EVOCOWORK: A BENCHMARK FOR REAL-WORLD OFFICE TASKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Office workflows are a demanding testbed for self-improving agents because their deliverables must be correct, natively editable, and mutually consistent. A central question is whether gains on feedback tasks reflect reusable procedures rather than task-specific fixes. We introduce EVOCOWORK, a 240-task benchmark for evaluating harness-level recursive self-improvement across four office modalities, with model weights held fixed. The benchmark covers presentations, documents, spreadsheets, and cross-office workflows, and scores submitted artifacts for content, native structure, formulas, cross-file consistency, and visual quality where needed. Its protocol separates feedback, revision selection, and final testing: methods learn from single-modality tasks, then are evaluated on unseen single-modality tasks and cross-office workflows under a shared initial harness and explicit budgets. In static evaluations of two models and four harnesses, the best configuration scores 42.25 overall and 30.62 on cross-office tasks. Starting from the same MSA harness with GLM 5.3, ACE and AHE improve in-domain scores by 4.90 and 2.64 points, respectively, but change cross-office scores by only $+0.03$ and -1.72. Thus, gains on the file types that provide feedback do not necessarily transfer to coordinated workflows. EVOCOWORK makes this distinction explicit while recording the capability, transfer, and cost of harness evolution.

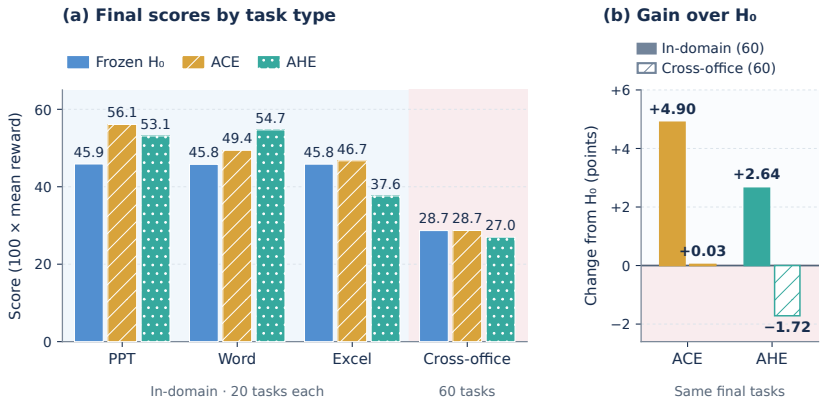


Figure 1: Final scores and evolution gains with GLM 5.3 and a shared MSA seed. (a) Frozen H_0, ACE, and AHE on 20 tasks per single-modality track and 60 cross-office tasks. (b) Changes from H_0 on the 60 in-domain tasks and 60 cross-office tasks, matching Table 4. Both methods improve the in-domain average, while cross-office gains are near zero for ACE and negative for AHE. Bars summarize the fixed final task sets from one run per method.

1 INTRODUCTION

Office agents often carry out related work: they update a workbook, use its figures in a report, and prepare a presentation from the same material. Professional workflow benchmarks test this

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

work across files and applications (Vidgen et al., 2026; Tang et al., 2026), and their feedback can guide how an agent inspects files, uses tools, and checks outputs. Cross-office workflows are the sharpest instance: their deliverables must stay natively structured and mutually consistent, which distinguishes a harness that learned reusable procedures from one that merely fixed the tasks that generated the feedback. These behaviors depend on the harness—the instructions, tools, context, and execution logic around the model (Lee et al., 2026). We study harness-level recursive self-improvement (RSI) by repeatedly revising this layer while model weights stay fixed, and test whether the revisions make later office work more reliable.

Existing benchmarks provide complementary ways to assess this capability. Office benchmarks measure workflow execution and deliverable quality, including requirements that span several files (Wang et al., 2024; Zhou et al., 2026; Tang et al., 2026). Harness-evolution benchmarks test revisions on held-out tasks, including office tasks, though without coordinated multi-file deliverables (Huang et al., 2026; Wu et al., 2026a), while FinEvo-Bench measures learning across professional financial workflows (Deng et al., 2026). These evaluations address both initial capability and improvement from experience. However, none of these suites measures whether changes learned on one file type transfer to workflows whose deliverables must agree, and none scores office artifacts at the requirement level needed to see that difference. Our question concerns a specific form of transfer: do changes learned from feedback on individual file types help on new workflows whose deliverables must agree? Reporting these two conditions separately matters because an overall evolution gain can hide improvement on one and regression on the other.

Several design choices are needed to make this comparison informative. Feedback tasks must be separated from final tests, because fixing the tasks that prompted a revision does not establish that it will help elsewhere. A shared initial harness and model make gains comparable, while explicit budgets help distinguish the effect of an update from that of additional computation (Wang et al., 2026; Zheng et al., 2026). Scores must check the requested work in the files themselves: a report can disagree with its workbook, and a spreadsheet can contain the right number without the required formula. Evaluation therefore needs checks of content, native structure, and consistency, with visual assessment where layout matters. Rubric judgments also need their own reliability checks (Peng et al., 2026). Reserving cross-office workflows for final testing then asks whether procedures learned on individual file types remain useful when information must pass between files.

EvoCowork implements these requirements in a suite of 240 tasks adapted from public sources, covering presentations, documents, spreadsheets, and cross-office workflows. Each task has fixed input files and explicit output requirements. Automated checks and, where applicable, visual rubrics score the submitted artifacts and provide feedback. Evolution methods learn from single-modality feedback tasks; selection tasks choose the revision, and the two final partitions measure generalization to new single-modality tasks and transfer to cross-office workflows; the selected harness is frozen before final testing. Each method starts from the same harness and model. Search and inference costs are recorded alongside changes in task scores.

Our evaluation compares ACE (Zhang et al., 2026b) and AHE (Lin et al., 2026) with the frozen initial harness, H_0. The static results in Table 3 describe the starting capabilities of the evaluated agents. The paired comparison in Table 4 tests changes from H_0 on unseen tasks. ACE gains 2.46 points overall and AHE 0.46, but both split along the two test sets: in-domain scores rise by 4.90 and 2.64 points while cross-office scores change by $+0.03$ and -1.72. Selection scores point the other way for AHE, which loses 7.03 points there and still gains overall on the final tasks.

Overall, our contributions are threefold: (1) an artifact-aware Office benchmark of 240 tasks whose checks score content, native editability, and cross-file consistency in the submitted files; (2) a controlled RSI protocol that separates feedback, selection, and final testing, and measures gains on unseen tasks and transfer to cross-office workflows under a shared starting harness and explicit budgets; and (3) an empirical finding that gains on isolated file types do not automatically transfer to cross-office coordination, which reporting the two conditions separately keeps visible.

2 Related Work

2.1 Office Tasks and Artifact Evaluation

Office-agent benchmarks cover both interactive workflows and work-product quality. OSWorld evaluates real desktop interaction (Xie et al., 2024), while OfficeBench emphasizes planning across office applications (Wang et al., 2024). APEX-Agents tests professional work across applications (Vidgen et al., 2026), and GDPval uses expert comparisons of occupational deliverables (Patwardhan et al., 2025). Native office files also require distinct forms of evaluation. Office Comprehension Benchmark tests understanding of their content and structure (Shaik et al., 2026); PPTArena and PPT-Eval test PowerPoint editing and creation (Ofengenden et al., 2025; Gandhi et al., 2026), and SpreadsheetBench 2 covers business spreadsheet workflows (Zhu et al., 2026). OmegaUse-OfficeVal evaluates office-suite deliverables with economic grounding (Zhou et al., 2026), while Workspace-Bench tests dependencies among heterogeneous files (Tang et al., 2026). These tasks motivate evaluating the work product that a task requires, with scoring tied to its intended use (Hua et al., 2026). EvoCowork uses native office deliverables to test whether harness improvements transfer from individual file types to coordinated workflows.

2.2 Controlled Evaluation of Agent Harnesses

ACE updates a context playbook (Zhang et al., 2026b), while AHE and Meta-Harness edit harness code using execution evidence (Lin et al., 2026; Lee et al., 2026). Self-Harness checks revisions for regressions, while DarwinX searches a population of harnesses with the model fixed (Zhang et al., 2026a;c). Evaluations must separate revision gains from extra computation (Wang et al., 2026). HarnessOpt-Bench, Evo-Bench, and HarnessDev evaluate revisions on held-out tasks (Ursekar et al., 2026; Huang et al., 2026; Wu et al., 2026a); PAST-Bench and ContinualSkillBench study experience reuse (Xue et al., 2026; Guan et al., 2026). S3Gym separates self-testing, self-judging, and self-improvement (Shi et al., 2026), while Aspire studies evolution from vague goals under hidden evaluation (Wu et al., 2026b). SEAGym compares methods across training, validation, ID/OOD tests, and cost (Zheng et al., 2026), while FinEvo-Bench tracks learning through sequential financial tasks (Deng et al., 2026). EvoCowork tests transfer from single-modality feedback to cross-office work: ACE and AHE share a model and initial harness, and each selected revision is frozen before testing native file structure and agreement across deliverables.

3 The EvoCowork Benchmark

3.1 Design Rationale

Office tasks require both correct information and usable files. A report can state the right numbers while omitting required editable tables; a workbook and its summary can each look plausible while reporting different totals. We therefore evaluate the submitted artifacts against explicit content, structure, and consistency requirements. Visual checks supplement these requirements on tasks where layout matters. This connects the work being represented, the required product, and the result actually scored, as advocated in work-centered benchmark design (Hua et al., 2026).

The four modalities separate work within a file type from coordination across types. Presentation, document, and spreadsheet tasks test their respective operations, while cross-office tasks require agents to carry information through dependent deliverables. With 60 tasks per modality, each contributes equally to the static overall score. This balanced allocation defines the scope of the release. Workflow categories describe coverage, whereas difficulty labels summarize the performance of a fixed calibration panel. Their generalization must be tested with a held-out model.

Table 1: Per-modality scale and artifact counts. Families and sources are deduplicated.

Modality	Fam.	Src.	In	Out
Presentation	12	6	105	67
Document	17	5	107	62
Spreadsheet	22	9	190	60
Cross-office	9	4	223	343
Total	**59**	**11**	**625**	**532**

Public source tasks provide requests and inputs. Freezing those inputs and specifying the output contract lets different harnesses face the same task and supports replay after a harness changes.

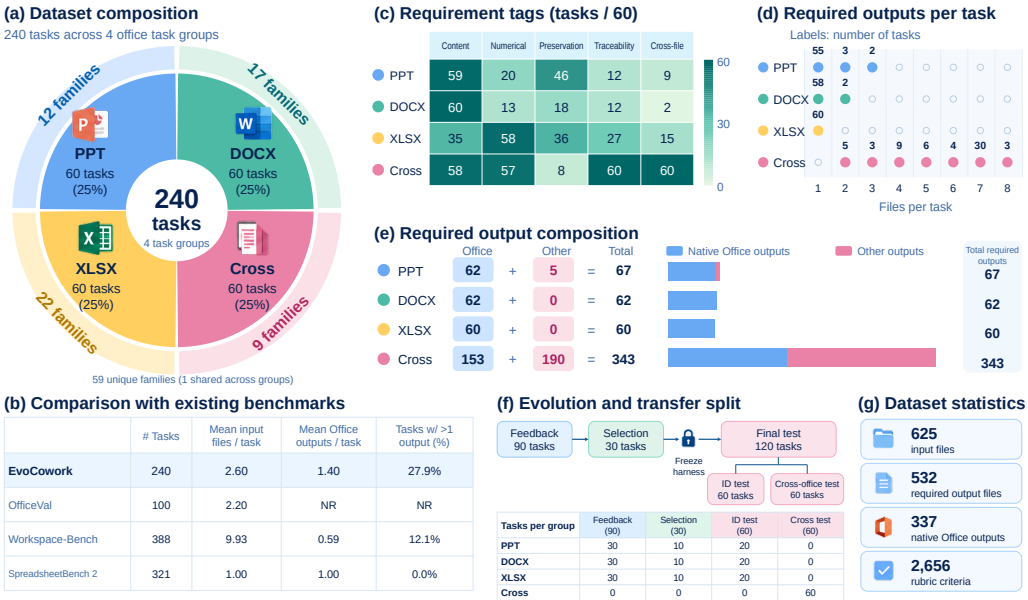


Figure 2: Task composition, artifact requirements, and evolution splits in EvoCowork. Panels (a–b) show task coverage and released-task metadata across benchmarks; (c–e) show requirement tags and output demands; (f–g) show the evaluation split and dataset totals. Requirement tags overlap. NR denotes an unavailable comparable field.

Static evaluation measures initial task performance. Evolution evaluation asks whether changes learned from feedback improve performance on separate final tasks. We therefore separate feedback, revision selection, and final evaluation, and reserve cross-office tasks to test transfer from single-modality feedback. Separate update-validation and transfer views are also used by SEAGym (Zheng et al., 2026); here the transfer changes the coordination required among office deliverables (Appendix C).

3.2 Task Scope and Taxonomy

EvoCowork scores the files an agent submits. Tasks ask agents to create, edit, analyze, repair, or audit office files. Task families describe operations such as reconstruction, formula writing, visual redesign, and reconciliation across files. Within each modality, we also group tasks by workflow structure. Presentation tasks range from editing a slide to producing a complete deck from several sources. Document tasks cover content editing, native-object reconstruction, evidence synthesis, and controlled review. Spreadsheet tasks cover local transformations, workbook computation, source reconciliation, and decision workflows. Cross-office tasks range from handing off one file to coordinating several dependent outputs; their 343 required outputs across 60 tasks reflect the coordination demands that make them harder than single-modality work (Table 1). These categories describe coverage; empirical difficulty is assigned separately. Appendix A gives the definitions and examples.

Difficulty is calibrated by one common protocol. Every modality is evaluated by the same frozen panel $\mathcal{A}$ of model–harness configurations under matched budgets and a common seed policy. For task i with visual scoring enabled, harness a, and seed s, the task reward and panel summary are

$$r_{i,a,s} = 0.7r^{\text{det}}_{i,a,s} + 0.3r^{\text{vis}}_{i,a,s}, \qquad \bar{r}_i = \frac{1}{|\mathcal{A}|S}\sum_{a\in\mathcal{A}}\sum_{s=1}^{S} r_{i,a,s}, \qquad d_i = 1 - \bar{r}_i, \tag{1}$$

where r^{det} is the deterministic artifact score, r^{vis} is the task-specific visual score, and d_i is empirical difficulty. Tasks without visual scoring use $r_{i,a,s} = r^{\text{det}}_{i,a,s}$. Tasks are ranked by $\bar{r}_i$ within each modality and assigned to precommitted 20/30/30/20 bins for *Easy*/*Medium*/*Hard*/*Expert*; a boundary tie triggers an audit against that modality's structural regime before stable task-ID tie-breaking.

