The Art of Building Verifiers for Computer Use Agents

Corby Rosset[1,*], **Pratyusha Sharma**[1], **Andrew Zhao**[1],
Miguel Gonzalez-Fernandez[2], **Ahmed Awadallah**[1]

[1]Microsoft Research [2]Browserbase
{corbyrosset,pratysharma,andrewzhao,ahmed.awadallah}@microsoft.com

Abstract

Verifying the success of computer use agent (CUA) trajectories is a critical challenge: without reliable verification, neither evaluation nor training signal can be trusted. In this paper, we present lessons learned from building a best-in-class verifier for web tasks we call the Universal Verifier. We design the Universal Verifier around four key principles: 1) constructing rubrics with meaningful, non-overlapping criteria to reduce noise; 2) separating process and outcome rewards that yield complementary signals, capturing cases where an agent follows the right steps but gets blocked or succeeds through an unexpected path; 3) distinguishing between controllable and uncontrollable failures scored via a cascading-error-free strategy for finer-grained failure understanding; and 4) a divide-and-conquer context management scheme that attends to all screenshots in a trajectory, improving reliability on longer task horizons. We validate these findings on CUAVerifierBench, a new set of CUA trajectories with both process and outcome human labels, showing that our Universal Verifier agrees with humans as often as humans agree with each other. We report a reduction in false positive rates to near zero compared to baselines like WebVoyager ($\geq$ 45%) and WebJudge ($\geq$ 22%). We emphasize that these gains stem from the cumulative effect of the design choices above. We also find that an auto-research agent achieves 70% of expert quality in 5% of the time, but fails to discover all strategies required to replicate the Universal Verifier. We open-source our Universal Verifier system along with CUAVerifierBench[1].

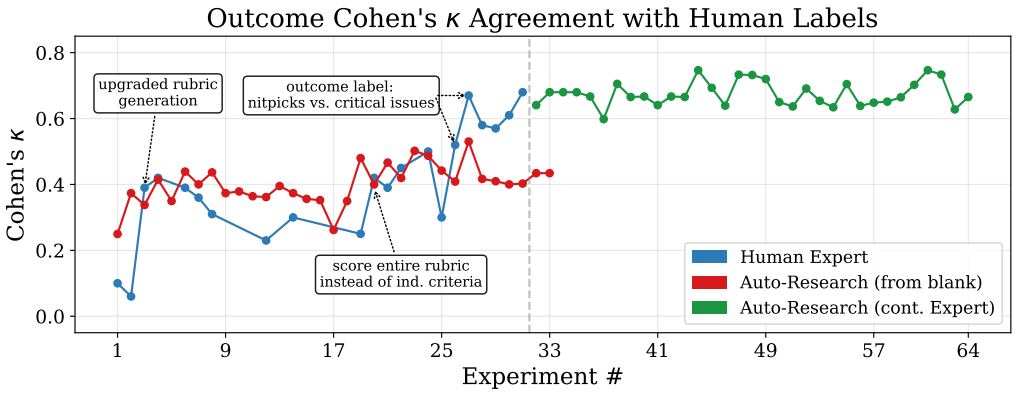

Figure 1: We compare whether an auto-research system can design a CUA trajectory verifier as well as the expert human-designed Universal Verifier, as measured in agreement with human labels. The human expert iterated over 32 experiments across three weeks; the auto-research agent completed the same in roughly one day. Qualitatively, auto-research edits tended to be conservative and incremental, missing the design intuition behind the human's highest-impact structural decisions (tagged)

[*]Corresponding author.
[1]Code for the Universal Verifier is at github/fara; data on huggingface/CUAVerifierBench

