate through multi-step trajectories. Additional examples are provided in the Supplementary Material.

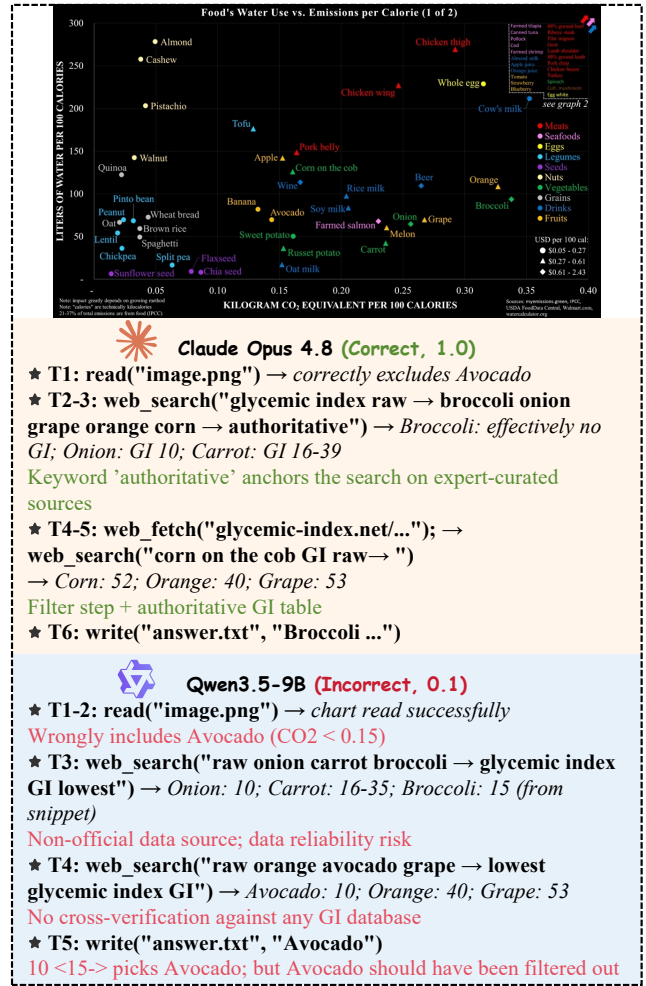


Figure 3: Multi-Step Visual Search on a Nutrition Scatter Chart. **Example Task:** Among all Fruits and Vegetables with $\text{CO}_2 > 0.15$ kg per 100 calories and price tier of either Highest or Medium, which single food has the lowest Glycemic Index (GI)?

5 Conclusion

We introduce DAREBench, a workload- and deployment-aware benchmark for evaluating models as agents. Built on a shared OpenClaw environment, it organizes 233 tasks into six workload groups and evaluates 35 API and locally deployed open-weight models using accuracy, token usage, and reference cost. Results show that model rankings vary across workloads, text and multimodal tasks exhibit distinct accuracy–cost trade-offs, and local models remain competitive in several groups but trail frontier models overall. We further identify 143 positive scores unsupported by execution evidence, highlighting the need to audit LLM-based judgments against trajectories and final artifacts. It also exposes deployment-specific failures that aggregate leaderboards often obscure. Overall, DAREBench supports workload-specific model comparison and more reliable deployment decisions.

References

- Anthropic. 2025. Claude Haiku 4.5 System Card. <https://www-cdn.anthropic.com/7aad69bf12627d42234e01ee7c36305dc2f6a970.pdf>.
- Anthropic. 2026a. Claude Opus 4.6 System Card. <https://www.anthropic.com/claude-opus-4-6-system-card>.
- Anthropic. 2026b. Claude Opus 4.8 System Card. <https://www.anthropic.com/claude-opus-4-8-system-card>.
- Anthropic. 2026c. Claude Sonnet 4.6 System Card. <https://www.anthropic.com/claude-sonnet-4-6-system-card>.
- Bai, S.; Cai, Y.; Chen, R.; Chen, K.; Chen, X.; Cheng, Z.; Deng, L.; Ding, W.; Gao, C.; Ge, C.; et al. 2025a. Qwen3-VL Technical Report. *arXiv preprint arXiv:2511.21631*.
- Bai, S.; Chen, K.; Liu, X.; Wang, J.; Ge, W.; Song, S.; Dang, K.; Wang, P.; Wang, S.; Tang, Y.; et al. 2025b. Qwen2.5-VL Technical Report. *arXiv preprint arXiv:2502.13923*.
- Bai, Y.; Tu, S.; Zhang, J.; Peng, H.; Wang, X.; Lv, X.; Cao, S.; Xu, J.; Hou, L.; Dong, Y.; et al. 2025c. Longbench v2: Towards deeper understanding and reasoning on realistic long-context multitasks. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*.
- Bytedance Seed. 2026. Seed2.0 Model Card: Towards Intelligence Frontier for Real-World Complexity. *arXiv preprint arXiv:2607.00248*.
- Center for AI Safety; Scale AI; and HLE Contributors Consortium. 2026. A benchmark of expert-level academic questions to assess AI capabilities. *Nature*.
- Chen, D.; Chen, R.; Zhang, S.; Wang, Y.; Liu, Y.; Zhou, H.; Zhang, Q.; Wan, Y.; Zhou, P.; and Sun, L. 2024. MLLM-as-a-Judge: Assessing Multimodal LLM-as-a-Judge with Vision-Language Benchmark. In *Proceedings of the 41st International Conference on Machine Learning*.
- Chen, Z.; Chen, W.; Smiley, C.; Shah, S.; Borova, I.; Langdon, D.; Moussa, R.; Beane, M.; Huang, T.-H.; Routledge, B.; and Wang, W. Y. 2021. FinQA: A Dataset of Numerical Reasoning over Financial Data. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.
- DeepSeek-AI. 2025. DeepSeek-V3.2: Pushing the Frontier of Open Large Language Models. *arXiv preprint arXiv:2512.02556*.
- DeepSeek-AI. 2026. DeepSeek-V4: Towards Highly Efficient Million-Token Context Intelligence. *arXiv preprint arXiv:2606.19348*.
- Ding, S.; Dai, X.; Xing, L.; Ding, S.; Liu, Z.; Yang, J.; Yang, P.; Zhang, Z.; Wei, X.; Fang, X.; Ma, Y.; Duan, H.; Shao, J.; Wang, J.; Lin, D.; Chen, K.; and Zang, Y. 2026. WildClawBench. Software and dataset: <https://github.com/InternLM/WildClawBench>.
- Dubois, Y.; Liang, P.; and Hashimoto, T. B. 2024. Length-Controlled AlpacaEval: A Simple Way to Debias Automatic Evaluators. In *First Conference on Language Modeling*.
- Gemma Team. 2026. Gemma 4 Technical Report. *arXiv preprint arXiv:2607.02770*.
- Gemma Team; Kamath, A.; Ferret, J.; Pathak, S.; Vieillard, N.; Merhej, R.; et al. 2025. Gemma 3 Technical Report. *arXiv preprint arXiv:2503.19786*.
- GLM-5-Team; et al. 2026. GLM-5: from Vibe Coding to Agentic Engineering. *arXiv preprint arXiv:2602.15763*.
- Google DeepMind. 2026a. Gemini 3.1 Pro Model Card. <https://deepmind.google/models/model-cards/gemini-3-1-pro/>.
- Google DeepMind. 2026b. Gemini 3.5 Flash Model Card. <https://deepmind.google/models/model-cards/gemini-3-5-flash/>.
- Gupta, N.; Chatterjee, R.; Haas, L.; Tao, C.; Wang, A.; Liu, C.; Oiwa, H.; Gribovskaya, E.; Ackermann, J.; Blitzer, J.; et al. 2026. DeepSearchQA: Bridging the Comprehensiveness Gap for Deep Research Agents. *arXiv preprint arXiv:2601.20975*.
- Gurram, B. 2026. Evaluating Tool-Using Language Agents: Judge Reliability, Propagation Cascades, and Runtime Mitigation in AgentProp-Bench. *arXiv preprint arXiv:2604.16706*.
- He, Y.; Li, W.; Zhang, H.; Li, S.; Mandyam, K.; Khosla, S.; Xiong, Y.; Wang, N.; Peng, X.; Li, B.; et al. 2025. Advancedif: Rubric-based benchmarking and reinforcement learning for advancing llm instruction following. *arXiv preprint arXiv:2511.10507*.
- Hendrycks, D.; Burns, C.; Basart, S.; Zou, A.; Mazeika, M.; Song, D.; and Steinhardt, J. 2021. Measuring Massive Multitask Language Understanding. In *International Conference on Learning Representations*.
- Ji, H.; Xiong, K.; Han, S.; Xia, P.; Qiu, S.; Zhou, Y.; Liu, J.; Li, J.; Li, B.; Zheng, Z.; Xie, C.; and Yao, H. 2026. ClawArena: Benchmarking AI Agents in Evolving Information Environments. *arXiv preprint arXiv:2604.04202*.
- Jiang, D.; Zhang, R.; Guo, Z.; Wu, Y.; Lei, J.; Qiu, P.; Lu, P.; Chen, Z.; Song, G.; Gao, P.; Liu, Y.; Li, C.; and Li, H. 2025. MMSearch: Unveiling the Potential of Large Models as Multi-modal Search Engines. In *The Thirteenth International Conference on Learning Representations*.
- Jiang, H.; Chen, Y.; Cao, Y.; Lee, H.-y.; and Tan, R. T. 2026. CodeJudgeBench: Benchmarking LLM-as-a-Judge for Coding Tasks. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.
- Jimenez, C. E.; Yang, J.; Wettig, A.; Yao, S.; Pei, K.; Press, O.; and Narasimhan, K. 2024. SWE-bench: Can Language Models Resolve Real-World GitHub Issues? In *The Twelfth International Conference on Learning Representations*.
- Kimi Team. 2026. Kimi K2.5: Visual Agentic Intelligence. *arXiv preprint arXiv:2602.02276*.
- Kwon, W.; Li, Z.; Zhuang, S.; Sheng, Y.; Zheng, L.; Yu, C. H.; Gonzalez, J. E.; Zhang, H.; and Stoica, I. 2023. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