Table 2: Office-agent benchmark capabilities. ✓/●/✗ indicate full, partial, and no support; P/D/S/X denote presentation, document, spreadsheet, and cross-office tasks.

Benchmark	Tasks	Scope	Real source grounding	Editable artifacts	Isolated replay	Atomic reward	Three-way split	Harness mutation
PPT-Eval (Gandhi et al., 2026)	120	P	●	✓	●	✓	✗	✗
OmegaUse-OfficeVal (Zhou et al., 2026)	100	P/D/S	✓	✓	●	✓	✗	✗
OfficeBench (Wang et al., 2024)	300	Multi-app	●	●	✓	●	✗	✗
OSWorld (Xie et al., 2024)	369	Desktop	✓	●	✓	●	✗	✗
SpreadsheetBench (Ma et al., 2024)	912	S	✓	✓	●	●	✗	✗
TheAgentCompany (Xu et al., 2024)	175	Workplace	●	●	✓	✓	✗	✗
STATE-Bench (Microsoft, 2026)	450	Enterprise	✗	✗	✓	●	●	●
HarnessOpt-Bench (Ursekar et al., 2026)	4 targets	General	●	●	✓	●	✓	✓
Evo-Bench (Huang et al., 2026)	608	S/O/G	●	●	✓	●	●	✓
EvoCowork (ours)	**240**	**P/D/S/X**	✓	✓	✓	✓	✓	✓

Table 2 compares evaluation objects, artifact-related capabilities, and evolution protocols. Raw task counts use each source's released evaluation unit. EvoCowork focuses on source-grounded, editable Office deliverables with isolated replay and task-specific artifact rewards.

3.3 Source-Grounded Task Construction

We select tasks from public, license-compatible sources and adapt them individually. The presentation track draws from PPTArena (Ofengenden et al., 2025), OmegaUse-OfficeVal (Zhou et al., 2026), AutomationBench, TheAgentCompany (Xu et al., 2024), Cowork-Bench, and OfficeBench (Wang et al., 2024); the cross-office track draws from AutomationBench, Cowork-Bench, OfficeBench, and OmegaUse-OfficeVal. Replaying an upstream instance requires more than copying its prompt. Some requests leave the deliverable implicit or depend on mutable application and network state. A broad end-state check may miss an incorrect formula, an uneditable object, or a contradiction across deliverables. Each adaptation therefore freezes the required inputs, maps the request into an explicit deliverable contract, records checksums and the exact upstream revision, and rebuilds or strengthens the evaluator where necessary. Shared sources, templates, and near-duplicate workflows receive leakage-group identifiers for overlap audits and controlled partitioning. The presentation, document, spreadsheet, and cross-office source inventories and upstream task identifiers are reported in Appendix B, together with their frozen revisions and licenses.

To verify task quality, we sampled 24 tasks per modality, stratified by difficulty tier and upstream source, and each task receives two independent reviews from a three-person panel, with the third member adjudicating disagreements (Appendix E). Reviewers read the task instruction, source inputs, reference answer, and rubric, and assess whether the instruction is unambiguous, the reference answer factually correct, and every stated requirement satisfied. Tasks flagged for ambiguity or incorrect reference answers were revised before inclusion.

3.4 Replayable Task Packages and Artifact-Aware Judges

Each presentation or cross-office task runs as a self-contained package. The agent receives the instruction, fixed input files, execution environment, and output requirements. Task-specific deterministic judges inspect file validity, editable structures, factual content, formulas, cross-artifact consistency, provenance, and preservation constraints as applicable. Tasks with visually judged PPTX or PDF deliverables additionally use a task-specific rubric for hierarchy, spacing, balance, visual fidelity, and professional quality with a fixed Codex–VLM evaluator. Exact-recreation tasks expose the reference rendering to this evaluator; open-ended generation tasks do not. The two scores are retained separately and combined using the 70/30 rule in Equation 1. Reference and negative-output replays audit the deterministic component and the visual-evaluation contract. Document and spreadsheet tasks follow the same packaging protocol and are scored exclusively by task-specific deterministic judges. Replays of the reference solutions check execution, while model–harness evaluations measure agent performance (Table 3).

Using criteria specific to each task is also studied in agent-trajectory evaluation (Ding, 2026). Their specificity alone does not establish judge reliability: rubric verifiers can still misread complex agent

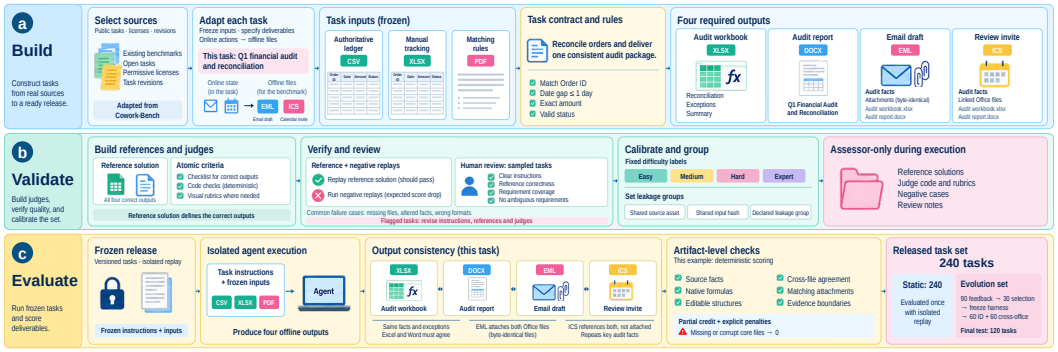


Figure 3: A cross-office audit adapted from Cowork-Bench illustrates (a) task construction, (b) validation of the reference solution and rubric, and (c) artifact evaluation for correctness, editability, and cross-file consistency across the four deliverables.

outputs (Peng et al., 2026). We therefore distinguish implementation checks from human agreement in Section 4.4 and Appendix D, where both are reported per modality.

3.5 Replay Controls and Data Access

Static evaluation keeps the model–harness configuration, task inputs, judges, attempt policy, and resource limits fixed for the corresponding comparison. The agent receives the instruction and allowed inputs, while reference answers and verifier internals remain outside its writable workspace. Saved outputs, execution traces, and task-level scores support later auditing. The primary static diagnostic covers all 240 tasks. Evolution uses separate feedback, selection, and final partitions (Appendix C). Harness changes are made during feedback and selected on validation tasks; final evaluation freezes the selected revision. Static and final evolution scores are reported on their respective task sets.

4 Results and Analysis

4.1 Experimental Setup

Static evaluation.(a) **Benchmark and baselines.** The static evaluation covers all 240 tasks. Table 3 compares Sonnet 5 and GLM 5.3 with Claude Code, OpenCode, OpenHands, and mini-SWE-agent (MSA). GLM 5.3 is excluded from difficulty calibration and uses the fixed Sonnet 5 labels. Versions, score reuse, and reporting details appear in Appendix I.1. (b) **Metrics.** Scores are 100 times mean task reward: each modality averages its 60 tasks, and the overall score averages the four modality means. Valid agent failures receive the task judge's score; unresolved infrastructure failures are recorded separately in the run records.

Evolution evaluation.(a) **Baselines.** The comparison evaluates ACE (Zhang et al., 2026b) and AHE (Lin et al., 2026) against a frozen MSA harness H_0, with GLM 5.3 as the solver and updater. ACE updates a context playbook; AHE revises harness components using execution evidence. Both start from the same harness. (b) **Protocol.** The protocol separates feedback, revision selection, and final testing. Each selected harness is frozen before evaluation on held-out single-modality and cross-office tasks, using the same task versions, inference budgets, and judges as H_0. Appendix C gives the split and access rules; Appendix I.2 describes the shared budget controls and configuration requirements.

4.2 Static Model–Harness Comparison

Table 3 reports the completed model–harness results. With Sonnet 5, MSA scores 38.08 overall, OpenHands 36.73, Claude Code 33.85, and OpenCode 32.47. Within that panel, each harness scores lower on cross-office tasks than on any single modality. These are mean rewards, not percentages of tasks solved; spreadsheet judges can also assign negative penalties. The 240-task Claude Code run

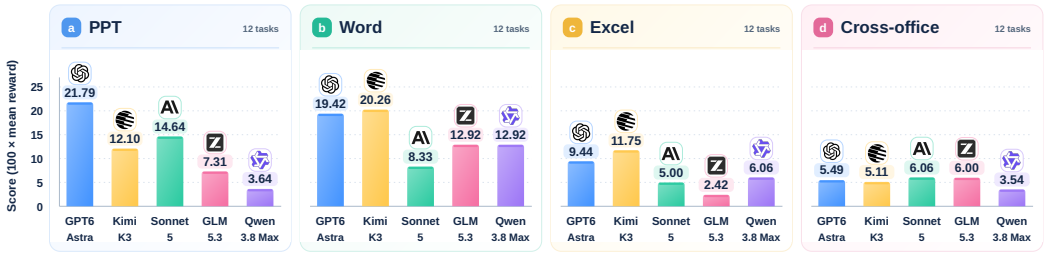


Figure 4: Expert-tier performance with a frozen MSA harness. Each panel reports 100 × mean reward over the same 12 tasks for all five models.

Table 3: Static performance by modality and difficulty (100 × mean reward). All summarizes 60 tasks per modality; tiers follow the calibration described in Appendix A. Overall averages the four modality means at each difficulty level. MSA denotes mini-SWE-agent. Each reported configuration covers all 240 tasks.

Harness	Modality	Sonnet 5					GLM 5.3				
		All	Easy	Medium	Hard	Expert	**All**	Easy	Medium	Hard	Expert
Claude Code	PPT	**37.29**	96.96	35.51	18.02	9.19	**49.16**	99.29	64.97	25.77	10.38
	Word	**36.59**	90.59	50.21	11.37	0.00	**42.48**	80.46	62.75	19.00	9.33
	Excel	**38.38**	79.45	51.33	23.58	0.09	**46.76**	81.08	60.75	29.81	16.89
	Cross-office	**23.12**	72.20	18.61	10.32	0.02	**30.62**	72.58	27.09	22.83	5.63
	Overall	**33.85**	**84.80**	**38.91**	**15.82**	**2.32**	**42.25**	**83.35**	**53.89**	**24.35**	**10.56**
OpenCode	PPT	**36.87**	97.29	37.27	16.53	6.37	**40.90**	93.38	59.02	12.09	4.47
	Word	**38.62**	84.13	59.52	13.14	0.00	**36.61**	73.69	59.00	8.36	8.33
	Excel	**34.03**	69.25	36.65	26.31	6.47	**15.76**	33.12	21.78	11.49	-4.25
	Cross-office	**20.34**	71.91	15.93	3.93	0.00	**6.88**	21.66	4.35	4.13	0.00
	Overall	**32.47**	**80.64**	**37.34**	**14.98**	**3.21**	**25.04**	**55.47**	**36.04**	**9.02**	**2.14**
OpenHands	PPT	**38.07**	97.98	40.61	16.23	7.11	**20.71**	68.72	12.70	7.79	4.11
	Word	**40.60**	85.99	66.77	11.22	0.00	**37.30**	68.77	53.94	10.39	21.25
	Excel	**44.98**	83.29	61.61	27.44	8.00	**12.16**	42.65	10.47	7.65	-9.06
	Cross-office	**23.26**	65.51	24.69	8.99	0.26	**18.66**	48.18	16.48	10.84	4.12
	Overall	**36.73**	**83.20**	**48.42**	**15.97**	**3.84**	**22.21**	**57.08**	**23.40**	**9.17**	**5.11**
MSA	PPT	**47.92**	98.19	62.93	21.59	14.64	**47.58**	99.07	75.94	11.73	7.31
	Word	**38.13**	84.10	53.72	11.75	8.33	**41.06**	78.08	60.46	15.73	12.92
	Excel	**38.30**	64.57	59.88	21.39	5.00	**40.29**	73.42	59.91	23.81	2.42
	Cross-office	**27.99**	74.75	24.62	14.80	6.06	**28.68**	69.29	26.70	18.69	6.00
	Overall	**38.08**	**80.40**	**50.29**	**17.38**	**8.51**	**39.40**	**79.97**	**55.76**	**17.49**	**7.16**

with GLM 5.3 scores 42.25 overall and 30.62 on cross-office tasks. MSA scores 39.40 overall and 28.68 on cross-office tasks; OpenCode and OpenHands score 25.04 and 22.21 overall, respectively. All GLM 5.3 results use the fixed difficulty labels.

Figure 4 compares five models on the 48 Expert tasks with a frozen MSA harness. GPT6 Astra leads in presentations and Kimi K3 in documents and spreadsheets, while all five models score 3.54–6.06 on cross-office tasks under the same protocol.

4.3 Evolution and Cross-Office Transfer

Table 4 compares the frozen MSA harness H_0 with the ACE and AHE revisions on 120 final tasks: 60 in-domain tasks (20 each for presentations, documents, and spreadsheets) and 60 cross-office tasks. All three use GLM 5.3 with matched per-task limits and scoring rules. ACE scores 39.72 overall (+2.46 over H_0) and AHE 37.72 (+0.46), with in-domain gains of 4.90 and 2.64, respectively. Cross-office scores are 28.70 for ACE (+0.03) and 26.96 for AHE (-1.72). AHE crashes in 31/120

Table 4: Evolution results (GLM 5.3, MSA seed). Scores are 100× mean reward; Δ is the change from H_0 on the same tasks. Selection and final sets are disjoint. AHE reports the evolved revision, without fallback to H_0.

	Selection (30)		In-domain (60)		Cross-office (60)		All (120)	
Method	Score	Δ	Score	Δ	Score	Δ	Score	Δ
Frozen H_0	47.44	0.00	45.83	0.00	28.68	0.00	37.25	0.00
ACE (Zhang et al., 2026b)	54.98	**+7.55**	50.73	**+4.90**	28.70	**+0.03**	39.72	**+2.46**
AHE (Lin et al., 2026)	40.41	**−7.03**	48.47	**+2.64**	26.96	**−1.72**	37.72	**+0.46**

final attempts, including 17 cross-office cases, in its added completion check; the crash-excluded diagnostic appears in Table A13. Both methods therefore raise overall performance while cross-office scores remain nearly unchanged or decline, which the separate test sets make visible.

Selection scores also differ from final gains: ACE improves by 7.55 points on the 30 selection tasks, whereas AHE loses 7.03 yet improves slightly on final tasks. These outcomes support separate selection and final tests and direct measurement of cross-office transfer. We report AHE's evolved revision even though its original selection rule retained H_0; the reporting rule is disclosed in Appendix I.2. Each method contributes one fixed-model run.

To calibrate run-to-run variation, we rerun frozen H_0 on the 60 cross-office final tasks with a different sampling seed. The rerun scores 28.31, 0.37 points below 28.68, our noise scale for one run. ACE's $+0.03$ lies within this noise; AHE's -1.72 is roughly five times larger.

4.4 Judge Reliability and Human Audit

Table 5 reports checks on the full 240-task release alongside the human audit. All reference solutions receive a deterministic reward of 1. Empty and corrupt outputs receive exactly 0 on all presentation and cross-office tasks, but some document and spreadsheet judges retain positive credit without a valid deliverable. Spreadsheet penalties also produce negative rewards; these count as deviations from the expected zero score in this table. Appendix D separates these outcomes and describes the checks and human audit protocol.

The human audit rates 60 outputs per modality with three raters each. Ordinal Krippendorff's α ranges from 0.969 to 0.994, so the panels apply the rubrics consistently and the adjudicated consensus rests on a stable target. Score MAE against that consensus is 3.779 points on spreadsheets, 7.393 on cross-office workflows, 8.501 on documents, and 9.576 on presentations, on the 100-times-reward scale. The ordering runs opposite to the implementation checks: spreadsheet judges retain the most credit for invalid outputs yet track human scores most closely, while presentation judges pass every deterministic check and still differ most from human readers. Deterministic checks and human agreement therefore measure different things, and neither substitutes for the other.

4.5 Failure Types

Table 6 separates static configurations from evolved harnesses on the same final tasks. Delivery columns count omitted required files; quality columns classify judge-recorded requirements into seven rule-based categories, with an agent-assisted pass where no rule matches. Human reviewers confirmed the assignments on 160 stratified cases (Appendix F.1), with no errors. Missing-file checks are not counted as additional violations; coverage and denominators are in Appendix F.

Incorrect facts and calculations are the largest category in all eleven configurations, covering 0.154 to 0.558 of tasks and exceeding the delivery-failure rate in nine of them. Invalid native structure is second in nine configurations, at 0.050 to 0.221; for H_0 and ACE, unrequested scope or privacy changes take second place. Unrequested changes to protected content and cross-file inconsistency stay at or below 0.058 everywhere. The evolution panel separates delivery from content: ACE and AHE both deliver more files than H_0, lowering the delivery-failure rate from 0.267 to 0.217 and 0.183, while violations rise in the content categories — incorrect facts and calculations go from 0.508 to 0.542 for ACE and 0.558 for AHE, invalid native structure from 0.108 to 0.125 and 0.158,

Table 5: Deterministic judge checks and human audit. Implementation checks report the fraction of the 60 tasks that receive the expected reward; values below 1.00 cover tasks scoring above or below it.

		Implementation checks			Human audit			
Modality	Tasks	Ref. replay ($r = 1$)	Empty out ($r = 0$)	Corrupt out ($r = 0$)	Outputs	Raters	IAA α	Score MAE
Presentation	60	1.00	1.00	1.00	60	3	0.969	9.576
Document	60	1.00	0.88	0.88	60	3	0.994	8.501
Spreadsheet	60	1.00	0.45	0.43	60	3	0.981	3.779
Cross-office	60	1.00	1.00	1.00	60	3	0.983	7.393

Table 6: Delivery failures and artifact-quality violations. The upper panel covers static configurations; the lower panel compares evolution methods on the same 120 final tasks. Entries are proportions on a 0–1 scale, rounded to three decimal places. Quality columns count rule-identified violations over 240 tasks per static configuration or 120 tasks per evolution configuration. Human reviewers confirmed the category assignments on a stratified sample (Appendix F.1). A task can fall in several categories, so the columns do not sum.

		Delivery failures ↓					Artifact-quality violations ↓						
Model	**Harness**	**PPT**	**Word**	**Excel**	**Cross-office**	**All**	**Facts / calcs**	**Native structure**	**Protected content**	**Evidence trace**	**Scope / privacy**	**Cross-file consist.**	**Visual layout**
STATIC EVALUATION	Frozen model–harness configurations · 240 tasks each												
Sonnet 5	Claude Code	0.000	0.250	0.117	0.100	**0.117**	0.517	0.204	0.042	0.037	0.075	0.021	0.154
	OpenCode	0.017	0.217	0.183	0.267	**0.171**	0.442	0.200	0.058	0.029	0.058	0.017	0.142
	OpenHands	0.000	0.233	0.067	0.117	**0.104**	0.483	0.221	0.054	0.029	0.071	0.021	0.163
	MSA	0.200	0.350	0.217	0.200	**0.242**	0.492	0.129	0.021	0.063	0.083	0.013	0.075
GLM 5.3	Claude Code	0.150	0.167	0.100	0.050	**0.117**	0.558	0.154	0.029	0.079	0.108	0.025	0.096
	OpenCode	0.450	0.424	0.467	0.867	**0.552**	0.154	0.050	0.004	0.008	0.008	0.000	0.033
	OpenHands	0.650	0.417	0.667	0.517	**0.563**	0.233	0.071	0.004	0.042	0.063	0.008	0.033
	MSA	0.283	0.217	0.233	0.267	**0.250**	0.442	0.121	0.017	0.054	0.087	0.017	0.054
EVOLUTION EVALUATION	Shared MSA seed · Same 120 final tasks												
GLM 5.3	MSA (H_0)	0.250	0.300	0.250	0.267	**0.267**	0.508	0.108	0.008	0.092	0.158	0.033	0.058
	MSA + ACE	0.150	0.250	0.200	0.233	**0.217**	0.542	0.125	0.042	0.092	0.150	0.033	0.042
	MSA + AHE	0.050	0.000	0.200	0.283	**0.183**	0.558	0.158	0.033	0.083	0.133	0.025	0.100

and protected-content changes from 0.008 to 0.042 and 0.033 — partly because more delivered files bring more checks into scope, while scope and cross-file violations stay flat or fall.

4.6 Difficulty Generalization to a Held-Out Model

Across four GLM 5.3 harnesses and 240 tasks, tier means remain non-increasing under fixed Sonnet 5 labels, except at the document Hard–Expert boundary: all 12 Expert tasks have zero calibration scores and require tie-breaking. We keep these labels fixed for GLM 5.3 to test transfer across models without rebuilding the tiers. The exception matches its cause: the document Expert tier was cut from Hard by tie-breaking within a block of zero calibration scores, and under GLM 5.3 one harness scores higher on Expert than on Hard documents, while every boundary backed by measured scores stays ordered. Expert tasks also remain hardest for GLM 5.3 itself, at 2.14–10.56 across its four harnesses, so the tiers track difficulty for both models rather than the calibration panel alone.

AI USE STATEMENT

AI tools assisted with manuscript proofreading, language editing, reference verification, and dataset construction, including task synthesis and adaptation. The authors are responsible for verifying the resulting text, citations, dataset content, and reported experimental results.

ETHICS STATEMENT

Task packages should preserve upstream attribution and license obligations. Adapted business scenarios and automated scores are research evaluation material, not authorization to act on real accounts or a substitute for professional review. Any release of artifacts or traces must exclude credentials and review source materials for personal or otherwise restricted information.

REFERENCES

Bo Deng, Kang Zhou, Lifan Guo, Chongyang Tao, Xuanren Chen, Chenggang Xie, Renzhao Liang, Feng Chen, and Chi Zhang. FinEvo-Bench: A Longitudinal Benchmark for Self-Evolving Agents in Professional Financial Workflows, 2026. URL https://arxiv.org/abs/2608.06144.

Liang Ding. AdaRubric: Task-Adaptive Rubrics for Reliable LLM Agent Evaluation and Reward Learning, 2026. URL https://arxiv.org/abs/2603.21362.

Apurva Gandhi, Vishwas Suryanarayanan, Raja Hasnain Anwar, Firoz Shaik, Shubhang Desai, Thong Q. Nguyen, Muhammad Taqi Raza, Vishal Chowdhary, and Graham Neubig. PPT-Eval: A benchmark for computer-use agents on PowerPoint tasks. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://microsoft.github.io/ppteval/.

Tianyi Guan, Yiding Wang, Haotong Yang, Siyuan Cao, Shirui Liu, Yi Hu, Jiaqi Li, and Muhan Zhang. ContinualSkillBench: Can LLM Agents Truly Evolve Their Capabilities?, 2026. URL https://arxiv.org/abs/2608.03874.

Yining Hua, Hongbin Na, Cyrus Ayubcha, and Levi Lian. Designing Benchmarks for Knowledge Work, 2026. URL https://arxiv.org/abs/2605.23262.

Lisheng Huang, Chen Yang, Hao Zhou, Huatong Song, Zongchao Chen, Ran Le, Yang Song, Wayne Xin Zhao, and Tao Zhang. Evo-Bench: Can language models improve agent harness? arXiv:2608.09096, 2026. URL https://arxiv.org/abs/2608.09096. Preprint.

Yoonho Lee, Roshen Nair, Qizheng Zhang, Kangwook Lee, Omar Khattab, and Chelsea Finn. Meta-Harness: End-to-End Optimization of Model Harnesses, 2026. URL https://arxiv.org/abs/2603.28052.

Jiahang Lin, Shichun Liu, Chengjun Pan, Lizhi Lin, Shihan Dou, Zhiheng Xi, Xuanjing Huang, Hang Yan, Zhenhua Han, Tao Gui, and Yu-Gang Jiang. Agentic harness engineering: Observability-driven automatic evolution of coding-agent harnesses. arXiv:2604.25850, 2026. URL https://arxiv.org/abs/2604.25850. Preprint.

Zeyao Ma, Bohan Zhang, Jing Zhang, Jifan Yu, Xiaokang Zhang, Xiaohan Zhang, Sijia Luo, Xi Wang, and Jie Tang. SpreadsheetBench: Towards challenging real world spreadsheet manipulation. In *Advances in Neural Information Processing Systems*, 2024. doi: 10.52202/079017-3007.

Microsoft. STATE-Bench: Benchmark AI agents on enterprise workflows. GitHub repository, 2026. URL https://github.com/microsoft/STATE-Bench. Accessed September 20, 2026.

Michael Ofengenden, Yunze Man, Ziqi Pang, Liang-Yan Gui, and Yu-Xiong Wang. PPTArena: A Benchmark for PowerPoint Editing, 2025. URL https://arxiv.org/abs/2512.03042.

Tejal Patwardhan, Rachel Dias, Elizabeth Proehl, Grace Kim, Michele Wang, Olivia Watkins, Simón Posada Fishman, Marwan Aljubeh, Phoebe Thacker, Laurance Fauconnet, Natalie S. Kim, Patrick Chao, Samuel Miserendino, Gildas Chabot, David Li, Michael Sharman, Alexandra Barr, Amelia Glaese, and Jerry Tworek. GDPval: Evaluating AI Model Performance on Real-World Economically Valuable Tasks, 2025. URL https://arxiv.org/abs/2510.04374.

Yangda Peng, Yunjia Qi, Haotian Xia, Guanzhong He, Xintong Shi, Richeng Xuan, Songyuanyi Lu, Yixian Liu, Zhichao Hu, Yuhong Liu, and Hao Peng. Can LLM-as-a-Judge Reliably Verify Rubrics in Agentic Scenarios?, 2026. URL https://arxiv.org/abs/2606.29920.

Firoz Shaik, Mateus Picanço Lima Gomes, Tanvir Aumi, Jingci Wang, Milos Milunovic, Filip Basara, Ivana Jovanovic, Vishwas Suryanarayanan, Neha Nandan Kenkare, Weiyao Xie, Zhipeng Han, Zheng Zhang, Waleed Shahid, Jay Rathi, Russell Scherer, Thong Q. Nguyen, Michael Bentley, Tamara Stankovic, Rasika Chakravarthy, and Vishal Chowdhary. Office Comprehension Benchmark, 2026. URL https://arxiv.org/abs/2607.01245.

Jiajun Shi, Siyuan Tao, Yuhao Wu, Zexuan Wang, Jingyuan Zhang, Jiaheng Liu, Xinping Lei, Xinrong Zhang, Siyuan Fang, Zhewen Tan, Tianle Cai, Junhao Fang, Jiameng Huang, Yueyang Wang, Jinkai Liu, Yuxuan Zhang, Jian Yang, Zhoujun Li, Shen Yan, Wenhao Huang, and Ge Zhang. S3Gym: Can LLMs Turn Self-Testing and Self-Judging into Self-Improvement?, 2026. URL https://arxiv.org/abs/2608.31100.

Zirui Tang, Xuanhe Zhou, Yumou Liu, Linchun Li, Yukai Wu, Weizheng Wang, Hongzhang Huang, Wei Zhou, Jun Zhou, Jiachen Song, Shaoli Yu, Jinqi Wang, Zihang Zhou, Hongyi Zhou, Yuting Lv, Jinyang Li, Jiashuo Liu, Ruoyu Chen, Chunwei Liu, GuoLiang Li, Jihua Kang, and Fan Wu. Workspace-Bench 1.0: Benchmarking AI Agents on Workspace Tasks with Large-Scale File Dependencies, 2026. URL https://arxiv.org/abs/2605.03596.

Varun Ursekar, Apaar Shanker, Yash Maurya, Shehab Yasser, Vijay S. Kalmath, Veronica Chatrath, and Yuan Xue. HarnessOpt-Bench: Evaluating LLMs at harness optimization. arXiv:2608.06301, 2026. URL https://arxiv.org/abs/2608.06301. Preprint.

Bertie Vidgen, Austin Mann, Abby Fennelly, John Wright Stanly, Lucas Rothman, Marco Burstein, Julien Benchek, David Ostrofsky, Anirudh Ravichandran, Debnil Sur, Neel Venugopal, Alannah Hsia, Isaac Robinson, Calix Huang, Olivia Varones, Daniyal Khan, Michael Haines, Austin Bridges, Jesse Boyle, Koby Twist, Zach Richards, Chirag Mahapatra, Brendan Foody, and Osvald Nitski. APEX-Agents, 2026. URL https://arxiv.org/abs/2601.14242.