- Li, C.; Tang, Z.; Huang, M.; Lin, Y.; Huang, S.; Liu, S.; Ye, B.; Li, R.; Li, L.; Wang, B.; and Yuan, Y. 2026a. Claw-Eval-Live: A Live Agent Benchmark for Evolving Real-World Workflows. *arXiv preprint arXiv:2604.28139*.
- Li, D.; Jiang, B.; Huang, L.; Beigi, A.; Zhao, C.; Tan, Z.; Bhattacharjee, A.; Jiang, Y.; Chen, C.; Wu, T.; et al. 2025. From Generation to Judgment: Opportunities and Challenges of LLM-as-a-Judge. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*.
- Li, H.; Chen, Y.; Ai, Q.; Wu, Y.; Zhang, R.; and Liu, Y. 2024. Lexeval: A comprehensive chinese legal benchmark for evaluating large language models. *Advances in Neural Information Processing Systems*.
- Li, X.; Choe, K. W.; Liu, Y.; Chen, X.; Tao, C.; You, B.; Chen, W.; Di, Z.; Sun, J.; Zheng, S.; et al. 2026b. ClawsBench: Evaluating Capability and Safety of LLM Productivity Agents in Simulated Workspaces. *arXiv preprint arXiv:2604.05172*.
- Liu, J.; Cui, L.; Liu, H.; Huang, D.; Wang, Y.; and Zhang, Y. 2020. LogiQA: A Challenge Dataset for Machine Reading Comprehension with Logical Reasoning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*.
- Liu, X.; Yu, H.; Zhang, H.; Xu, Y.; Lei, X.; Lai, H.; Gu, Y.; Ding, H.; Men, K.; Yang, K.; et al. 2024. AgentBench: Evaluating LLMs as Agents. In *The Twelfth International Conference on Learning Representations*.
- Merrill, M. A.; Shaw, A. G.; Carlini, N.; Li, B.; Raj, H.; Bercovich, I.; Shi, L.; Shin, J. Y.; Walshe, T.; Buchanan, E. K.; et al. 2026. Terminal-bench: Benchmarking agents on hard, realistic tasks in command line interfaces. *arXiv preprint arXiv:2601.11868*.
- Mialon, G.; Fourrier, C.; Swift, C.; Wolf, T.; LeCun, Y.; and Scialom, T. 2024. GAIA: A Benchmark for General AI Assistants. In *The Twelfth International Conference on Learning Representations*.
- MiniMax. 2026. The MiniMax-M2 Series: Mini Activations Unleashing Max Real-World Intelligence. *arXiv preprint arXiv:2605.26494*.
- OpenAI. 2026a. GPT-5.4 Thinking System Card. <https://deploymentsafety.openai.com/gpt-5-4-thinking>.
- OpenAI. 2026b. GPT-5.4 Thinking System Card: Appendix—GPT-5.4 mini. <https://deploymentsafety.openai.com/gpt-5-4-thinking/appendix-gpt-5.4-mini>.
- OpenAI. 2026c. GPT-5.5 System Card. <https://openai.com/index/gpt-5-5-system-card/>.
- Pham, T.; Nguyen, N.; Zunjare, P.; Chen, W.; Tseng, Y.-M.; and Vu, T. 2026. SealQA: Raising the Bar for Reasoning in Search-Augmented Language Models. In *The Fourteenth International Conference on Learning Representations*.
- Press, O.; Zhang, M.; Min, S.; Schmidt, L.; Smith, N. A.; and Lewis, M. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*.
- Qwen Team. 2026a. Qwen3.5: Towards Native Multimodal Agents. <https://qwen.ai/blog?id=qwen3.5>.
- Qwen Team. 2026b. Qwen3.6-27B: Flagship-Level Coding in a 27B Dense Model. <https://qwen.ai/blog?id=qwen3.6-27b>.
- Qwen Team. 2026c. Qwen3.6-35B-A3B: Agentic Coding Power, Now Open to All. <https://qwen.ai/blog?id=qwen3.6-35b-a3b>.
- Qwen Team. 2026d. Qwen3.6-Plus: Towards Real World Agents. <https://qwen.ai/blog?id=qwen3.6>.
- Rein, D.; Hou, B. L.; Stickland, A. C.; Petty, J.; Pang, R. Y.; Dirani, J.; Michael, J.; and Bowman, S. R. 2024. GPQA: A Graduate-Level Google-Proof Q&A Benchmark. In *First Conference on Language Modeling*.
- Schick, T.; Dwivedi-Yu, J.; Dessi, R.; Raileanu, R.; Lomeli, M.; Hambro, E.; Zettlemoyer, L.; Cancedda, N.; and Scialom, T. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. In *Advances in Neural Information Processing Systems*.
- Shao, M.; Rani, N.; Milner, K.; Xi, H.; Udeshi, M.; Aggarwal, S.; Putrevu, V. S. C.; Shukla, S. K.; Krishnamurthy, P.; Khorrami, F.; Karri, R.; and Shafique, M. 2026. Towards Effective Offensive Security LLM Agents: Hyperparameter Tuning, LLM as a Judge, and a Lightweight CTF Benchmark. *Proceedings of the AAAI Conference on Artificial Intelligence*.
- SpaceXAI. 2026. Introducing Grok 4.5. <https://x.ai/news/grok-4-5>.
- Steinberger, P.; and OpenClaw Contributors. 2026. OpenClaw—Personal AI Assistant. <https://github.com/openclaw/openclaw>.
- Su, Z.; Gao, J.; Guo, H.; Liu, Z.; Zhang, L.; Geng, X.; Huang, S.; Xia, P.; Jiang, G.; Wang, C.; et al. 2026. Agentvista: Evaluating multimodal agents in ultra-challenging realistic visual scenarios. *arXiv preprint arXiv:2602.23166*.
- Tao, X.; Teng, Y.; Su, X.; Fu, X.; Wu, J.; Tao, C.; Liu, Z.; Bai, H.; Liu, R.; and Kong, L. 2026. MMSearch-Plus: Benchmarking Provenance-Aware Search for Multimodal Browsing Agents. In *The Fourteenth International Conference on Learning Representations*.
- Vijayvargiya, S.; Soni, A. B.; Zhou, X.; Wang, Z. Z.; Dziri, N.; Neubig, G.; and Sap, M. 2026. OpenAgentSafety: A Comprehensive Framework for Evaluating Real-World AI Agent Safety. In *The Fourteenth International Conference on Learning Representations*.
- Wang, Y.; Zhao, Y.; Chen, X.; Guo, S.; Liu, L.; Li, H.; Xiao, Y.; Zhang, J.; Li, Q.; and Xu, K. 2025. VisualSimpleQA: A Benchmark for Decoupled Evaluation of Large Vision-Language Models in Fact-Seeking Question Answering. *arXiv preprint arXiv:2503.06492*.
- Wang, Z.; Xia, M.; He, L.; Chen, H.; Liu, Y.; Zhu, R.; Liang, K.; Wu, X.; Liu, H.; Malladi, S.; et al. 2024. CharXiv: Charting Gaps in Realistic Chart Understanding in Multimodal LLMs. *arXiv preprint arXiv:2406.18521*.
- Wei, J.; Nguyen, K.; Chung, H. W.; Jiao, Y. J.; Papay, S.; Glaese, A.; Schulman, J.; and Fedus, W. 2024. Measuring Short-Form Factuality in Large Language Models. *arXiv preprint arXiv:2411.04368*.

- Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E. H.; Le, Q. V.; and Zhou, D. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems*.
- Wong, R.; Wang, J.; Zhao, J.; Chen, L.; Gao, Y.; Zhang, L.; Zhou, X.; Wang, Z.; Xiang, K.; Zhang, G.; et al. 2025. Widesearch: Benchmarking agentic broad info-seeking. *arXiv preprint arXiv:2508.07999*.
- Wu, X.; Yang, J.; Chai, L.; Zhang, G.; Liu, J.; Du, X.; Liang, D.; Shu, D.; Cheng, X.; Sun, T.; Li, T.; Li, Z.; and Niu, G. 2025. TableBench: A Comprehensive and Complex Benchmark for Table Question Answering. *Proceedings of the AAAI Conference on Artificial Intelligence*.
- Xiaomi MiMo Team. 2026a. Xiaomi MiMo-V2-Omni: Omni-Modal Agentic Foundation Model that Sees, Understands and Acts. <https://mimo.mi.com/docs/en-US/news/previous-news/v2-omni-release>.
- Xiaomi MiMo Team. 2026b. Xiaomi MiMo-V2-Pro. <https://mimo.xiaomi.com/mimo-v2-pro>.
- Xiaomi MiMo Team. 2026c. Xiaomi MiMo-V2.5. <https://mimo.xiaomi.com/mimo-v2-5/>.
- Xiaomi MiMo Team. 2026d. Xiaomi MiMo-V2.5-Pro. <https://mimo.xiaomi.com/mimo-v2-5-pro/>.
- Xie, T.; Zhang, D.; Chen, J.; Li, X.; Zhao, S.; Cao, R.; Hua, T. J.; Cheng, Z.; Shin, D.; Lei, F.; Liu, Y.; Xu, Y.; Zhou, S.; Savarese, S.; Xiong, C.; Zhong, V.; and Yu, T. 2024. OS-World: Benchmarking Multimodal Agents for Open-Ended Tasks in Real Computer Environments. In *Advances in Neural Information Processing Systems*.
- Yang, A.; Li, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Gao, C.; Huang, C.; Lv, C.; Zheng, C.; Liu, D.; Zhou, F.; Huang, F.; Hu, F.; Ge, H.; Wei, H.; Lin, H.; Tang, J.; Yang, J.; Tu, J.; Zhang, J.; Yang, J.; Yang, J.; Zhou, J.; Zhou, J.; Lin, J.; Dang, K.; Bao, K.; Yang, K.; Yu, L.; Deng, L.; Li, M.; Xue, M.; Li, M.; Zhang, P.; Wang, P.; Zhu, Q.; Men, R.; Gao, R.; Liu, S.; Luo, S.; Li, T.; Tang, T.; Yin, W.; Ren, X.; Wang, X.; Zhang, X.; Ren, X.; Fan, Y.; Su, Y.; Zhang, Y.; Zhang, Y.; Wan, Y.; Liu, Y.; Wang, Z.; Cui, Z.; Zhang, Z.; Zhou, Z.; and Qiu, Z. 2025. Qwen3 Technical Report. *arXiv preprint arXiv:2505.09388*.
- Yao, S.; Shinn, N.; Razavi, P.; and Narasimhan, K. 2025. τ -bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains. In *The Thirteenth International Conference on Learning Representations*.
- Ye, B.; Li, R.; Yang, Q.; Liu, Y.; Yao, L.; Lv, H.; Xie, Z.; An, C.; Li, L.; Kong, L.; et al. 2026. Claw-Eval: Toward Trustworthy Evaluation of Autonomous Agents. *arXiv preprint arXiv:2604.06132*.
- You, R.; Cai, H.; Zhang, C.; Xu, Q.; Liu, M.; Yu, T.; Li, Y.; and Li, W. 2026. Agent-as-a-Judge. *arXiv preprint arXiv:2601.05111*.
- Yue, X.; Ni, Y.; Zhang, K.; Zheng, T.; Liu, R.; Zhang, G.; Stevens, S.; Jiang, D.; Ren, W.; Sun, Y.; et al. 2024. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.
- Z.AI. 2025. GLM-4.7: Advancing the Coding Capability. <https://z.ai/blog/glm-4.7>.
- Z.AI. 2026. ZClawBench: A Benchmark for the Open-Claw Agent Scenario. <https://docs.z.ai/guides/llm/glm-5-turbo/zclawbench-a-benchmark-for-the-openclaw-agent-scenario>.
- Zhang, Y.; Wang, Y.; Zhu, Y.; Du, P.; Miao, J.; Lu, X.; Li, Z.; Qu, X.; Guo, Z.; Shen, Y.; Song, D.; Zhou, H.; Zheng, T.; Wu, X.; Yu, H.; Cai, S.; Lu, Y.; Hao, Y.; Lei, M.; Chen, L.; Zou, K.; Yin, H.; Xu, W.; Jiang, D.; Nie, P.; Liu, J.; Chen, W.; and Allen, K. R. 2026. ClawBench: Can AI Agents Complete Everyday Online Tasks? *arXiv preprint arXiv:2604.08523*.
- Zheng, L.; Chiang, W.-L.; Sheng, Y.; Zhuang, S.; Wu, Z.; Zhuang, Y.; Lin, Z.; Li, Z.; Li, D.; Xing, E. P.; Zhang, H.; Gonzalez, J. E.; and Stoica, I. 2023. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems*.
- Zhou, S.; Xu, F. F.; Zhu, H.; Zhou, X.; Lo, R.; Sridhar, A.; Cheng, X.; Ou, T.; Bisk, Y.; Fried, D.; Alon, U.; and Neubig, G. 2024. WebArena: A Realistic Web Environment for Building Autonomous Agents. In *The Twelfth International Conference on Learning Representations*.
- Zuo, Y.; Qu, S.; Li, Y.; Chen, Z.-R.; Zhu, X.; Hua, E.; Zhang, K.; Ding, N.; and Zhou, B. 2025. MedXpertQA: Benchmarking Expert-Level Medical Reasoning and Understanding. In *Proceedings of the 42nd International Conference on Machine Learning*.

A Details of Experimental Setup

This section complements the experimental setup summarized in the main paper with the details required for reproduction. We focus on inference configuration and the tool & skill suite; in addition, the models used and their per-token prices are listed in Table 7 (§E) and the judge/meta-judge prompts are reproduced verbatim in Figure 8 and Figure 9.

A.1 Inference Configuration

All models are evaluated under the OpenClaw (Steinberger and OpenClaw Contributors 2026) defaults to avoid per-model tuning bias. The random seed is fixed to 42 across all decode calls. Local models are served with vLLM (Kwon et al. 2023) on the NVIDIA RTX PRO 6000 workstation (96 GB VRAM) described in the main paper; no quantization or tensor parallelism is applied. Prefix caching is enabled to amortize the recurring system prompt. The software environment is Ubuntu 24.04.4 LTS, vLLM 0.19.0, OpenClaw 2026.3.20, and CUDA 13.1.

A.2 Tool and Skill Suite

Beyond the default OpenClaw tool suite (file I/O, shell execution, and agent-loop control primitives), every model is given `web_search` (powered by Brave Search) and `web_fetch` (URL-to-markdown rendering); both tools enforce a uniform network policy across models. We additionally integrate 9 Clawhub skills, grouped by function:

- **Self-evolution and logical reasoning:** `ontology` (cross-skill state sharing via a shared key-value store), `self-improving-agent` (injects a short reflective prompt after a failed turn to encourage plan revision), and `skill-vetter` (validates outgoing tool calls against the skill’s input schema before dispatch).
- **Automation capabilities:** `Agent Browser` (headless browser automation via Playwright), `desktop-control-1.0.0` (GUI-level keyboard, mouse, and window control for desktop-automation tasks), and `github` (repository inspection and file retrieval through the GitHub REST API, with session-scoped credentials never exposed to the model).
- **General utilities:** `summarize` (bounded-length multimodal summarization to keep long-horizon trajectories within the context limit), `proactive-agent` (deferred sub-task dispatch within the agent loop), and `humanizer` (draft-to-prose post-processing, applied only when the artifact contract expects a human-readable report).

All tools and skills are loaded under identical visibility and permission boundaries across models: every model sees the same tool list, schema, and permission grants, preventing infrastructure-side bias from confounding the capability comparison.

B Additional Experiments

B.1 Model Evolution

Figure 4 plots the accuracy trajectories of the Qwen and Gemma series among locally deployed open-weight models across the six DAREBench task groups, in order to

examine performance changes under generational evolution. Overall, both Qwen and Gemma follow two relatively clear evolutionary patterns: (1) Under the same architecture, larger parameter scale usually performs better; for example, the Overall accuracy of Qwen3.5-27B exceeds that of Qwen3.5-9B by 15.7 (49.5 – 33.8), while mixture-of-experts (MoE) models, despite often having more total parameters than same-generation dense models, typically achieve slightly lower Overall scores because fewer parameters are activated. Comparisons between the 27B dense and 35B-A3B MoE variants in both Qwen3.5 and Qwen3.6 confirm this pattern (Qwen3.5: 49.5 vs. 47.2; Qwen3.6: 59.8 vs. 56.6). (2) Generational iteration brings steady gains overall; for example, the Overall performance of the Qwen3-VL 8B model (Overall 40.3) already surpasses that of the Qwen2.5-VL 32B model (Overall 39.0), and Gemma-3-27B to Gemma-4-26B-A4B likewise rises from 21.7 to 36.5. Further decomposing by workload shows that generational gains on multimodal tasks are typically larger than those on text-only tasks (e.g., for Qwen3.5-27B→Qwen3.6-27B the average text-only gain is +9.9 while the multimodal gain is +11.2); along the execution-complexity axis, the improvement on Multi-Step+Tools is consistently the smallest, while the gains on Single-Step and Multi-Step are relatively close. This ordering is consistent with the difficulty ranking implied by our workload taxonomy: harder tasks exhibit slower generational progress. This observation also provides indirect support for partitioning workloads by modality and execution complexity.

B.2 Error Pattern Analysis

To further investigate the causes of task failure, we randomly sampled 30 tasks stratified by task composition, collected all models’ failed execution trajectories on these tasks, and conducted a fine-grained analysis of each model’s failure patterns. Through manual review, we found that the primary failure modes can be categorized into the six patterns shown in Table 4. We first prompted Grok 4.5 (SpaceXAI 2026) to read each task description together with the corresponding model execution traces, classify each failure into one of the six error modes (assigning cases outside these categories to Other), and provide a rationale. To ensure accuracy, we further manually audited the classifications; the final results are reported in Table 5.