Yike Wang, Huaisheng Zhu, Zhengyu Hu, Yige Yuan, Zhengyu Chen, Shakti Senthil, Hannaneh Hajishirzi, Yulia Tsvetkov, Pradeep Dasigi, and Teng Xiao. Rethinking the evaluation of harness evolution for agents. arXiv:2607.12227, 2026. URL https://arxiv.org/abs/2607.12227. Preprint.

Zilong Wang, Yuedong Cui, Li Zhong, Zimin Zhang, Da Yin, Bill Yuchen Lin, and Jingbo Shang. OfficeBench: Benchmarking language agents across multiple applications for office automation. *arXiv preprint arXiv:2407.19056*, 2024.

Yuhao Wu, Jingyuan Zhang, Jiajun Shi, Xinping Lei, Qingshui Gu, Yuxuan Zhang, Zexuan Wang, Chen He, Chen Huang, Maojia Song, Zhiyuan Zeng, Shaowen Wang, Jinkai Liu, Yunfeng Shi, Jiaheng Liu, Shen Yan, Wenhao Huang, Ge Zhang, and Wenxuan Zhang. HarnessDev: Can LLMs Create and Evolve Their Own Agent Harness?, 2026a. URL https://arxiv.org/abs/2609.01437.

Yuhao Wu, Jingyuan Zhang, Jiajun Shi, Yuxuan Zhang, Xinping Lei, Junting Zhou, Zexuan Wang, Yuchen Wu, Huan Zhou, Duo Wang, Yinzhu Piao, Yongchang Peng, Yunfeng Shi, Jin Chen, Zuo Wang, Jinkai Liu, Jiaheng Liu, Wenxuan Zhang, Shen Yan, Wenhao Huang, and Ge Zhang. Aspire: Can Models Self-Evolve from Vague Goals?, 2026b. URL https://arxiv.org/abs/2608.31111.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. OSWorld: Benchmarking multimodal

agents for open-ended tasks in real computer environments. *arXiv preprint arXiv:2404.07972*, 2024.

Frank F. Xu, Yufan Song, Boxuan Li, Yuxuan Tang, Kritanjali Jain, Mengxue Bao, Zora Z. Wang, Xuhui Zhou, Zhitong Guo, Murong Cao, Mingyang Yang, Hao Yang Lu, Amaad Martin, Zhe Su, Leander Maben, Raj Mehta, Wayne Chi, Lawrence Jang, Yiqing Xie, Shuyan Zhou, and Graham Neubig. TheAgentCompany: Benchmarking LLM agents on consequential real world tasks. arXiv:2412.14161, 2024. URL https://arxiv.org/abs/2412.14161. Preprint.

Shuhan Xue, Zixin Ding, Yichen Shen, Yinjie Wang, Zhenfei Yin, Yingcheng Wu, Yuxin Chen, Mengdi Wang, and Ling Yang. PAST-Bench: Benchmarking the Foundations of Recursive Self-Improvement in Personal Agents, 2026. URL https://arxiv.org/abs/2608.04003.

Hangfan Zhang, Shao Zhang, Kangcong Li, Chen Zhang, Yang Chen, Yiqun Zhang, Lei Bai, and Shuyue Hu. Self-Harness: Harnesses That Improve Themselves, 2026a. URL https://arxiv.org/abs/2606.09498.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, Urmish Thakker, James Zou, and Kunle Olukotun. Agentic context engineering: Evolving contexts for self-improving language models. In *International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=eC4ygDs02R. Preprint: https://arxiv.org/abs/2510.04618.

Yifan Zhang, Yutong Dai, Juntao Tan, Luyu Yang, Rishi Mullur, Thai Hoang, Zhiyuan Hu, James Zhu, Phil Mui, Silvio Savarese, Ran Xu, and Zeyuan Chen. DarwinX: Evolving Agent Harnesses Through Natural Selection, 2026c. URL https://arxiv.org/abs/2608.07545.

Congjie Zheng, Chuanyi Xue, Bin Liang, Jun Yang, and Changshui Zhang. SEAGym: An Evaluation Environment for Self-Evolving LLM Agents, 2026. URL https://arxiv.org/abs/2606.17546.

Jingbo Zhou, Yusai Zhao, Qi Bao, Jingjia Cao, Zhenghai Chen, Chang Gao, Kaiqi Guo, Muxin Guo, Mingxuan Li, Xinjiang Lu, Yanru Ma, Yixiong Xiao, Zenghui Zhang, Le Zhang, and Hua Wu. OmegaUse-OfficeVal: Benchmarking LLM agents on long-horizon office-suite tasks with economic grounding. *arXiv preprint arXiv:2607.27155*, 2026.

Jian Zhu, Yuzheng Zhang, Zeyao Ma, Bohan Zhang, Armin Schoepf, Daniel Woloch, Peter Yiliu Wang, Guangyu Robert Yang, Samuel Jacob, Siddharth Nagisetty, Abhiram Chundru, Jean Lin, Spencer Mateega, and Jing Zhang. SpreadsheetBench 2: Evaluating agents on end-to-end business spreadsheet workflows. *arXiv preprint arXiv:2606.29955*, 2026.

CONTENTS

A Structural Regimes and Difficulty Calibration 14
A.1 Common Empirical Difficulty Calibration 14
A.2 Presentation 14
A.3 Document 15
A.4 Spreadsheet 16
A.5 Cross-Office 17

B Domain-Specific Task Sources and Adaptation Records 18
B.1 Presentation 18
B.2 Document 18
B.3 Spreadsheet 18
B.4 Cross-Office 19

C Evolution Split and Information Boundaries 19

D Judge Reliability and Human Audit 20
D.1 Recorded Checks and Version Boundaries 20
D.2 Sample Frame and Stratification 20
D.3 Rater Panel and Adjudication 21
D.4 Inter-Annotator Agreement and Judge Alignment 22

E Task Review Interface 22
E.1 Task Quality Review Records 23

F Qualitative Failure Analysis 23
F.1 Human Review of Failure Classifications 23
F.2 Static Evaluation Cases 25
F.3 Evolution Evaluation Cases 26

G Cost and Latency 27
G.1 Solver Resources 27
G.2 Updater Cost and Final Gains 28

H Case Study of Harness Revisions 28
H.1 Revision Details 28
H.2 Crash-Excluded Diagnostic for AHE 30

I Reproducibility and Experimental Configurations 30
I.1 Static Evaluation Details 31
I.2 Evolution Evaluation Details 31

J Conclusion and Limitations 32
J.1 Conclusion 32
J.2 Limitations 32

A Structural Regimes and Difficulty Calibration

Each task is described by its workflow category and empirical difficulty. Workflow categories are modality-specific; difficulty uses the same calibration protocol across all four modalities.

A.1 Common Empirical Difficulty Calibration

Empirical calibration uses one frozen panel of model–harness configurations, keeping model versions, task packages, judges, seed policy, evaluation budget, and resource accounting fixed. Infrastructure failures are retried or excluded; a valid run that omits a required deliverable receives the score the judge assigns. Tasks are sorted by $\bar{r}_i$ and partitioned into precommitted 20/30/30/20 rank bins. A tie at a boundary is resolved first by structural regime and then by stable task ID. For tier ℓ with task set $\mathcal{T}_\ell$, the calibration statistic is

$$\bar{r}_\ell = \frac{1}{|\mathcal{T}_\ell|} \sum_{i \in \mathcal{T}_\ell} \bar{r}_i. \tag{2}$$

The workflow regime describes task requirements and serves as a tie-breaking reference; it is not an alternative difficulty scale.

The calibration panel uses Claude Code, OpenCode, and OpenHands with the Sonnet 5 backbone. Each task contributes one selected completion per harness, and within-modality ranks define the difficulty labels reported here.

A.2 Presentation

Presentation structure is annotated from five observable properties: operation scope, evidence dependency, reasoning and synthesis, PowerPoint structural depth, and constraint coupling. Slide count alone is not a difficulty criterion. These properties place a task into one of four workflow regimes:

S1 Local editing — 15 tasks
Definition A supplied deck receives localized, explicit, and deterministic edits with no external evidence synthesis.
Requirements Typography, footer, border, or other isolated slide-object changes.

S2 Deck-level composition — 16 tasks
Definition One deck requires coordinated changes across objects or slides, or manipulation of a bounded native PowerPoint structure.
Requirements Reordering slides, editing charts or tables, notes, hyperlinks, or reading order.

S3 Cross-source synthesis — 17 tasks
Definition The task requires substantial reconstruction or deck-wide transformation from visual, tabular, textual, or multiple reference materials while satisfying coupled fidelity constraints.
Requirements Image-to-editable diagrams, spreadsheet-to-deck conversion, localization, or template transfer.

S4 End-to-end orchestration — 12 tasks
Definition A business objective requires heterogeneous evidence, long-horizon analysis or reconciliation, and a complete presentation workflow with traceability or multiple deliverables.
Requirements Leadership reviews, board briefings, operational analyses, or presentation packages.

Figure A1: Structural workflow regimes for presentation tasks, with the current structural audit counts. S1–S4 are not the empirical difficulty tiers.

The per-regime counts in Figure A1 measure workflow coverage. Table A1 reports the resulting rank partition.

Table A1: Hybrid-reward calibration of the 60 presentation tasks. Values are percentages; the four rows are precommitted 20/30/30/20 rank bins.

Tier	Tasks	Claude Code	OpenCode	OpenHands	Panel mean $\bar{r}_\ell$	$1 - \bar{r}_\ell$
Easy	12	96.96	97.29	97.98	97.41	2.59
Medium	18	35.51	37.27	40.61	37.80	62.20
Hard	18	18.02	16.53	16.23	16.93	83.07
Expert	12	9.19	6.37	7.11	7.56	92.44

Deterministic judges are combined with task-specific visual rubrics. Of 180 scoring attempts, 179 contained the required deliverable; OpenCode omitted the required PPTX for one task. The panel mean is 37.41%.

A.3 DOCUMENT

Document structure is characterized by the scope of content changes, the depth of native Word objects, evidence synthesis, preservation constraints, and review or release controls. Figure A2 distinguishes four workflow regimes using requirements from the current task pool. Page count and the number of input files do not determine the regime.

D1 Bounded content operations

Definition Extract specified facts or apply an explicit text transformation without substantive evidence synthesis or coupled document reconstruction.

Requirements Turn a concert announcement into a contact sheet; accept all specified contract redlines while preserving the accepted text.

D2 Native document reconstruction

Definition Rebuild or coordinate editable Word structures and formatting from supplied content, without resolving competing factual sources.

Requirements Reconstruct an image as a native training table or flowchart; transfer mathematics into a template with editable equations.

D3 Evidence-grounded synthesis

Definition Derive a report or plan from source evidence and stated rules; reconcile constraints and distinguish supported conclusions from missing information.

Requirements Audit six authorization letters against supplied risk rules; select a feasible trip and document its schedule and budget.

D4 Controlled review and release

Definition Couple evidence-based content decisions with mandatory review, source-preservation, or release controls embedded in the document.

Requirements Integrate policy through tracked insertions and anchored comments; produce a product-selection dossier with a linked evidence matrix and release controls.

Figure A2: Structural workflow regimes for document tasks, with examples from the current pool. D1–D4 are not the empirical difficulty tiers.

The required workflow determines the regime: native-object complexity alone belongs to D2, substantive evidence synthesis to D3, and a mandatory controlled-review or release process to D4. A comment or bookmark alone does not imply D4. Inputs may include PDFs, images, spreadsheets, or emails while the required deliverable remains a Word document; heterogeneous inputs alone do not make a task cross-office. Table A2 reports the empirical partition.

Table A2: Deterministic-reward pilot calibration of the 60 document tasks. Values are percentages; the four rows use 20/30/30/20 rank bins.

Tier	Tasks	Claude Code	OpenCode	OpenHands	Panel mean $\bar{r}_\ell$	$1 - \bar{r}_\ell$
Easy	12	90.59	84.13	85.99	86.90	13.10
Medium	18	50.21	59.52	66.77	58.83	41.17
Hard	18	11.37	13.14	11.22	11.91	88.09
Expert	12	0.00	0.00	0.00	0.00	100.00

The same Sonnet 5 panel contributes 17 fresh upgraded-task evaluations and 43 reused, reused historical scores for unchanged tasks per harness; document scoring is deterministic only. The panel mean is 38.60%. Twenty tasks receive zero across all three harnesses, including all 12 Expert tasks.

A.4 SPREADSHEET

Spreadsheet structure is characterized by the scope of cell operations, formula and cross-sheet dependencies, alignment of source entities and periods, analytical reasoning, and the coupling of assumptions to downstream decisions. Figure A3 separates four workflow regimes. A single-file model can require deep dependency repair, whereas a large table can require only a local transformation; file, sheet, and row counts are therefore not sufficient classification rules.

X1 Local transformations

Definition Apply explicit formatting, cleaning, or cell-level calculations within a bounded range while preserving the source data.
Requirements Clean email text with formulas; format a copied respondent-count sheet without changing its values or merged cells.

X2 Workbook computation

Definition Build or repair coupled formulas and native workbook structures under a supplied data model, without substantive source-conflict reconciliation.
Requirements Create dependent dropdowns; repair named ranges and cascading references; complete linked financial forecasts.

X3 Source reconciliation

Definition Align records, periods, units, or accounting definitions across source tables; retain discrepancies and produce evidence-grounded reconciliations.
Requirements Reconcile vendor statements with payable balances and exclusions; join product sales on two keys while retaining unmatched records from both tables.

X4 Decision and audit workflows

Definition Connect source preparation and validation to a multi-stage scenario, policy-based decision, or qualified audit conclusion, with consistent downstream calculations.
Requirements Link stock and supplier-risk analysis to a replenishment plan; reconcile payroll through monthly, employee, and department views under an explicit deduction scenario.

Figure A3: Structural workflow regimes for spreadsheet tasks, with examples from the current pool. X1–X4 are not the empirical difficulty tiers.

The category depends on the required sequence of operations: a supplied financial model repair can be X2, an evidence reconciliation X3, and a workflow that additionally couples scenarios or operational decisions to those checks X4. A chart or summary sheet alone does not imply X4. Workbook-internal email drafts or recommendations do not create an additional Office modality. Spreadsheet scores use deterministic judges, and Easy–Expert membership follows the same three-harness calibration protocol used for the other modalities.

Table A3: Deterministic-reward pilot calibration of the 60 spreadsheet tasks. Rewards and empirical difficulty are scaled by 100; the four rows use 20/30/30/20 rank bins.

Tier	Tasks	Claude Code	OpenCode	OpenHands	Panel mean $\bar{r}_\ell$	$1 - \bar{r}_\ell$
Easy	12	79.45	69.25	83.29	77.33	22.67
Medium	18	51.33	36.65	61.61	49.86	50.14
Hard	18	23.58	26.31	27.44	25.78	74.22
Expert	12	0.09	6.47	8.00	4.85	95.15
All	60	38.38	34.03	44.98	39.13	60.87

Table A3 uses 180 valid scores: per harness, seven upgraded tasks are newly executed and 53 unchanged tasks reuse saved outputs with task and verifier versions matched. The panel mean is 39.13%.

A.5 CROSS-OFFICE

Cross-office structure is annotated from five observable properties: the number and heterogeneity of applications and artifacts, dependency-graph depth, evidence reconciliation, coupling among deliverables, and workflow and review requirements. These properties place a task into one of four workflow regimes:

C1 Direct artifact handoff
Definition An explicit, one-way transfer or embedding connects two Office artifacts without reconciling competing evidence.
Requirements Insert or link a supplied table, chart, or presentation while preserving editability.

C2 Coordinated production
Definition A bounded source of truth drives coordinated updates to multiple Office deliverables under explicit consistency constraints.
Requirements Propagate approved facts into a workbook, document, deck, or communication package.

C3 Cross-source reconciliation
Definition Multiple heterogeneous sources must be joined, checked for conflicts or missing values, and converted into mutually consistent deliverables with traceability.
Requirements Reconcile records, apply policy precedence, document exceptions, and produce linked analysis and reports.

C4 End-to-end operational workflow
Definition A long-horizon business objective couples multi-stage decisions, governance constraints, and several dependent deliverables into one auditable package.
Requirements Compliance, onboarding, finance, research, or operations workflows with evidence trails and communication outputs.

Figure A4: Structural workflow regimes for cross-office tasks. C1–C4 are not the empirical difficulty tiers.

The four regimes explain the observable structure of a task; the final ordinal label is assigned by the shared empirical protocol. Table A4 reports the current 20/30/30/20 rank partition.

Table A4: Hybrid-reward calibration of the 60 cross-office tasks.

Tier	Tasks	Claude Code	OpenCode	OpenHands	Panel mean $\bar{r}_\ell$	$1 - \bar{r}_\ell$
Easy	12	72.20	71.91	65.51	69.87	30.13
Medium	18	18.61	15.93	24.69	19.74	80.26
Hard	18	10.32	3.93	8.99	7.75	92.25
Expert	12	0.02	0.00	0.26	0.09	99.91

Thirty-six tasks use the 70/30 deterministic–visual reward and 24 use deterministic reward only. Of 180 evaluations, 153 contained the required deliverables and 27 valid runs omitted them. The panel mean is 22.24%.

B Domain-Specific Task Sources and Adaptation Records

Each task is adapted individually from a public, license-compatible source. Adaptation freezes the required inputs, maps the original request into an explicit deliverable contract, and rebuilds or strengthens the evaluator where the upstream check is insufficient. Shared sources, templates, and near-duplicate workflows receive leakage-group identifiers for partitioning. Full commit hashes, input checksums, original requests, licenses, and per-task transformation notes remain inside each released task package.

B.1 Presentation

Table A5 reports the frozen source composition for the 60 presentation tasks.

Table A5: Frozen source inventory for the 60 presentation tasks.

Upstream source	Tasks	License and frozen revision	Upstream task identifiers
PPTArena	31	MIT; `6f26a63` (code), `20fb890` (data)	Cases 001, 003, 004, 014, 015, 016, 021, 023, 025, 027, 028, 029, 030, 031, 036, 037, 041, 044, 047, 050, 053, 059, 061, 064, 067, 068, 071, 074, 082, 083, and 093.
OmegaUse-OfficeVal	17	Apache-2.0; `ffbeecb` (code), `cd6ba6d` (data)	OfficeVal 041, 043, 046, 051, 054, 055, 056, 060, 063, 064, 066, 068, 069, 070, 074, 075, and 076.
AutomationBench	4	MIT; `4a8e106`	Marketing 1115; Finance 4040, 4044, and 4062.
TheAgentCompany	4	MIT; `98b68ef`	`pm-prepare-meeting-with-customers`, `pm-present-gitlab-info-as-ppt`, `hr-internal-tooling-slides`, and `ds-stock-analysis-slides`.
Cowork-Bench	2	Apache-2.0; `d943e75`	`sf-department-budget-ppt` and `ppt-clickhouse-executive`.
OfficeBench	2	Apache-2.0; `b978b80`	Tasks 3-2/subtask-0 and 3-5/subtask-0.
Total	**60**		

B.2 Document

Table A6 reports the frozen source composition for the 60 document tasks, which are restricted to tasks whose solver-facing deliverables are native, editable DOCX files.

Table A6: Frozen source inventory for the 60 document tasks.

Upstream source	Tasks	License and frozen revision	Upstream task identifiers
OfficeBench	46	Apache-2.0; `b978b808`	1-12/0, 1-12/1, 1-12/2, 1-13/1, 1-13/0, 1-15/0, 1-15/2, 1-15/1, 1-16/0, 1-16/1, 1-19/4, 1-19/1, 1-19/0, 1-3/0, 2-18/0, 2-18/1, 2-18/2, 2-19/0, 2-2/1, 2-2/0, 2-20/0, 2-20/1, 2-20/2, 2-21/0, 2-22/1, 2-22/0, 2-26/3, 2-26/1, 2-30/0, 2-36/0, 2-37/3, 2-39/3, 2-39/1, 2-39/0, 2-6/0, 2-7/0, 3-22/0, 3-56/0, 3-58/0, 3-60/0, 3-63/0, 3-77/0, 3-8/7, 3-82/0, 3-84/0, and 3-91/0.
OmegaUse-OfficeVal	9	Apache-2.0; `ffbeecb8`	`officeval_003`, `officeval_006`, `officeval_010`, `officeval_018`, `officeval_022`, `officeval_029`, `officeval_033`, `officeval_036`, and `officeval_040`.
llm-docx-editing	3	MIT; `498a4946`	D3, F6, and S4.
Cowork-Bench	1	Apache-2.0; `d943e75b`	`rzd-msk-spb-team-trip-gcal-word`.
FORTE	1	MIT; `345c1ec1`	`Legal-020`.
Total	**60**		

B.3 Spreadsheet

Table A7 reports the frozen source composition for the 60 spreadsheet tasks, each with one native XLSX solver deliverable.

Table A7: Frozen source inventory for the 60 spreadsheet tasks.

Upstream source	Tasks	License and frozen revision	Upstream task identifiers
OfficeBench	8	Apache-2.0; `b978b808`	1-6/1, 1-9/2, 1-9/4, 1-5/3, 1-10/3, 1-10/2, 1-9/0, and 1-14/1.
OmegaUse-OfficeVal	2	Apache-2.0; `ffbeecb8`	`officeval_088` and `officeval_093`.
SpreadsheetBench-2 sample	19	CC-BY-SA-4.0; `d8a14da1`	Debugging/01_03, Debugging/02_03, Debugging/03_03, Debugging/04_03, Debugging/08_03, Debugging/09_01, Financial_Model/02_03, Financial_Model/03_02, Financial_Model/05_01, Financial_Model/07_02, Financial_Model/12_01, Financial_Model/13_02, Financial_Model/16_02, Financial_Model/18_03, Financial_Model/20_03, Template/06_19, Template/14_09, Visualization/Task_129, and Visualization/Task_1437004.
Spreadsheet-RL	12	CC-BY-SA-4.0; `6f274cd4`	`excelforum/excel-general/task-job-6f73fa37587dfb5e`, `excelforum/excel-general/task-job-1d7e618b48fc03cc`, `excelforum/excel-general/task-job-93b8d5ae5b1f85f1`, `excelforum/excel-general/task-job-5cd12fc29412660e`, `excelforum/excel-general/task-job-2d603b0f50935350`, `excelforum/excel-general/task-job-654f78f4eb0df932`, `excelforum/excel-general/task-job-2e1e331a1b9dc473`, `excelforum/excel-general/task-job-1813a28727a048cb`, `excelforum/excel-general/task-job-f59390cb071bb9fd`, `excelforum/excel-general/task-job-68594670fd5aafc2`, `excelforum/excel-general/task-job-9926a63ec2ac727f`, and `excelforum/excel-general/task-job-b07dec0d38d63663`.
Workspace-Bench-Lite	4	Apache-2.0; metadata `60b08b1c`, workspaces `e245d63b`	tasks 159, 255, 277, and 386.
Cowork-Bench	8	Apache-2.0; `d943e75b`	`course-enrollment-analytics-dashboard`, `financial-audit-cross-system-reconciliation`, `insales-inventory-reorder-forecast`, `moex-sector-rotation-dashboard`, `sf-regional-sales-forecast`, `sf-sla-breach-audit`, `supplier-contract-compliance-audit`, and `inventory-supply-chain-optimization`.
AutomationBench	4	MIT; `4a8e1061`	`finance.annual_budget_prep`, `finance.cash_flow_forecast`, `finance.gl_variance_analysis`, and `finance.vendor_statement_reconciliation`.
TheAgentCompany	2	MIT; `98b68ef8`	`finance-budget-variance` and `ds-format-excel-sheets`.
FORTE	1	MIT; `345c1ec1`	`Finance-018`.
Total	**60**		

B.4 Cross-Office

Table A8 reports the source composition for the 60 cross-office tasks.

Table A8: Source inventory for the 60 cross-office tasks.

Upstream source	Tasks	License and frozen revision	Upstream task identifiers
AutomationBench	42	MIT; `4a8e106`	Sales 1202; Finance 4001; HR 5004, 5011, 5057, 5070, 5090, and 5101–5135.
Cowork-Bench	9	Apache-2.0; `d943e75`	Literature review, meal planning, program committee, financial audit, portfolio analysis, refund analysis, scholarly trends, tax propagation, and student retention tasks.
OfficeBench	5	Apache-2.0; `b978b80`	Tasks 3-6, 3-8, 3-19/3-20, 3-69, and 3-91.
OmegaUse-OfficeVal	4	Apache-2.0; `ffbeecb` (code), `cd6ba6d` (data)	OfficeVal 059, 078, 080, and 088.
Total	**60**		

C Evolution Split and Information Boundaries

Presentation, document, and spreadsheet tasks use a 30/10/20 feedback–selection–final allocation. All 60 cross-office tasks are reserved for compositional OOD evaluation. Tasks sharing a source

document, template, entity, or near-duplicate workflow are assigned to one split through a common leakage group.

Feedback trajectories and rubric diagnostics are available for updates; selection exposes a metered aggregate score. Final tasks, answers, and previous static-run traces are not available to the updater or used to select revisions. Each selected revision is frozen before final evaluation.

Table A9: Evolution split allocation. Feedback exposes trajectories, artifacts, scores, and rubric diagnostics; selection returns a metered aggregate signal; final tasks remain inaccessible to evolving agents until the selected harness is frozen.

Modality	Feedback	Selection	Final ID	Final OOD
Presentation	30	10	20	0
Document	30	10	20	0
Spreadsheet	30	10	20	0
Cross-office	0	0	0	60
Total	90	30	60	60

D Judge Reliability and Human Audit

D.1 Recorded Checks and Version Boundaries

The implementation checks in Table 5 cover all 240 tasks in the current release. Each task uses its frozen Docker image, reference-solution script, and deterministic judge, with networking disabled. We retain one result for each of three conditions: the reference solution's output, an empty output directory, and a corrupt output package. The corrupt package keeps every reference output filename but replaces every file's contents with fixed, nonempty invalid bytes. The expected rewards are 1, 0, and 0, respectively, with tolerance 10^{-6}. This yields 720 reported checks. Task files and source inputs are checked for changes, and raw judge results, rewards, image identifiers, and file hashes are retained. The counts cover this release only.

All 240 reference solutions receive reward 1. For both negative conditions, seven document tasks retain positive credit: six receive 0.10 for unchanged inputs, and one receives 0.25 for unchanged inputs and the absence of placeholder content. Among spreadsheet tasks, 15 empty-output cases and 16 corrupt-output cases receive positive rewards of 0.05–0.20. Another 18 tasks receive negative rewards of -0.20 to -0.10 under both conditions because of existing penalties, which move the score away from zero without awarding credit. The table measures exact expected-score behavior: positive residual rewards locate gaps in invalid-output handling. The audit uses the frozen judges and existing agent scores.

These checks cover full reference outputs and two severe invalid-output cases; the human audit below covers judge behaviour on agent artifacts. A separate, targeted check on seven revised spreadsheet tasks found that constant-formula substitutions reduced rewards to 0.75–0.79.