Among identifiable failures, PPPA accounts for the largest share (88/308, 28.6%), and is substantially more frequent for local models than for API models (62 vs. 26; 38.5% vs. 17.7% of their respective failure pools). This gap indicates that long-horizon planning and contract adherence—finishing within the time limit and writing the required artifact—remain the primary bottleneck, especially for weaker open-weight agents whose tool-calling or file-writing protocols are also more prone to interruption. SRU and VEB are the next most common failure modes, with SRU manifesting as constraint violations or missed aggregations, and VEB as visual evidence bypass. On VEB errors, the API and local counts are nearly identical (18 vs. 17), suggesting that models preferentially rely on web retrieval or prior guesses rather than reading and retrieving information from local images.

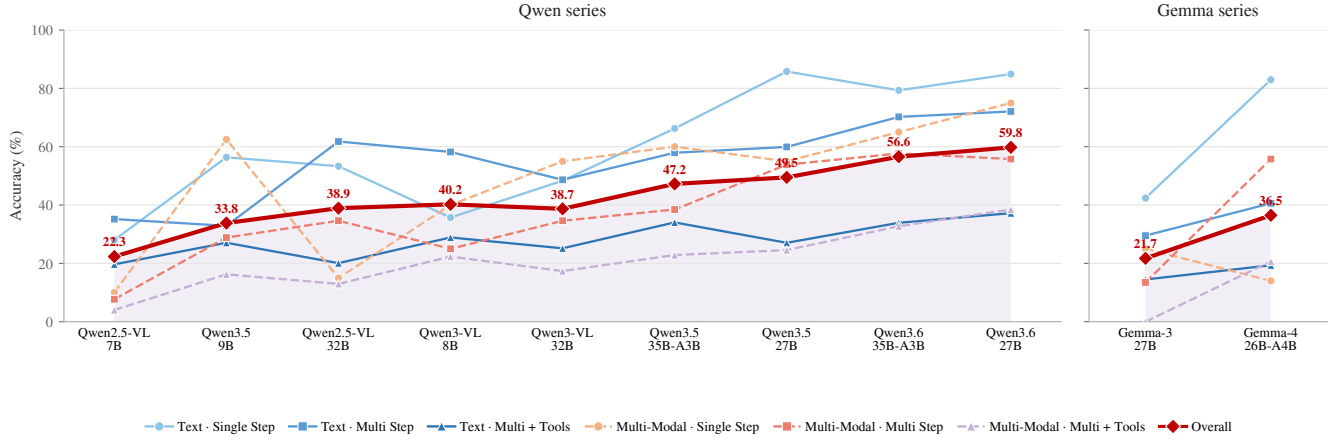


Figure 4: Accuracy trajectories of locally deployed open-weight Qwen and Gemma series across the six task groups. Thin lines denote per-group accuracies; the bold red line and shading denote Overall.

ID	Definition
SRU	Semantic Reasoning & Understanding: violates constraints, misses aggregation, or answers off-question.
TUIE	Tool Use & Information Extraction: insufficient web search/fetch retrieval capability.
PPPA	Path Planning & Protocol Adherence: fails to finish within the time limit or to write the required contract artifact (e.g., <code>answer.txt</code>).
FCR	Fine-grained Chart Reading: inspects the figure but misreads legends, curve names, markers, or numeric values.
VEB	Visual Evidence Bypass: the task depends on an image, yet the model barely looks at it and guesses via web search or priors.
VCS	Visual Common Sense & Entity Recognition: misidentifies the target object, attribute, or person in the image.
OTH	Other: residual error types that cause task failure.

Table 4: Capability-oriented failure taxonomy.

B.3 Local Model Token Usage and Cost

Table 6 reports per-task token consumption and estimated USD cost for the 12 locally deployed open-weight models, where Cost\$/task is computed from each model’s OpenRouter hosted-reference rate. For local models, token usage more directly reflects compute and inference-time cost, while the estimated dollar figures serve as a complementary economic reference for API-style hosting of open-weight models. Relative to the API pool in the main paper, locally served models complete tasks with substantially fewer tokens, especially on Multi-Step+Tools buckets: the strongest local model, Qwen3.6-27B, averages about 62k (Text) and 81k (MM) tokens/task, whereas many commercial API models exceed 200k under the same aggregation. Importantly, this does *not* imply higher reasoning efficiency: as reported in Section B.2, the apparent “low spend” often reflects PPPA-

style aborted or truncated trajectories rather than Pareto-efficient agent performance.

C Details of Evidence-Based Audit

This section supplements the Evidence-Based Audit described in the main paper (§Evaluation Protocol).

LLM-as-a-judge provides the semantic evaluation capacity necessary for open-ended tasks; however, even with manually designed task-specific rubrics, our pilot experiments revealed systematic risks of *judging hallucination* in LLM judges, which has also been reported in related work (You et al. 2026; Gurram 2026). Through careful sampling-based verification, we systematically identify five major recurring hallucination patterns in DAREBench:

1. rubric leakage: the judge mistakes answer information embedded in the grading rubrics for the agent’s execution output.
2. deadlock loop: the agent repeatedly cycles through materially equivalent reasoning or actions without making workspace progress, eventually exhausting the 32K context window and leaving the task incomplete, while the judge still awards positive credit.
3. pseudo-tool-call credulity: the model outputs tool-like text rather than an actually executable tool call.
4. output instability: incomplete or malformed output.
5. timeout-with-credit: timed-out trajectories still receiving positive scores.

Figures 5 and 6 further visualize two real cases that expose these hallucination patterns. To address these patterns, we designed a set of deterministic post-hoc rules that perform high-recall pre-filtering of potential hallucination cases, followed by a second-line adjudication by an independent meta-judge, thereby enabling cost-controllable evaluation and screening. The complete meta-judge pipeline is given in Algorithm 1.

Among the 7,587 task-model evaluations, 3,340 are judge/hybrid calls, from which the pre-filter set $\mathcal{R}$ recalled

Model	Sampled	SRU	TUIE	PPPA	FCR	VEB	VCS	OTH
API Deployment								
Gemini-3.1-Pro	2	1	0	0	0	0	0	1
Claude-Opus-4.8	3	1	0	0	0	1	0	1
GPT-5.5	4	1	0	1	0	0	0	2
Claude-Opus-4.6	3	0	0	0	0	1	0	2
GPT-5.4	5	1	0	1	0	1	0	2
MiMo-V2.5	4	1	0	1	0	1	0	1
Claude-Sonnet-4.6	7	4	0	1	0	0	1	1
Qwen3.6-Plus	6	1	0	0	1	1	0	3
Qwen3.5-Plus	4	2	0	1	0	1	0	0
Doubao-Seed2.0-Pro	8	3	0	0	1	1	0	3
Gemini-3.5-Flash	5	2	0	0	0	0	0	3
GPT-5.4-mini	6	1	0	1	0	1	1	2
MiMo-V2.5-Pro	7	2	1	1	1	1	1	0
DeepSeek-V4-Pro	9	1	0	2	2	1	0	3
Claude-Haiku-4.5	6	2	0	1	1	1	1	0
Doubao-Seed2.0-Mini	9	0	0	3	0	1	1	4
MiMo-V2-Omni	9	1	1	3	0	1	2	1
Kimi-K2.5	5	1	0	1	1	0	1	1
MiMo-V2-Pro	8	3	1	1	1	2	0	0
MiniMax-M2.7	10	2	0	3	2	1	1	1
GLM-5	8	2	0	1	2	1	1	1
DeepSeek-V3.2	10	2	0	2	2	0	1	3
GLM-4.7	9	2	0	2	1	1	0	3
Local Deployment								
Qwen3.6-27B	9	1	0	5	1	0	0	2
Qwen3.6-35B-A3B	9	1	0	5	1	1	1	0
Qwen3.5-27B	13	2	0	4	1	1	2	3
Qwen3.5-35B-A3B	11	3	0	3	0	2	2	1
Qwen3-Coder-30B-A3B	14	5	0	6	0	0	0	3
Qwen3-VL-32B	11	1	0	7	2	0	0	1
Qwen3-VL-8B	11	2	1	3	2	1	2	0
Gemma-4-26B-A4B	15	1	0	5	1	0	0	8
Qwen2.5-VL-32B	13	1	3	4	0	2	0	3
Qwen3.5-9B	16	3	0	5	0	1	2	5
Gemma-3-27B	19	0	0	9	1	2	2	5
Qwen2.5-VL-7B	20	1	3	6	0	7	2	1

Table 5: Fine-grained failure distribution over zero-score trajectories from 35 models on 30 stratified tasks. Models are ordered by overall accuracy on the full benchmark (descending).

182 potential hallucination cases. As reported in the main paper (§LLM Judge Reliability), after a second-line review by Claude-Opus-4.6, 143 were finally confirmed as judging hallucinations.

To ensure the reliability of the meta-judge method and its experimental results, we further invited human experts to validate the meta-judge outcomes. We stratified-sampled 100 execution trajectories from the 3,340 judge/hybrid calls along original score, task group, and deployment mode, and had human experts score them independently. We report the mean absolute score error (MAE):

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |s_i^{\text{judge}} - s_i^{\text{expert}}|, \quad n = 100. \quad (15)$$

Here s_i^{judge} denotes the score assigned by either the primary judge or the meta-judge to the i -th trajectory, and s_i^{expert} denotes the experts’ score. The results show that the MAE between the meta-judge-corrected scores and the human scores is 0.049, approximately half that of the primary judge (0.098),

indicating that the meta-judge correction is not only more accurate in its verdicts but also closer to the experts in the assigned scores.