D.2 Sample Frame and Stratification

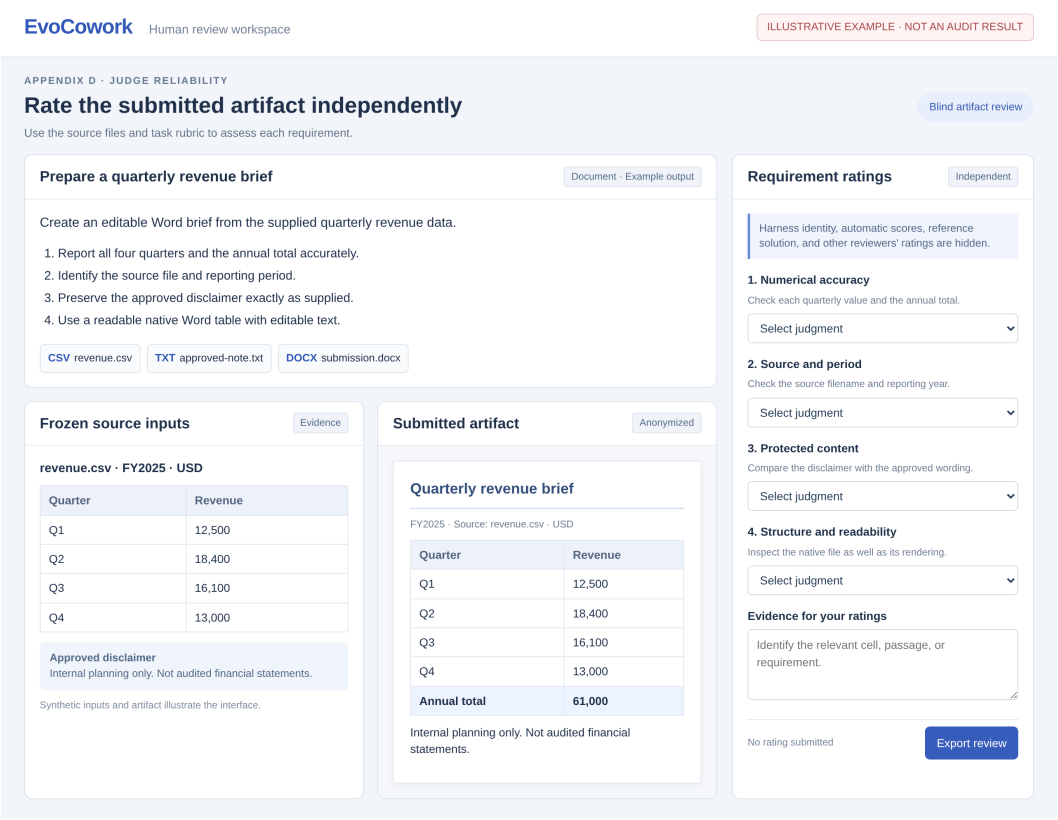
This audit rates *agent outputs* to measure whether the automatic judges agree with human readers. It is separate from the task review in Appendix E, which rates *task packages* and their reference answers.

The sample frame is every scored candidate output from the model–harness configurations reported in Table 3, frozen at a recorded snapshot before sampling: 1,440 outputs for the six configurations completed at the time of sampling. From that frame we draw 240 outputs, 60 per modality.

Within a modality, the draw is stratified by frozen difficulty tier, fifteen outputs per tier, so that Easy through Expert are equally represented rather than following the 12/18/18/12 task counts. Within each modality–tier cell, outputs are stratified again by automatic score into four bands — exactly zero, $(0, 33]$, $(33, 67]$, and $(67, 100]$ — taking at least two outputs from every non-empty band and allocating the remaining draws in proportion to band frequency. Zero-score outputs, including

missing deliverables, stay in the frame: they are where a judge error is most consequential. At most two outputs per task enter the sample, so each modality covers at least 30 distinct tasks, and harnesses are balanced as evenly as each cell allows.

Raters receive the task instruction, the frozen source inputs, the submitted artifacts, and the task-specific rubric. They do not receive the harness identity, the automatic score, the reference solution, or file names that reveal the producing configuration; presentation order is randomised. Exact-recreation tasks are the one exception on the reference solution, because the reference rendering is part of the stated requirement, matching the policy used for the visual evaluator. Outputs that fail to render because of an infrastructure error are repaired before review.



D.3 Rater Panel and Adjudication

Each modality has a panel of three raters with professional experience in the corresponding office application. Raters are independent of the material they audit: none adapted the sampled tasks, implemented or revised their judges, or took part in the task review of Appendix E. Each rater serves on at most two modality panels.

Every sampled output receives two independent ratings, assigned by rotating over the three pairs within a panel. This yields 120 ratings per modality and 480 ratings in total. Raters record a decision for each rubric requirement on an ordered scale — satisfied, partially satisfied, not satisfied — together with free-text notes, and they submit before seeing the other rating. Where the two ratings disagree on a requirement, the third rater on that panel adjudicates and their decision becomes the human consensus for that requirement. Adjudication assignments follow from which pair produced the ratings, so each rater covers a different subset of the sample.

Before the audit proper, all raters score a shared calibration set of eight outputs, two per modality, and discuss the rubric wording. Calibration ratings are discarded.

D.4 Inter-Annotator Agreement and Judge Alignment

For the ordered rubric decisions, inter-annotator agreement (IAA) is measured with ordinal Krippendorff's α on the independent, pre-adjudication ratings, which accommodates raters covering different subsets of the sample.

Judge alignment is measured against the adjudicated consensus as score error. The human rubric decisions are aggregated into a task score using the same weights the judge applies, including the 70/30 deterministic–visual combination in Equation 1, and we report the mean absolute error (MAE) between the judge score and this human score.

Table 5 reports audited outputs, rater counts, α, and score MAE by modality. Score MAE is in points on the 100-times-reward scale.

E Task Review Interface

To verify task quality, we randomly sampled tasks from each modality and conducted a structured human review using a purpose-built web interface. Each reviewer was presented with the task instruction, input files, and reference answer for one task at a time. They assessed three aspects: (1) whether the instruction is unambiguous, (2) whether the reference answer is factually correct and complete, and (3) whether the reference answer fully satisfies the stated requirements. Reviewers could record notes and assign an overall verdict of Pass, Flag, or Reject. Flagged and rejected tasks were revised before inclusion in the released benchmark.

Figure A5 shows the review interface. The left panel displays the task instruction, input files, and reference answer. The right panel shows the reviewer's assessment form and the progress queue for the current batch.

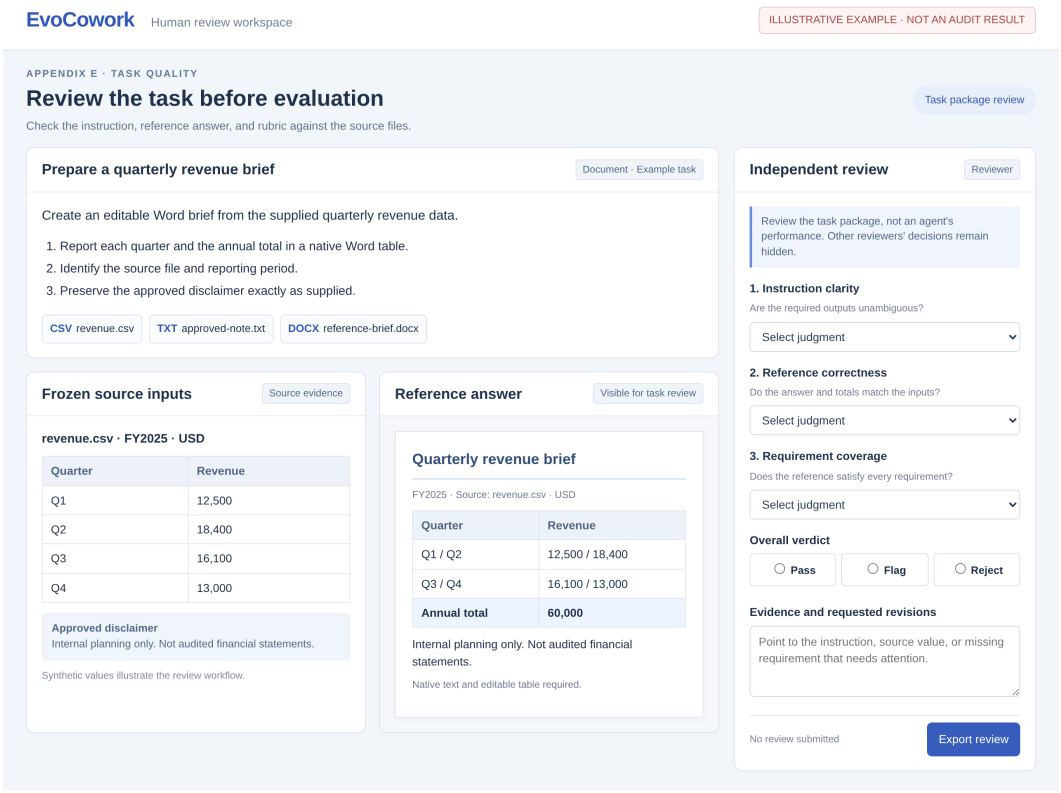


Figure A5: Task review interface used for human quality verification of sampled tasks. Reviewers assess instruction clarity, reference answer correctness, and requirement coverage. The right panel tracks batch progress and flags tasks for revision.

E.1 Task Quality Review Records

The review covers 24 of the 60 tasks in each modality, 96 tasks in total. Tasks are drawn at random within frozen difficulty tier, six per tier, so that Easy through Expert contribute equally rather than in proportion to the 12/18/18/12 counts. The draw is stratified by upstream source as well, taking at least one task from every source that contributes to a modality, so that adaptation decisions specific to one source cannot escape review. Each modality has a three-person panel drawn from the authors and from contributors who prepared task packages for other modalities; nobody reviews a task they adapted themselves.

Every sampled task receives two independent reviews. Reviewers see the instruction, the frozen inputs, the reference answer, and the task-specific rubric, and record the three judgments above on an ordered scale — satisfied, partially satisfied, not satisfied — together with a Pass, Flag, or Reject verdict and free-text notes. They submit before seeing the other review. Where the two verdicts differ, the third panel member adjudicates and their verdict becomes the recorded outcome, giving 192 independent reviews plus adjudications across the sample. Agreement is measured on the independent, pre-adjudication judgments with ordinal Krippendorff's α, alongside the share of tasks on which both reviewers reach the same verdict.

The review returned 84 Pass, 10 Flag, and 2 Reject verdicts. The two independent reviews agreed on the verdict for 92 of 96 tasks; the third panel member adjudicated the remaining 4, and ordinal Krippendorff's α over the pre-adjudication judgments is 0.86. The 12 flagged or rejected tasks were revised and rechecked before inclusion: seven for ambiguous instruction wording and five for reference answers that missed a stated requirement.

Task adaptation, reference-answer construction, and rubric drafting are AI-assisted and human-checked: an agent proposes the deliverable contract, the reference solution, and the requirement list from the upstream task, and a person verifies each against the frozen inputs before the package enters review. The review above then treats that package as the object under audit. Verdict counts, the issues they name, and the revision status of each flagged task are kept in the review log retained with the task package.

F Qualitative Failure Analysis

Failure-rate coverage. Table 6 counts a delivery failure when at least one required file is missing or empty. Static denominators are normally 60 per modality and 240 overall. All static configurations have 240 task scores. The Sonnet–Claude Code delivery audit covers all 240 tasks; all 60 PPT outputs contain every required nonempty file. One GLM–OpenCode Word run has conflicting artifact and judge records; this run remains unclassified, so the denominators are 59 for Word and 239 overall. Final-test denominators are 20 per single modality, 60 for cross-office, and 120 overall. Quality columns divide the number of runs with at least one rule-identified violation by all 240 static or 120 final tasks. Skipped or unresolved checks are not counted as additional violations, so the columns report violations the rules identify.

We inspect two static outputs and two paired final-test cases. Each example is checked against the task files, submitted artifacts, and, where needed, execution trace. These are illustrative cases rather than a random sample or an estimate of failure frequency. A failed judge check alone is not treated as sufficient evidence of an agent error.

F.1 Human Review of Failure Classifications

This review checks whether the rule-based and agent-assisted classification behind the quality columns assigns cases to the right category. It is separate from the audit in Appendix D, which asks whether the judge scores match human readers; here the score is taken as given and only the category label is under review.

The sample frame is every case that the classification flagged in Table 6, together with the cases whose requirement text matched no rule. From that frame we draw 20 cases per quality category, 140 in total, stratified across the eleven reported configurations so that no configuration supplies more than a quarter of any category, and stratified within a category across the four modalities where

the check applies. Cases that reached the agent-assisted pass are sampled at twice their share of the frame, since they carry the classification decisions least constrained by an explicit rule. Unmatched checks form a separate stratum of 20 cases, drawn from the same frame.

Each modality panel is the one described in Appendix D.3: three reviewers, two independent labels per case, adjudication by the third where they differ, giving 320 independent labels plus adjudications. Reviewers see the failing check, the requirement it belongs to, the submitted artifact, and the relevant source inputs. They do not see the category the classification assigned, so labels are recorded before any comparison with the automatic mapping. For each case a reviewer assigns one primary category from the seven, any additional categories the same case satisfies, or one of two outcomes outside the taxonomy: the check identifies no error in the artifact, or the evidence does not settle the question. The second outcome is kept separate from the first, since an unresolved case is not a confirmed non-error.

Agreement is measured on the independent, pre-adjudication labels with nominal Krippendorff's α over the primary category, and the adjudicated label is compared with the automatic mapping to give a per-category agreement rate. Where adjudication overturns a mapping, both the corrected label and the rule or prompt that produced the original are recorded, and a category definition is revised only when the same disagreement recurs across configurations. Corrections that change any proportion in Table 6 are applied to the table and noted with the affected cells; the review log retains the case identifiers, the two independent labels, and the adjudicated outcome.

Figure A6 illustrates the review interface with a synthetic missing-formula example.

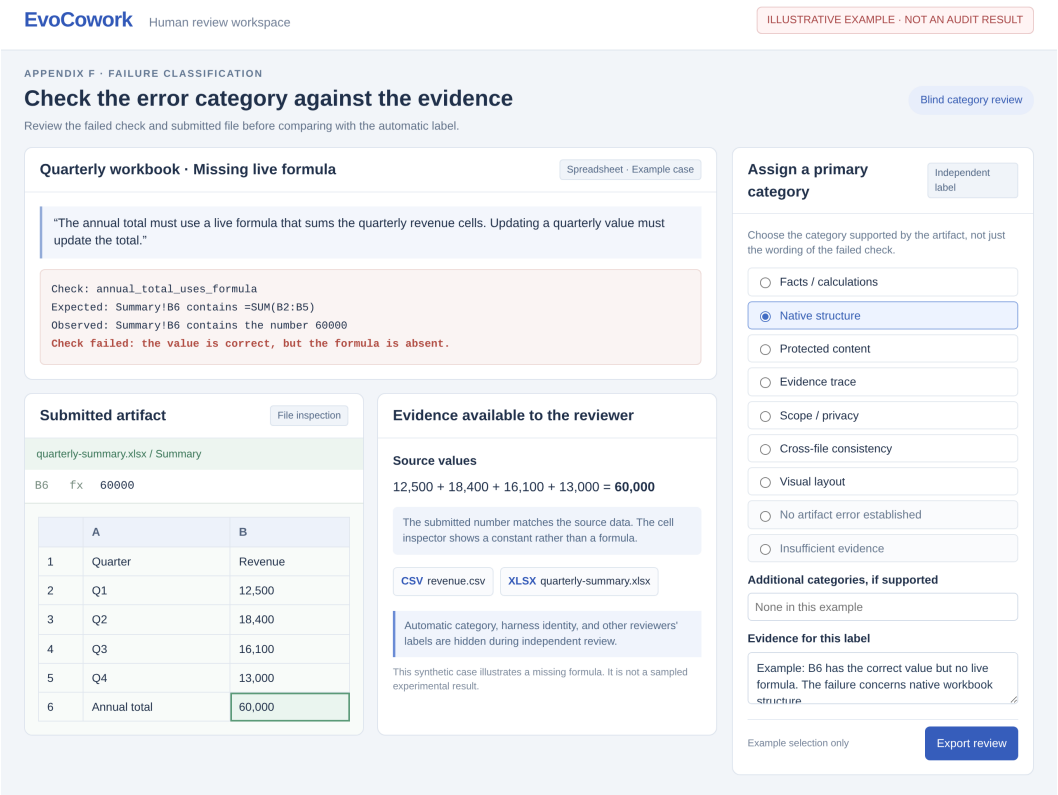


Figure A6: Illustrative interface for reviewing failure classifications. The left panels show the requirement, failed check, and artifact evidence; the right panel records category labels and supporting evidence. The synthetic example shows a correct total stored as a constant instead of the required live formula.