D Case Study

As additional examples supplementing the main paper’s case study, this section further illustrates how workload-level gaps manifest in concrete agent trajectories. Cases 1 and 2 compare a successful API trajectory against a failed local trajectory, revealing the primary performance differences between strong and weak models, and illustrating cross-step state loss in a tool-mediated text task and visual-symbolic grounding failure in a compact multimodal task. Cases 3–5 focus on hard multimodal tasks that expose common verification failures: chart alignment, chemistry formula bookkeeping, and plate-map option grounding.

Model	Tokens/task								Cost\$/task (OpenRouter rates)								Ctx
	Text				MM				Text				MM				
	SS	MS	MT	Avg	SS	MS	MT	Avg	SS	MS	MT	Avg	SS	MS	MT	Avg	
Local Deployment																	
Qwen3.6-27B	47k	52k	83k	62k	58k	54k	128k	81k	0.037	0.034	0.056	0.041	0.037	0.041	0.084	0.055	256K
Qwen3.6-35B-A3B	53k	57k	86k	66k	57k	68k	114k	81k	0.010	0.009	0.014	0.011	0.009	0.012	0.018	0.013	256K
Qwen3.5-27B	39k	64k	85k	68k	49k	53k	126k	78k	0.016	0.021	0.028	0.023	0.016	0.020	0.041	0.026	256K
Qwen3.5-35B-A3B	35k	45k	78k	55k	61k	62k	105k	77k	0.011	0.012	0.021	0.015	0.016	0.017	0.027	0.021	262K
Qwen3-VL-8B	43k	42k	152k	79k	48k	42k	78k	57k	0.009	0.008	0.028	0.015	0.009	0.008	0.014	0.010	256K
Qwen2.5-VL-32B	24k	31k	45k	35k	32k	25k	8k	21k	0.005	0.006	0.009	0.007	0.007	0.005	0.002	0.004	128K
Qwen3-VL-32B	114k	98k	184k	129k	61k	188k	104k	123k	0.018	0.016	0.030	0.021	0.010	0.030	0.017	0.020	256K
Gemma-4-26B-A4B	111k	53k	54k	60k	46k	105k	202k	123k	0.008	0.004	0.004	0.004	0.004	0.008	0.015	0.009	128K
Qwen3.5-9B	50k	60k	110k	75k	61k	59k	109k	77k	0.005	0.006	0.011	0.008	0.006	0.006	0.011	0.008	256K
Qwen2.5-VL-7B	29k	31k	125k	62k	35k	164k	108k	108k	0.006	0.006	0.025	0.012	0.007	0.033	0.022	0.022	128K
Gemma-3-27B	43k	36k	50k	42k	59k	34k	9k	32k	0.003	0.003	0.004	0.003	0.005	0.003	<0.001	0.003	128K
Qwen3-Coder-30B-A3B	65k	138k	182k	144k	—	—	—	—	0.006	0.007	0.011	0.008	—	—	—	—	160K

Table 6: Per-model token consumption and dollar cost per task for locally deployed open-weight models on DAREBench. Notation and ranking follow the main paper.



Gemma-3-27B (Pseudo Tool Call)

★ T1: `read("image.png")` → *image read successfully*
★ T2: `web_search("克劳塞维茨 死亡原因")` → 5 hits (Wikipedia, biography, WeRead...)
★ T3: **directly response** [`write(path="answer.txt", content="克劳塞维茨卒于霍乱。")`]
Reply in the format of a tool call, but do not actually initiate a tool call

LLM Judge (Undetected File Write Incompletion)

The answer '克劳塞维茨卒于霍乱。' is semantically equivalent to the reference answer '霍乱' and factually correct — Carl von Clausewitz died of cholera in 1831. The response correctly identifies the disease without extraneous information.

Figure 5: Judge hallucination example A (*pseudo-tool-call credulity*). **Task:** What disease did the author of the book in the image die from? Write your answer to ‘answer.txt’ in the workspace. The model embeds a factually correct answer inside a tool-like `write(...)` string but never executes a real file write; the LLM judge overlooks the missing workspace artifact and still awards full credit.

place	code	area (km 2)	population	most spoken language
alldays	90901	11.75	385	northern sotho
bahanawa	90902	390.17	19068	northern sotho
...	...	...	...	...
remainder of the municipality	90904	2198.72	5539	northern sotho

Qwen2.5-VL-7B (Rubric Leakage)

★ T1: **directly response** No anomalies are detected in the table.
The model gives up and fails to provide the correct answer.

LLM Judge

(mistakenly treating the rubric as the model's response)

Rubric Prompt for LLM Judge:
Evaluate the model's answer against the reference answer below...
Reference Answer: The two anomalies are the extremely large area for 'dichoeng' (100000.0) and the extremely small population for 'ramutla' (1)...

Judge Response: The agent correctly identified the two anomalies (Dichoeng's area = 100000.0 and Ramutla's population = 1) and described them accurately, matching the reference answer's key facts. However, the reference explicitly states 'the two anomalies are...', while the agent wrote 'Two anomalies are detected: row 6 with Dichoeng...

Figure 6: Judge hallucination example B (*rubric leakage*). **Task:** Read the table. Can you identify any locations within the table whose area or population values significantly deviate from the patterns observed in other locations? The model fails and answers “No anomalies are detected,” yet the LLM judge mistakes the reference answer embedded in the grading rubric for the agent’s response and incorrectly scores the trajectory as correct.

Algorithm 1: Evidence-Based Audit

Input: Primary scores $\{s_{m,i}\}$ ($s_{m,i} > 0$); trajectories $\{\text{traj}_i\}$; prompts $\{p_i\}$; artifacts $\{\mathcal{W}_i^*\}$; rubrics $\{\text{rubric}_i\}$; rules $\mathcal{R}$

Output: Audited scores $\{s_{m,i}^{\text{audit}}\}$

foreach *judge/hybrid call* i **do**

```
 $\phi_i \leftarrow \{$   
  ASSTCHARS(traji),  
  HASWRITE(traji),  
  HASTOOL(traji),  
  TOKTOTAL(traji)  
 $\}$ ;  
/* flag positive scores with weak  
  execution evidence */  
if PREFILTER( $\phi_i, s_{m,i}, \mathcal{R}$ ) = True then  
   $v_i \leftarrow$   
    METAJUDGE( $p_i, \text{traj}_i, \mathcal{W}_i^*, \text{rubric}_i, s_{m,i}$ );  
    if  $v_i$  = hallucination then  
      |  $s_{m,i}^{\text{audit}} \leftarrow 0$ ;  
    else  
      |  $s_{m,i}^{\text{audit}} \leftarrow s_{m,i}$ ;  
else  
  |  $s_{m,i}^{\text{audit}} \leftarrow s_{m,i}$ ;  
return  $\{s_{m,i}^{\text{audit}}\}$ ;
```

D.1 Case 1 — A Multi-Hop Academic Author Retrieval Workflow

Case 1: Multi-Hop Author Retrieval (Text Multi+Tools)

Task. Of the authors in the first article of the 65th edition of the Journal of Artificial Intelligence Research, who was the most cited on Google Scholar in 2019?

Gemini 3.1 Pro (Correct, 1.0)

```
T1: web_search("Journal of Artificial  
  ↳ Intelligence Research Volume 65")  
-> JAIR Vol 65 = 2019 (from jair.org  
  result)  
-> dblp link spotted (from result #4)  
T2: web_fetch("https://dblp.org/db/  
  ↳ journals/jair/jair65.html")  
-> Full TOC scraped; 1st article: "On  
  Overfitting..." (pp. 1-30)  
-> Authors: V. Francois-Lavet, G.  
  Rabusseau, J. Pineau, D. Ernst, R.  
  Fonteneau  
Confirmed correct article and extracted 5  
  target authors successfully  
T3: web_fetch("https://doi.org/10.1613/  
  ↳ jair.1.11478")  
-> Failed (406 Not Acceptable) fallback  
  unnecessary, TOC sufficient  
T4: web_search("Vincent Francois-Lavet ...  
  ↳ citations")  
-> Confirmed 5 authors on  
  arXiv/JAIR/ResearchGate  
T5-T14: web_search x10 (per author +  
  ↳ reconfirmations)
```

```
-> J. Pineau: 41,695 [highest] clear  
  winner identified  
-> D. Ernst: 25,039; R. Fonteneau: 1,859;  
  G. Rabusseau: 1,439  
-> V. Francois-Lavet: no Scholar profile  
T15: write_output("Joelle Pineau")  
T16: read_output()  
-> Verified: 'Joelle Pineau' Self-check  
  passed
```

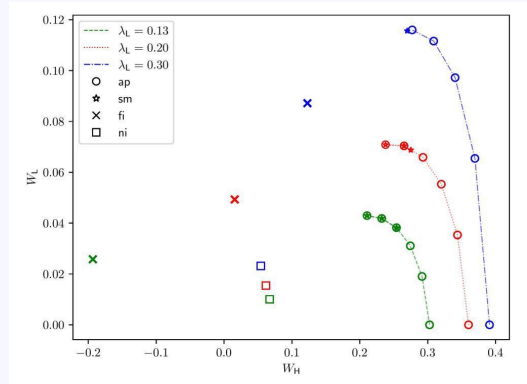
Qwen3-VL-32B (Incorrect, 0.0)

```
T1-3: web_search x3 (JAIR 65th edition  
  ↳ first article authors)  
-> JAIR homepage, Wikipedia, generic  
  publishers  
-> "Optstream..." (pp. 423-456) spotted;  
  Authors: F. Fioretto, P. Van  
  Hentenryck  
Mistakenly identified the last article as  
  the first  
T4: web_fetch("https://www.jair.org/index.  
  ↳ php/jair/issue/view/1153")  
-> Full TOC scraped; 1st article: "On  
  Overfitting..." (pp. 1-30)  
-> Authors: V. Francois-Lavet, G.  
  Rabusseau, J. Pineau, D. Ernst, R.  
  Fonteneau  
Correct first-article info retrieved, but  
  prior error not overwritten  
T5: web_search("F. Fioretto P. Van  
  ↳ Hentenryck Google Scholar citations  
  ↳ 2019")  
-> P. Van Hentenryck: 31,983 citations;  
  F. Fioretto: 3,548 citations  
Misled by early erroneous search: queried  
  authors of the WRONG article (T1-3)  
  instead of target article authors (T4)  
T6: write_output("Pascal Van Hentenryck")
```

D.2 Case 2 — Academic Diagram Understanding

Case 2: Academic Diagram Understanding (Multi-Modal Single Step)

Task. What is the name of the line that is the furthest away from its fi value from the W-H axis?