F.2 Static Evaluation Cases

S1: An editable slide changes the source facts. The NVIDIA briefing task asks for six editable slides based on a supplied financial report. Sonnet 5 with Claude Code produces a native table, but slide 3 changes several source values (Figure A7). Gross margin becomes 64.0% instead of 64.6%; prior-year operating expenses and income are also incorrect. Checking file existence and editability would miss these errors. The numbers must also be checked against the source report.

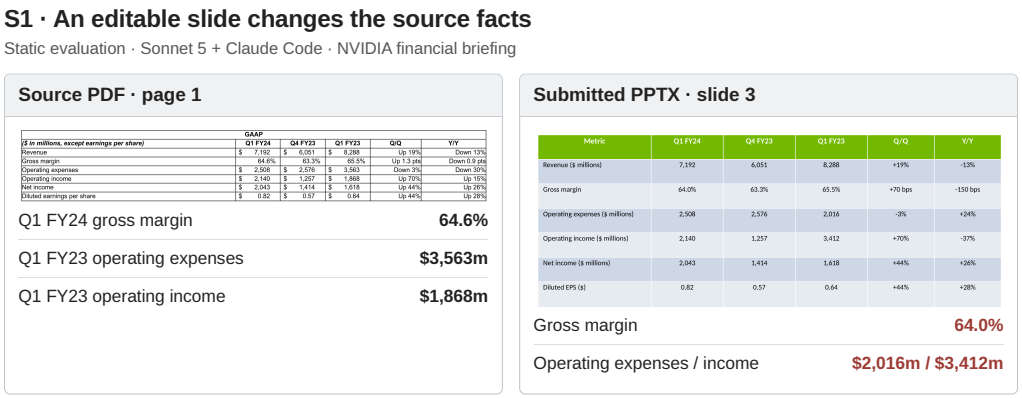


Figure A7: Static case S1: incorrect facts in an editable slide. Crops show the supplied PDF and the actual PPTX rendered with LibreOffice; callouts identify the mismatched cells.

S2: Product images retain promotional overlays. The product-comparison task asks for a workbook with washer images cropped from five source documents. It explicitly excludes marketing poster text. GLM 5.3 with Claude Code embeds five images, but two crops retain promotional overlays (Figure A8). The image anchored at E2 still contains part of a badge; E5 retains a blue text banner. File existence and image count therefore do not establish that the visual requirement is met. These defects can be checked directly in the embedded images.

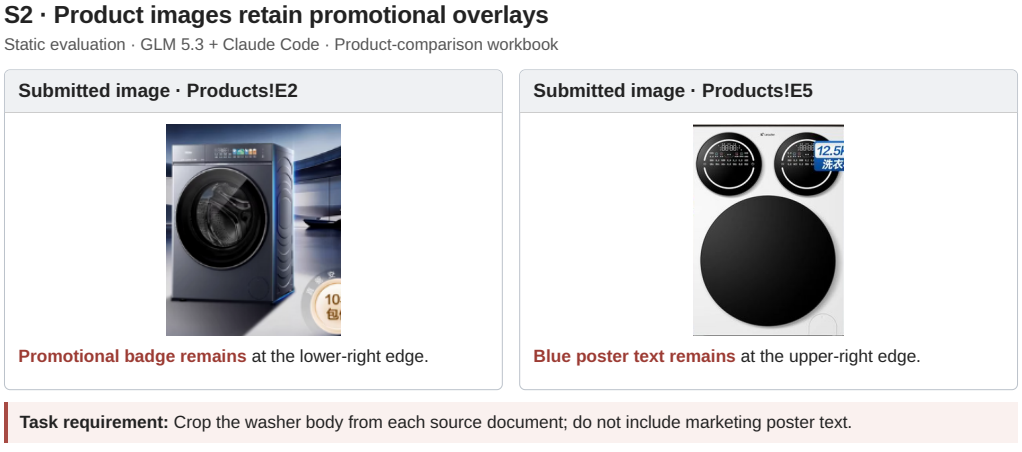


Figure A8: Static case S2: unwanted promotional overlays in product images. Both images are extracted unchanged from the submitted XLSX. The worksheet drawing anchors them at E2 and E5 of `Products`.

F.3 Evolution Evaluation Cases

D1: The updated agent stops before completing the package. The cafeteria task requires five files. On the same final task, H_0 saves all five, while ACE omits `meal_prep_calendar.ics` (Figure A9). ACE stops at the token-budget guard after 33 recorded model calls. Both runs reach that guard, but H_0 has already created every required file. The missing calendar fails the delivery hurdle, giving ACE zero reward. Work on four files does not compensate for the missing part of this coordinated package.

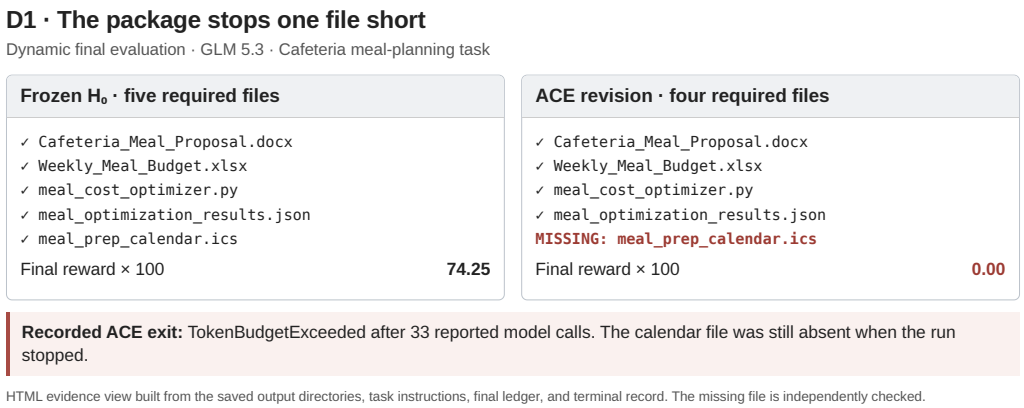


Figure A9: Dynamic case D1: ACE leaves a required calendar undelivered. Panels show file presence in the saved output directories, with scores and the exit condition from the final ledger and terminal record.

D2: A revised harness fails during error recovery. The shipping-and-tax task requires five coordinated files. H_0 produces all five, while the selected AHE candidate stops after four recorded model calls without saving any (Figure A10). Its trace shows a format error followed by a crash in the candidate's completion check, which calls `split` on a list-valued message. The revised harness fails during error recovery. It must remain executable when responses take unexpected forms.

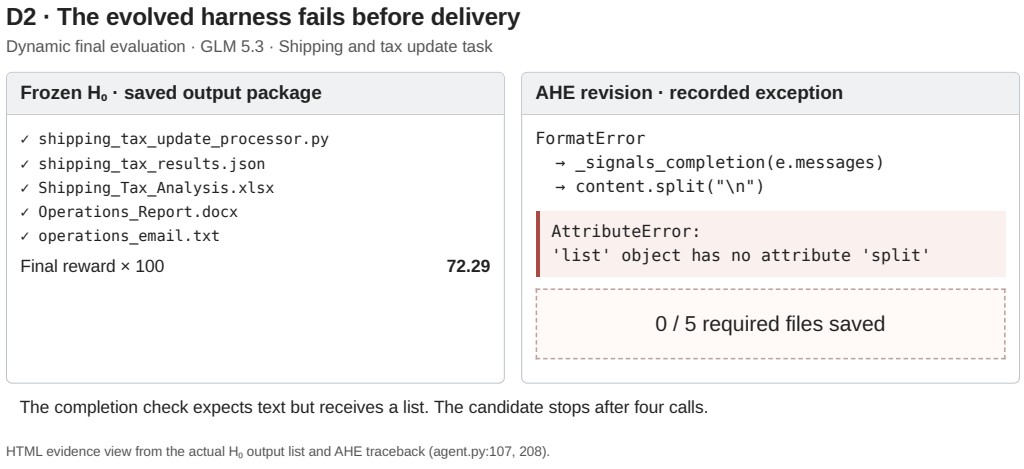


Figure A10: Dynamic case D2: AHE's completion check crashes. Panels use H_0's saved file list and the actual AHE traceback; no AHE artifacts were saved.

G Cost and Latency

G.1 Solver Resources

Measurement scope. Table A10 reports the solver resources of the selected task attempts used in the static and final-test results. Token counts sum reported input and output tokens across calls, including cached input once. For MSA, we use the selected attempt's gateway records; the other harnesses use their recorded agent counters. Cache-write and retry coverage vary across these counters. Execution time runs from agent start to finish, including tool use and request waits. It excludes container setup, judging, queueing, and harness search. Failed task outcomes are retained. Missing measurements are excluded rather than replaced by zero.

Table A10: Solver resources per task in static and evolution evaluations. Each pair gives mean reported tokens (T, thousands) and mean execution time (t, minutes). Coverage lists the numbers of available token and time records, (n_T, n_t); missing records are excluded from the corresponding mean. All includes failed task outcomes. Update generation is excluded.

Model	Harness	Coverage	PPT		Word		Excel		Cross-office		**All**	
		(n_T, n_t)	T	t	T	t	T	t	T	t	**T**	**t**
Static evaluation	240 tasks: 60 per track											
Sonnet 5	Claude Code	239, 239	263	2.1	933	8.2	904	11.0	756	5.3	**716**	**6.7**
	OpenCode	240, 240	229	4.0	630	7.3	334	9.8	506	5.7	**425**	**6.7**
	OpenHands	228, 240	263	3.2	428	7.1	885	9.1	836	8.0	**610**	**6.8**
	MSA	240, 240	1088	10.2	740	8.5	797	7.9	1389	13.8	**1003**	**10.1**
GLM 5.3	Claude Code	240, 240	1029	10.0	739	6.9	744	7.3	1233	9.6	**936**	**8.5**
	OpenCode	240, 240	646	12.5	561	7.2	1097	9.5	377	7.8	**670**	**9.3**
	OpenHands	221, 240	1057	9.2	683	5.8	685	9.2	1604	11.5	**1028**	**8.9**
	MSA	240, 240	1108	10.1	720	7.1	1018	9.4	1524	14.7	**1093**	**10.3**
Evolution evaluation	Same final 120 tasks: 20 per single-modality track, 60 cross-office											
GLM 5.3	MSA H_0	120, 120	1159	9.7	748	7.7	1126	10.6	1524	14.7	**1267**	**12.0**
	MSA + ACE	120, 120	993	10.5	939	8.8	1067	11.5	1461	13.3	**1230**	**11.8**
	MSA + AHE	120, 120	542	7.3	588	6.4	521	5.9	894	9.4	**722**	**8.0**

Observed solver use. The reported static configurations average roughly 0.42–1.09 million tokens and 6.7–10.3 minutes per task. On the shared final set, H_0 averages 1.27 million tokens and 12.0 minutes, ACE 1.23 million and 11.8 minutes, and AHE 0.72 million and 8.0 minutes. Early termination, including the candidate crash in Figure A10, reduces both tokens and time while leaving the requested work incomplete, so resource use is read together with completed work. Cross-office tasks take longer than each single-modality track for all three configurations in the final comparison.

Update generation costs. Separate updater records contain 2.75 million reported tokens for ACE and 12.66 million for AHE, covering the generation and debugging of updates; solver calls on feedback and selection tasks are accounted separately. Their budget accounting charges 2.95 and 13.04 million tokens, respectively, including conservative charges where usage was unavailable.

G.2 Updater Cost and Final Gains

Table A11 places the recorded resources next to the gains in Table 4. Updater tokens include the formal search history, including unsuccessful updates, rather than only the steps that produced the selected revision. Final-inference tokens are summed over the same 120 task attempts used for the reported scores.

Table A11: Recorded resource use and final gains with GLM 5.3 and a shared MSA seed. Tokens are reported input plus output, in millions (M). Updater use is cumulative over the formal search; final inference covers the 120 selected task attempts. Gains are score points relative to H_0, computed before rounding. H_0 is frozen and runs no updater. These stages exclude feedback/selection solver calls and judging, so their sum is not a complete experiment cost.

	Reported tokens (M)		Final gain over H_0		
Method	Updater	Final inference	In-domain	Cross-office	Overall
Frozen H_0	n/a	152.07	0.00	0.00	0.00
ACE	2.75	147.64	**+4.90**	**+0.03**	**+2.46**
AHE	12.66	86.65	**+2.64**	**−1.72**	**+0.46**

In these runs, AHE uses 4.60 times as many reported updater tokens as ACE, but gains 0.46 overall points compared with ACE's 2.46. Their in-domain gains are 2.64 and 4.90 points; cross-office gains are -1.72 and $+0.03$ points. More updater computation therefore does not correspond to a larger gain in this comparison. AHE uses fewer tokens during final inference, but the candidate failure in Figure A10 shows why low inference use must be read together with completed work.