Gemini 3.1 Pro (Correct, 1.0)

Reasoning: Examining Graph Data: zeroing

↳ in on the specific line furthest
 ↳ from
 the W-H axis. The legend provides lambda
 ↳ values: 0.13, 0.20, 0.30.
 Visually assess which (green dashed, red
 dotted, blue dash-dot) is most distant
Reasoning: Pinpointing The Answer:
 ↳ calculating distances for each line
 ↳ using the
 'fi' points as reference along the W-H
 ↳ axis.
 -> green line ($\lambda_L = 0.13$) appears most
 distant from its 'fi' value

Answer: $\lambda_L = 0.13$

Qwen3-VL-32B (Incorrect, 0.0)

Reasoning: The name of the line that is
 ↳ the furthest away from its fi value
 from the W-H axis is "fi".
 -> data points marked 'x' (representing
 "fi") at W-H values of approx. -0.2
 and 0.12

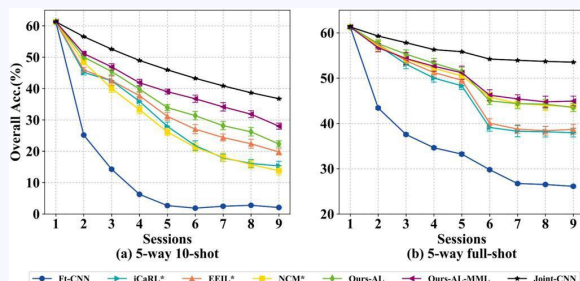
Confuses an axis/marker label with the
 requested curve identity

Answer: fi

D.3 Case 3 — Shared Chart-Grounding Failure

Case 3: A Chart Item That All Models Miss

Task. "Which model shows a greater decline in accuracy from Session 1 to Session 9 in the 5-way full-shot scenario?" The chart shows continual-learning curves for Ft-CNN, iCaRL, EEIL, NCM, Ours-AL, Ours-AL-MML, and Joint-CNN. Reference answer: Joint-CNN.



Gemini 3.1 Pro (Incorrect, 0.0)

T1: read_image("chart.png")
 -> Image loaded, 7 curves visible
Reasoning: scanning for the steepest
 visible decline.

T2: write_output("Ft-CNN")
 Selected a plausible visual drop, but did
 not match the reference curve
 comparison from Session 1 to Session
 9.

Qwen3-VL-32B (Incorrect, 0.0)

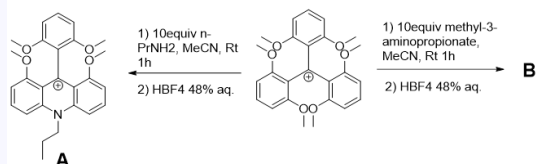
T1: read_image("chart.png")
 -> Image loaded; legend and curves
 visible
Reasoning: compares apparent drops among
 the curve labels.
T2: write_output("Ft-CNN")

Converges to the same wrong legend entry
 as several frontier APIs.

D.4 Case 4 — Image-Grounded Chemistry Formula Inference

Case 4: A Chemistry Item Where Formula Bookkeeping Fails

Task. A Humanity's Last Exam organic-chemistry item asks for the molecular formula of compound B. The prompt describes reactions of tris(2,6-dimethoxyphenyl)methylmethyl cation with either *n*-propanol or methyl-3-aminopropionate, and the diagram specifies the product structure. Reference answer: C₂₃H₂₂N₂O₄⁺.



MiMo-V2.5 (Incorrect, 0.0)

T1: read_image("hle_07_image.jpg")
 -> image parsed; identifies acridinium
 core and substituents

Reasoning: derives product B from the
 right-hand pathway.

T2: write_output("C₂₉H₃₂N₂O₇⁺")
 Formula does not match the reference
 C₂₃H₂₂N₂O₄⁺.

Qwen3-VL-32B (Incorrect, 0.0)

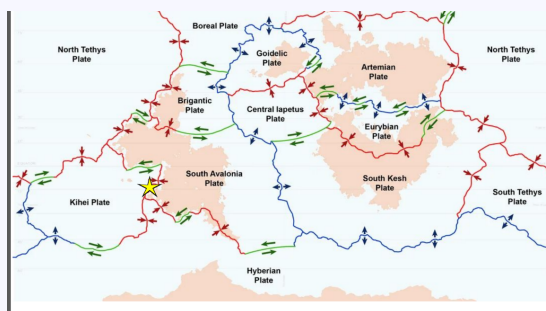
T1: read_image("hle_07_image.jpg")
 -> describes the reaction diagram and
 attempts atom accounting
Reasoning: counts atoms after substituent
 transfer.

T2: write_output("C₃₁H₃₄N₂O₈")
 Wrong atom count and missing charge;
 reference is C₂₃H₂₂N₂O₄⁺.

D.5 Case 5 — Spatial Plate-Boundary Reasoning

Case 5: A Plate-Map Item Where Option Grounding Fails

Task. An HLE structural-geology item shows a hypothetical tectonic map and asks: "Along which plate boundary might we expect the longest range of the tallest mountains on the planet shown above? Assume similar tectonic plate geology to Earth." The model must choose from nine named plate-boundary options. Reference answer: D. South Kesh Plate and Eurybian Plate.



Claude Opus 4.6 (Incorrect, 0.0)

```
T1: read_image("hle_02_image.jpg")
-> reads the map and applies the
    mountain-building criterion
Reasoning: tallest ranges should form
    along strong convergent/collision
    boundaries.
T2: write_output("G. Artemian Plate and
    ↳ Eurybian Plate")
Geologic rule is plausible, but the
    selected option is not the reference
    boundary D.
```

Qwen3.6-35B-A3B (Incorrect, 0.0)

```
T1: read_image("hle_02_image.jpg")
-> identifies candidate boundaries and
    compares plate interactions
Reasoning: searches for the most
    mountain-building boundary.
T2: write_output("H")
Chooses a different boundary option;
    reference is D, South Kesh Plate and
    Eurybian Plate.
```

E OpenRouter Pricing

Table 7 lists the per-million-token rates used for the USD cost columns in the main paper and Table 6. Per-task cost follows the four-term formula $\text{cost} = \frac{1}{10^6}(u_{\text{in}}\alpha_{\text{in}} + u_{\text{out}}\alpha_{\text{out}} + u_{\text{cr}}\alpha_{\text{cr}} + u_{\text{cw}}\alpha_{\text{cw}})$, where $u_{\text{in}}, u_{\text{out}}, u_{\text{cr}}, u_{\text{cw}}$ are the input, output, cache-read, and cache-write token counts and α the matching unit prices in Table 7; when a model exposes no separate cache-read or cache-write rate (“—”), the corresponding tokens are billed at the input price. All models use OpenRouter listing prices, except for Doubao-Seed2.0-Pro, MiMo-V2-Omni, and MiMo-V2-Pro (marked †), which are not listed on OpenRouter and are priced using their official provider rates.

F Details of DAREBench

F.1 Benchmark Composition

Figure 7 summarizes the composition of DAREBench. The suite comprises 233 tasks curated from 22 publicly available source benchmarks through the three-stage (feasibility, difficulty, representativeness) curation procedure described in the main paper. Tasks are partitioned into six workload groups along the input-modality axis (Text vs. Multi-Modal) and the execution-form axis (Single-Step, Multi-Step, Multi-Step+Tools), yielding group sizes of 20, 87, 55, 20, 26, and 25 for Text-SS, Text-MS, Text-MT, MM-SS, MM-MS, and MM-MT, respectively. The per-benchmark task counts, group assignments, and brief descriptions of each source benchmark are provided in §F.3.

F.2 Benchmark Scoring

Scoring-mode assignment. Each task is assigned exactly one of three scoring modes during benchmark construction, according to the verifiability of its artifact contract: tasks with deterministic answers or artifacts checkable by rule-based procedures use *automated* scoring; tasks with open-ended

outputs or semantic constraints that cannot be reliably verified by deterministic rules use *LLM-based* (judge) scoring; and tasks whose contracts contain both rule-verifiable and semantically judged criteria use *hybrid* scoring with fixed per-task weights.

Scoring-mode distribution. Table 8 reports the resulting distribution. Overall, 125 tasks (53.6%) are scored automatically, 73 tasks (31.3%) by the LLM judge, and 35 tasks (15.0%) in hybrid mode. The assignment is workload-dependent: multimodal single-step tasks (MM-SS) are exclusively judge-scored because their answers require semantic equivalence assessment against the reference, multimodal tool-use tasks (MM-MT) are exclusively hybrid, and text multi-step tasks (Text-MS, Text-MT) are predominantly automated. All scores produced with judge involvement (judge and hybrid modes) additionally pass through the evidence-based audit described in the main paper and detailed in §C.

F.3 Source Benchmarks

DAREBench curates 233 tasks from 22 widely used open-source benchmarks spanning reasoning, long-context understanding, multi-hop and search-grounded QA, instruction following, tabular and numerical reasoning, vertical domains (finance, law, medicine), code and shell workflows, desktop control, visual question answering, and multimodal search. Table 9 summarizes each source benchmark, its task count, and the workload group to which it is mapped under our modality×execution-complexity taxonomy (defined in the main paper’s Methodology section). Brief descriptions of each source benchmark follow.

- **LogiQA** (Liu et al. 2020) is a reading-comprehension benchmark for logical reasoning, where each instance provides a short passage and a multiple-choice question whose correct answer requires identifying the logical structure (premise, conclusion, assumption) of the argument.
- **GPQA (Graduate-Level Google-Proof Q&A)** (Rein et al. 2024) contains Ph.D.-level science questions in biology, chemistry, and physics that are resistant to web search; answering them correctly requires expert-level knowledge and multi-step reasoning rather than retrieval.
- **AdvancedIF** (He et al. 2025) targets complex instruction following with dozens of verifiable constraints per instruction (e.g., length, keyword, formatting, and structural requirements), testing a model’s ability to satisfy all constraints simultaneously in a single response.
- **Bamboogle** (Press et al. 2023) is a multi-hop question-answering dataset in which each question requires chaining two or more Wikipedia facts; the answer is not directly retrievable and demands cross-document reasoning.
- **SimpleQA** (Wei et al. 2024) is a short-form fact-seeking benchmark where each question has a single, unambiguous answer; it emphasizes precision and resistance to confabulation on entities, dates, and numerical facts.
- **LongBench** (Bai et al. 2025c) is a long-context benchmark covering summarization, retrieval, and reasoning over documents of up to tens of thousands of tokens; it evaluates a model’s ability to locate and synthesize information dispersed across long inputs.

Model	Provider	Pool	Input	Output	Cache Read	Cache Write
<i>API Deployment</i>						
Claude-Opus-4.8	Anthropic	API	5.00	25.00	0.50	6.25
Claude-Opus-4.6	Anthropic	API	5.00	25.00	0.50	6.25
Claude-Sonnet-4.6	Anthropic	API	3.00	15.00	0.30	3.75
Claude-Haiku-4.5	Anthropic	API	1.00	5.00	0.10	1.25
GPT-5.5	OpenAI	API	5.00	30.00	0.50	—
GPT-5.4	OpenAI	API	2.50	15.00	0.25	—
GPT-5.4-mini	OpenAI	API	0.75	4.50	0.075	—
Gemini-3.1-Pro	Google AI Studio	API	2.00	12.00	0.20	0.375
Gemini-3.5-Flash	Google AI Studio	API	1.50	9.00	0.15	0.083
Qwen3.6-Plus	Alibaba Cloud Int	API	0.50	3.00	0.05	—
Qwen3.5-Plus	Alibaba Cloud Int	API	0.40	2.40	0.04	—
Doubao-Seed2.0-Pro [†]	Volcengine	API	0.444	2.222	0.089	—
Doubao-Seed2.0-Mini	Seed	API	0.10	0.40	—	—
GLM-5	Z.ai	API	1.00	3.20	0.20	—
GLM-4.7	Z.ai	API	0.60	2.20	0.11	—
Kimi-K2.5	Moonshot AI	API	0.60	3.00	0.10	—
MiniMax-M2.7	MiniMax	API	0.30	1.20	0.06	—
DeepSeek-V4-Pro	DeepSeek	API	0.435	0.87	0.0036	—
DeepSeek-V3.2	Alibaba Cloud Int	API	0.3705	1.112	0.0741	—
MiMo-V2.5	Xiaomi	API	0.14	0.28	0.0028	—
MiMo-V2.5-Pro	Xiaomi	API	0.435	0.87	0.0036	—
MiMo-V2-Omni [†]	Xiaomi	API	0.14	0.28	0.0028	—
MiMo-V2-Pro [†]	Xiaomi	API	0.435	0.87	0.0036	—
<i>Local Deployment (hosted-reference rates)</i>						
Qwen3.6-27B	Alibaba Cloud Int	Local	0.60	3.60	—	—
Qwen3.6-35B-A3B	AkashML	Local	0.14	1.00	—	—
Qwen3.5-27B	Alibaba Cloud Int	Local	0.30	2.40	—	—
Qwen3.5-35B-A3B	Alibaba Cloud Int	Local	0.25	2.00	—	—
Qwen3.5-9B	SiliconFlow	Local	0.10	0.15	—	—
Qwen3-Coder-30B-A3B	Alibaba Cloud Int	Local	0.045	2.25	—	—
Qwen3-VL-32B	Alibaba Cloud Int	Local	0.16	0.64	—	—
Qwen3-VL-8B	Alibaba Cloud Int	Local	0.18	0.70	—	—
Gemma-4-26B-A4B	DeepInfra	Local	0.07	0.34	—	—
Gemma-3-27B	DeepInfra	Local	0.08	0.16	—	—
Qwen2.5-VL-32B	Alibaba Cloud Int	Local	0.20	0.60	—	—
Qwen2.5-VL-7B	Alibaba Cloud Int	Local	0.20	0.20	—	—

Table 7: Per-million-token prices (USD/1M tokens) for all 35 evaluated models. **Provider** indicates the inference backend from which the rate is sourced. “—” denotes that the provider does not expose a separate cache-read or cache-write rate. Entries marked [†] use the official provider rate (model not listed on OpenRouter).

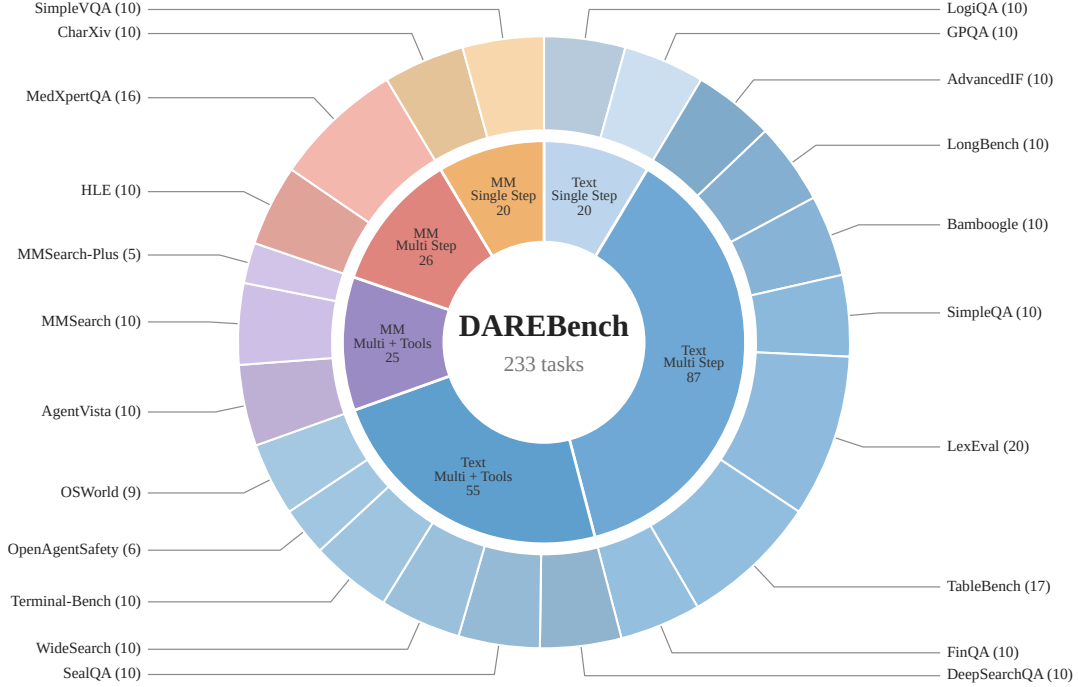


Figure 7: Task composition of DAREBench. The inner ring partitions the 233 tasks along the 2×3 modality $\times$ execution-complexity taxonomy; the outer ring shows the 22 contributing source benchmarks. Segment sizes are proportional to task counts.

Group	#Tasks	Auto	Judge	Hybrid
Text-SS	20	10	0	10
Text-MS	87	64	23	0
Text-MT	55	35	20	0
MM-SS	20	0	20	0
MM-MS	26	16	10	0
MM-MT	25	0	0	25
Total	233	125	73	35

Table 8: Distribution of the three scoring modes across the six DAREBench workload groups. Text/MM = Text/Multi-Modal; SS/MS/MT = Single-Step / Multi-Step / Multi-Step+Tools. Auto = automated (rule-based) scoring; Judge = LLM-based scoring; Hybrid = weighted combination of the two.

- **LexEval** (Li et al. 2024) is a comprehensive legal-domain evaluation suite covering legal knowledge, statute comprehension, and legal reasoning; it tests domain-specific terminology and normative reasoning ability.
- **TableBench** (Wu et al. 2025) is a table-understanding benchmark with diverse table structures and question types (lookup, aggregation, comparison, trend analysis), requiring both precise cell localization and multi-step numerical reasoning.