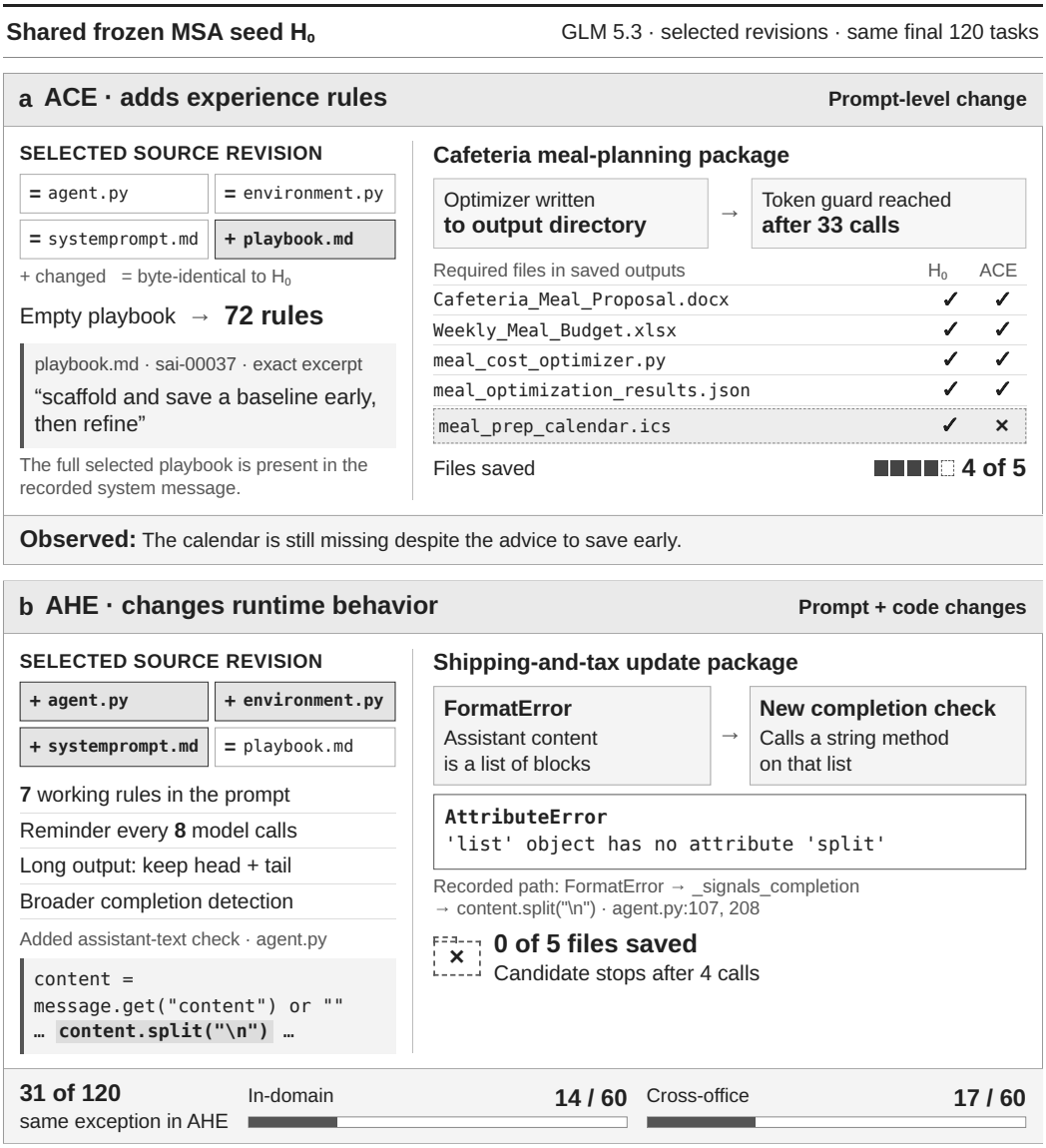
H Case Study of Harness Revisions

We compare the exact frozen revisions used for the final results with their shared initial harness. File hashes match the candidate IDs in all 120 final records for each method. Figure A11 connects the changes to two recorded task outcomes. Table A12 gives details of the source changes on the next page.

H.1 Revision Details

Table A12: Changes in the selected ACE and AHE harnesses relative to H_0. Entries describe source behavior, not separately measured contributions to score. The model, API, and inference budgets remain outside these revisions.

Component	Frozen H_0	ACE	AHE
Changed files	Initial four-file snapshot	`playbook.md`	`agent.py`, `environment.py`, `systemprompt.md`
Instructions	Base office prompt; empty playbook	Adds 72 experience rules; base prompt unchanged	Adds seven working rules to the prompt; playbook stays empty
Progress cues	No periodic reminder	Unchanged	Adds a reminder every eight model calls
Output handling	No truncation inside `environment.py`	Unchanged	Above 12,000 characters, keeps the first 8,000 and last 3,500 plus a notice
Completion	Signal must be the first output line of a successful command	Unchanged	Accepts any standalone output line; adds an assistant-text check after a format error



Evidence boundary. ACE's example does not identify a harmful rule. AHE's trace locates a failure in added code; its aggregate score also includes these runtime errors.

Figure A11: Selected harness changes and recorded outcomes. (a) ACE adds a playbook but still misses a required calendar. (b) AHE's added completion check crashes on list-valued content. Panels reorganize verified source, tool actions, file inventories, and terminal records; the bars count the same exception across all AHE final attempts.

ACE: experience rules without changes to the execution loop. ACE changes only `playbook.md`, from an empty file to 72 entries (48,917 bytes). The agent loop, environment code, and base system prompt match H_0 byte for byte. The entries cover native Office objects, exact labels, checks against source files, and saving deliverables before the budget runs out. Some explicitly target how checks extract text, including duplicating key figures in tables and text boxes. These rules could improve extractability; they also make exact wording and the scope of each rule important.

The cafeteria run in Figure A9 receives this full playbook. Rules `sai-00037`, `code-00039`, and `misc-00068` advise creating files early and reserving time to submit. The trace records the opti-

mizer being written directly to the output directory, but the run still ends with four of five required files: guidance in the playbook does not by itself produce a complete package.

AHE: changes to reminders, tool output, and completion handling.AHE leaves the playbook empty and changes the other three files. Its new instructions emphasize early file creation, short commands, and concrete checks before submission. The agent adds a reminder every eight model calls, and the environment shortens long command output while retaining its beginning and end. These changes target prolonged inspection; truncation trades that against hiding information in the middle of an output.

A separate change broadens completion detection. The new format-error handler checks assistant text for the completion signal, using `content.split("\n")`. In the shipping-and-tax run (Figure A10), the response content is a list of blocks. That new check raises `AttributeError` after four recorded model calls, before any required file is saved. The source diff and traceback locate this failure in an added completion check. The same exception is recorded in 31 of 120 AHE final attempts: 17 cross-office and 14 in-domain. No such exception is recorded for H_0 or ACE. These crashes do not all receive zero reward, since some runs save artifacts before failing. This repeated compatibility problem limits how much of AHE's aggregate result can be attributed to the intended update strategy.

H.2 Crash-Excluded Diagnostic for AHE

Table A13 splits AHE's 120 final-test runs by whether the added completion check crashed. The decline in Table 4 is concentrated in the crashing runs, which average 6.04 overall and 0.45 on cross-office tasks, against 48.75 and 37.44 on the runs that do not trigger the crash. On those runs, AHE scores above H_0 on the same tasks by 6.41 points overall and 4.87 points cross-office. The grouping is defined after seeing the failures, so the clean-subset difference is a diagnostic of the crash's contribution rather than a repaired-AHE result; the reported scores keep every run, crashed or not.

Table A13: Diagnostic split of AHE's final-test runs by whether the added completion check crashed. Scores are 100× mean reward on each task group; the clean Δ compares AHE with H_0 on the runs that do not trigger the crash. The split is a post-hoc diagnostic by run-log status and does not represent a repaired AHE.

Task group	Runs	Crashes	Share	AHE crashed	AHE clean	H_0 same tasks	Clean Δ
All final tasks	120	31	25.8%	6.04	48.75	42.34	**+6.41**
In-domain	60	14	23.3%	12.84	59.32	51.48	**+7.84**
PPT	20	5	25.0%	13.91	66.19	59.17	**+7.02**
Word	20	3	15.0%	30.05	59.03	48.02	**+11.02**
Excel	20	6	30.0%	3.33	52.31	47.45	**+4.86**
Cross-office	60	17	28.3%	0.45	37.44	32.57	**+4.87**

I Reproducibility and Experimental Configurations

All-task static diagnostics and held-out-model difficulty validation use fixed analysis sets. Each configuration is run once per task, so the scores carry no repeat-level variance; uncertainty estimates over leakage groups and repeated runs are not part of this release. Valid agent outcomes receive task rewards. Table A14 records the model and harness versions and the solver limits recovered from the saved run configurations.

I.1 Static Evaluation Details

The Sonnet 5 panel uses Claude Code 2.1.251, OpenCode 1.4.14, and OpenHands SDK 1.44.1. Each task contributes one selected completion per harness. Document and spreadsheet results include the pilot runs and reused scores described in Appendix A.

GLM 5.3 uses the same harnesses and fixed Sonnet 5 difficulty labels. Claude Code, OpenCode, and OpenHands each have all 240 task scores, including visual scoring where required. Missing deliverables receive the original judge's failure score, and native spreadsheet penalties are retained. The frozen MSA 2.4.6 run with GLM 5.3 has valid scores for all 240 tasks, including visual scoring where required. Its All, Easy, Medium, Hard, and Expert rows use the same fixed difficulty labels; each modality has 60, 12, 18, 18, and 12 tasks in these respective groups. The independent Sonnet 5–MSA run uses the same frozen harness, task set, and scoring runtime and completes all 240 tasks, scoring 38.08 overall. Its difficulty groups use the same fixed labels. Thirty tasks needed retries after provider failures; the first valid outcome is retained. The per-attempt solver limits are unchanged. Model settings and resource limits are recorded with each configuration.

Table A14: Settings used for the static model–harness comparisons. Token caps are cumulative per solver attempt; time limits are in seconds. Not set means that the benchmark runner imposes no separate cap.

Model	Harness version	Call/iteration limit	Token cap	Time cap (s)	Runs/ task
Sonnet 5	Claude Code 2.1.251	80 turns	Not set	600–3,600	1
	OpenCode 1.4.14	Not set	Not set	600–3,600	1
	OpenHands SDK 1.44.1	80 iterations	Not set	600–3,600	1
	MSA 2.4.6	80 API calls	2,000,000	600–3,600	1
GLM 5.3	Claude Code 2.1.251	80 turns	Not set	600–3,600	1
	OpenCode 1.4.14	Not set	Not set	600–3,600	1
	OpenHands SDK 1.44.1	80 iterations	Not set	600–3,600	1
	MSA 2.4.6	80 API calls	2,000,000	600–3,600	1
GPT6 Astra	MSA 2.4.6	80 API calls	2,000,000	600–3,600	1
Kimi K3	MSA 2.4.6	80 API calls	2,000,000	600–3,600	1
Qwen	MSA 2.4.6	80 API calls	2,000,000	600–3,600	1

Table A14 reports the settings of the saved static runs; the last three rows cover the Expert-tier comparison in Figure 4, which runs on the same MSA settings. Claude Code uses an 80-turn limit, OpenHands allows 80 iterations, and MSA allows 80 model calls including request retries; these counters measure different events. The runner sets no separate OpenCode step cap and no cumulative token cap for Claude Code, OpenCode, or OpenHands. MSA has a two-million-token cap per attempt. Client and provider limits still apply where the table says Not set. All configurations use the solver timeout in each task's `task.toml`, ranging from 600 to 3,600 seconds, separately from setup and judging. Each task contributes one retained outcome; infrastructure retries do not count as independent repeats.

I.2 Evolution Evaluation Details

The initial harness is a frozen mini-SWE-agent (MSA) 2.4.6 adapted to office tasks, with GLM 5.3 as the solver and updater. ACE updates the context playbook; AHE may revise the MSA prompt, tools, and agent loop. Both start from the same harness snapshot. The task manifest and source-overlap audit define the feedback, selection, and final partitions (Appendix C).

ACE and AHE each have a cumulative updater budget of 100 million tokens, covering all analysis roles and retries. Both permit at most three candidate submissions, four selection queries including H_0, and 24 hours of search. Token consumption accumulates across resumed runs against the same limit. Search and inference costs are reported separately. Final inference uses the same per-task

budget and scoring rules for all three rows, and the paired baseline uses the same final task IDs, task versions, and evaluation settings.

For reporting, we evaluate the highest-scoring evolved revision on selection without replacing it with H_0 when it scores lower. This reporting rule was added after inspecting the AHE selection result, before testing that revision on the final tasks. The original gate decision and its reused H_0 test records are retained separately. No additional selection query or harness update is made for this evaluation; the revised harness is frozen before final testing. Only one AHE revision completed selection evaluation. A subsequent feedback pass reached 58 of 90 tasks before the 24-hour search limit and did not produce another evaluated revision.

J Conclusion and Limitations

J.1 Conclusion

EvoCowork evaluates whether changes learned from office-task feedback help agents complete new work, including workflows that require several files to agree. Its 240 tasks combine artifact-level scoring with separate feedback, selection, and final sets, so improvements can be tested beyond the tasks used to develop a harness. In our GLM 5.3 experiments with a shared frozen MSA seed, ACE and AHE improve the in-domain average, while cross-office gains are near zero for ACE and negative for AHE. These results show why office-agent self-improvement needs a separate test of cross-office transfer: a higher overall score can hide little improvement or a decline on those workflows. A blinded audit of 60 outputs per modality places the judges within 3.8 to 9.6 score points of the human consensus, with rater agreement from 0.969 to 0.994, and the modality ordering differs from the one the deterministic checks produce. The benchmark provides a common setting for comparing harness revisions and examining whether their gains extend to new tasks and justify their cost.

J.2 Limitations

The benchmark measures offline office deliverables in isolated task environments, so its scores describe artifact quality under replayable conditions rather than live application state, unrestricted web access, or deployment with real business consequences. Source adaptation improves replayability and can change the interaction required by the upstream task, so cross-suite comparisons use the adaptation records in Appendix B.

Empirical difficulty is conditional on the calibration panel and current judges. Rank bins impose an ordering on that panel, and the document Hard–Expert boundary falls inside a block of tied zero scores, resolved by tie-breaking. The GLM 5.3 configurations in Table 3 test the fixed labels on a held-out model. Visual scoring may introduce evaluator-specific preferences, while deterministic checks are sensitive to artifact structure and exact formulas. The reported comparisons cover the frozen configurations tested here.