- **FinQA** (Chen et al. 2021) is a financial question-answering dataset over earnings reports and financial tables; answering requires programmatic reasoning (e.g., computing ratios, growth rates) over tabular data accompanied by natural-language context.
- **DeepSearchQA** (Gupta et al. 2026) is a deep search benchmark featuring hand-crafted, multi-step information-seeking tasks across 17 fields; unlike single-answer retrieval, it evaluates comprehensiveness and the ability to synthesize evidence from multiple retrieved sources.
- **WideSearch** (Wong et al. 2025) probes broad-scope information aggregation: each task requires gathering and reconciling information from many independent sources, testing the breadth and coverage of a search-augmented agent.
- **Seal0** (Pham et al. 2026) is drawn from the SealQA benchmark for search-augmented language models; its core subset (Seal-0) contains questions specifically curated to be challenging for frontier models, often involving ambiguity, conflicting evidence, or noisy search results.
- **TerminalBench2** (Merrill et al. 2026) is a command-line interface benchmark in which the agent must accomplish realistic system-administration, data-processing, and development tasks in a Linux terminal, requiring correct sequencing of shell commands and file operations.
- **OpenAgentSafety** (Vijayvargiya et al. 2026) evaluates open-world agent safety through scenarios in which an un-

Benchmark	Domain / Capability	#Tasks	Group	Scoring
<i>Text Single-Step (Text-SS)</i>				
LogiQA	Reading comprehension with logical reasoning	10	Text-SS	Auto
GPQA	Graduate-level science multiple-choice	10	Text-SS	Hybrid
<i>Text Multi-Step (Text-MS)</i>				
AdvancedIF	Complex multi-constraint instruction following	10	Text-MS	Judge
Bamboogle	Multi-hop question answering	10	Text-MS	Auto
SimpleQA	Short-form fact-seeking QA	10	Text-MS	Auto
LongBench	Long-context understanding and retrieval	10	Text-MS	Judge
LexEval	Legal-domain evaluation	20	Text-MS	Auto
TableBench	Table understanding and numerical reasoning	17	Text-MS	Auto (14) / Judge (3)
FinQA	Financial question answering over tables	10	Text-MS	Auto
<i>Text Multi-Step+Tools (Text-MT)</i>				
DeepSearchQA	Deep multi-step search-grounded QA	10	Text-MT	Judge
WideSearch	Broad-scope information aggregation search	10	Text-MT	Auto
Seal0	Search-augmented reasoning on ambiguous queries	10	Text-MT	Judge
TerminalBench2	Realistic CLI / shell task execution	10	Text-MT	Auto
OpenAgentSafety	Open-world agent safety scenarios	6	Text-MT	Auto
OSWorld	Desktop automation and control	9	Text-MT	Auto
<i>Multi-Modal Single-Step (MM-SS)</i>				
SimpleVQA	Short-form visual fact-seeking QA	10	MM-SS	Judge
HLE	Humanity’s Last Exam (cross-discipline)	10	MM-SS	Judge
<i>Multi-Modal Multi-Step (MM-MS)</i>				
CharXiv	Academic figure and chart understanding	10	MM-MS	Judge
MedXpertQA	Medical expert-level multimodal QA	16	MM-MS	Auto
<i>Multi-Modal Multi-Step+Tools (MM-MT)</i>				
AgentVista	Ultra-challenging multimodal agent tasks	10	MM-MT	Hybrid
MMSearch	End-to-end multimodal search	10	MM-MT	Hybrid
MMSearchPlus	Provenance-aware multimodal search	5	MM-MT	Hybrid
Total		233		

Table 9: Source benchmarks underlying DAREBench. #Tasks: number of tasks retained after the three-stage curation. Group: the workload group each benchmark is mapped to under the modality×execution-complexity taxonomy (Text/MM = Text/Multi-Modal; SS/MS/MT = Single-Step / Multi-Step / Multi-Step+Tools). Scoring: the scoring mode(s) used by the benchmark’s tasks (Auto = automated, Judge = LLM-based, Hybrid = weighted combination; see §F.2).

constrained agent may take harmful, reckless, or policy-violating actions; tasks test whether the agent completes the goal while respecting safety constraints.

- **OSWorld** (Xie et al. 2024) is a desktop-environment benchmark where the agent controls a virtual operating system (files, applications) to accomplish real-world computer tasks; it demands pixel-level perception, cross-application planning, and long-horizon tool use.

- **SimpleVQA** (Wang et al. 2025) is a short-form visual fact-seeking benchmark in which each question is grounded in a single image and has one correct answer; it isolates visual entity recognition and fine-grained perception from multi-hop reasoning.

- **HLE (Humanity’s Last Exam)** (Center for AI Safety, Scale AI, and HLE Contributors Consortium 2026) is a cross-discipline expert-level benchmark spanning mathematics, humanities, science, and professional domains; questions are designed to be resistant to web search and require deep reasoning, often grounded in images, diagrams, or specialized notation.

- **CharXiv** (Wang et al. 2024) is a chart-understanding benchmark built from real arXiv figures; it asks questions that require reading legends, curves, axes, and annotations, testing fine-grained visual reasoning beyond simple OCR or caption matching.

- **MedXpertQA** (Zuo et al. 2025) is a medical expert-level multimodal QA benchmark covering clinical cases, pathology images, and diagnostic reasoning; it evaluates whether a model can reach expert-level conclusions from multimodal medical evidence.

- **AgentVista** (Su et al. 2026) is a benchmark for generalist multimodal agents on realistic, ultra-challenging tasks spanning 25 sub-domains; each task requires long-horizon, interleaved tool use grounded in rich visual evidence, testing perception, planning, and tool-use integration jointly.

- **MMSearch** (Jiang et al. 2025) is an end-to-end multimodal search benchmark in which the agent must query, retrieve, and synthesize information that is distributed across both text and images, requiring multimodal query formulation and cross-modal evidence aggregation.

- **MMSearchPlus** (Tao et al. 2026) extends multimodal search with a provenance-aware evaluation: each task demands fine-grained multimodal reasoning while preventing shortcut answers, and correctness is judged by whether the agent cites and correctly interprets the originating evidence.

G Prompts

This section documents the two prompt templates used in DAREBench’s grading pipeline: the LLM-as-a-Judge grading prompt (Figure 8) and the meta-judge auditing prompt (Figure 9). Since all our experiments are conducted under the unified OpenClaw runtime, the system prompt used in the experiments is the default OpenClaw system prompt and is therefore not reported here.

LLM-as-a-Judge Prompt Template

You are a grading function. Your ONLY job is to output a single JSON object.

CRITICAL RULES:

- Do NOT use any tools (no Read, Write, exec, or any other tool calls)
- Do NOT create files or run commands
- Do NOT write any prose, explanation, or commentary outside the JSON
- Respond with ONLY a JSON object -- nothing else

Be a strict evaluator. Reserve 1.0 for genuinely excellent performance.

An average acceptable completion should score around 0.6-0.7.

Deduct points for unnecessary steps, verbose output, and inefficient tool usage.

Task

{task_prompt}

Expected Behavior

{expected_behavior}

Agent Transcript (summarized)

{transcript_summary}

Grading Rubric

{rubric}

Score each criterion from 0.0 to 1.0.

Respond with ONLY this JSON structure (no markdown, no code fences, no extra text):

{"scores": {"criterion_name": 0.0}, "total": 0.0, "notes": "brief justification"}

Figure 8: LLM-as-a-Judge prompt template. Field placeholders are denoted in curly braces.

Meta-Judge System Prompt

You are a meta-judge auditor for an AI agent benchmark. Your job is to determine whether a grading judge hallucinated when scoring an agent's work.

You will be given:

1. The task prompt (what the agent was asked to do)
2. The agent's session transcript (what the agent actually did)
3. The judge's score and notes (how the judge graded the agent)
4. Metadata: grading_type, original score, assistant output characters

Your task: Determine if the judge's positive score is justified by the agent's actual work, or if the judge hallucinated (fabricated evidence of work that didn't happen).

Hallucination Patterns (choose one if verdict is "hallucination"):

- rubric-leakage: The judge cites answer text not present in the transcript, apparently copying rubric/ground-truth information.
- context-overflow-confabulation: The agent hit the serving context limit and left an empty /truncated final turn; the judge then fabricated completion.
- pseudo-tool-call-credulity: The agent wrote tool-like text rather than an executable tool call, and the judge credited it as completed.
- workspace-gt-leakage: A workspace-side leakage bug exposed ground-truth or prior-run output and the judge credited it.
- timeout-with-credit: A timed-out trajectory received positive judge credit for unfinished work.
- output-instability: Small-model malformed, unstable, repetitive, or incomplete artifacts were over-interpreted as answers.
- mc-rubric-leak: For multiple-choice tasks, the judge copied the correct option from the rubric instead of the agent trace.
- mc-parse: The judge mis-parsed an MC answer format and credited the wrong choice.
- mc-hedge: The agent hedged or mentioned several MC choices and the judge credited the favorable one.
- mixed: Several taxonomy causes apply and no single one dominates.
- other: A hallucination pattern not covered above.

Output Format (JSON only, no markdown):

```
{
  "verdict": "hallucination" | "legitimate",
  "pattern": "<one of the patterns above, or null if legitimate>",
  "confidence": "high" | "medium" | "low",
  "reasoning": "<1-3 sentence explanation>"
}
```

Figure 9: Meta-judge system prompt.