1 Introduction

The ability of AI agents to operate computers autonomously—browsing the web, filling forms, navigating interfaces—has advanced rapidly Zhou et al. (2024); He et al. (2024a); Zheng et al. (2024); Koh et al. (2024); Xie et al. (2024); OpenAI (2025); Agashe et al. (2025); Awadallah et al. (2025a); Gupta et al. (2026). Yet progress in training and evaluating these systems is bottlenecked by a deceptively difficult question: *did the agent actually succeed?* Unlike text generation tasks where outputs can be compared directly, computer use trajectories are long, visually rich, and ambiguous, making human annotation both challenging and expensive. The notion of success itself is nuanced: a task may be partially completed; success may be achieved through unexpected paths; and failures may be subtle, appearing only transiently in a screenshot buried deep in a multi-step interaction. Building a verifier that reliably answers this question is far from straightforward—and the consequences of getting it wrong compound, corrupting both benchmarks and training data.

In this paper, we document the lessons learned from building a verifier for computer use agents, structured as a set of actionable design principles. Our approach rests on four core ideas. First, a good verifier requires well-designed rubrics with specific, non-overlapping criteria that enable consistent scoring across diverse tasks. Second, it must report both process and outcome rewards—these provide complementary signals that differ primarily in whether the environment prevented success despite correct agent behavior, or allowed success via an unexpected but valid path. Third, it must distinguish controllable failures from uncontrollable ones and score trajectories with a cascading-error-free rubric, so that a single early obstacle does not unfairly penalize all downstream steps. Fourth, it must attend effectively to all screenshot evidence in a trajectory, not just the most recent frames; longer tasks contain critical state changes that are systematically missed when context is truncated.

To support rigorous evaluation of these principles, we release **CUAVerifierBench**, a benchmark of human-labeled CUA trajectories. To our knowledge, CUAVerifierBench is the first benchmark designed specifically to measure verifier quality for both process and outcome rewards, enabling the community to compare verifier alignment with human judgment in a standardized way. We show that our verifier—which we call the *Universal Verifier*—substantially improves alignment with human labels over existing WebJudge, WebVoyager as measured by Cohen's κ, while reducing false positive rates from 30%+ to 1-8%.

Crucially, building a high-quality verifier is not a one-shot problem but an iterative development process, and this process is only possible when grounded in a reliable evaluation procedure. CUAVerifierBench serves exactly this role: each candidate verifier design can be scored against human judgments using Cohen's κ, providing a clear and immediate signal for what works and what does not. Figure 1 traces this iterative journey over 96 experiments. The expert-designed verifier begins with near-zero agreement and steadily improves through principled experimentation, reaching $\kappa \approx 0.7$ by experiment 32 as the four design principles are incrementally discovered and integrated.

We also explored whether an automated research agent could replicate this process. Starting from a blank slate, the auto-research-designed verifier follows a similar upward trend but consistently underperforms, with κ plateauing around 0.55—roughly 70% of expert-level quality. Qualitatively, the auto-research agent's edits tended to be conservative and incremental, struggling to encode the kind of evaluative judgment behind the large structural changes that drove the expert-designed verifier's step-function improvements. However, when initialized from the expert's best verifier configuration, the auto-research agent surpasses the expert-designed peak, suggesting that human expertise and automated optimization play complementary roles: the former is essential for discovering core design principles, while the latter excels at the fine-grained tuning that extracts remaining performance.

In summary, our contributions are as follows: (1) We identify and validate four design principles for building reliable CUA verifiers, showing that their cumulative effect yields a verifier that agrees with humans as often as humans agree with each other. (2) We release **CUAVerifierBench**, the first benchmark specifically designed to evaluate verifier quality for computer use agents, providing the community with a standardized way to measure verifier alignment with human judgment.

Verifier	LLM	Rubric	Screenshots	Action hist.	Final ans.
WebJudge (OM2W)	o4-mini	✗ Not used	✓ Top-k most relevant (scored 1–5, kept if $\geq$ threshold; capped at 5)	✓ Full	✗
WebVoyager GPT eval	gpt-4o	✗ Not used	✓ All screenshots (last N if over limit; default N=30)	✗	✓
Universal Verifier (Ours)	gpt-5.2	✓ Per-task success criteria	✓ Top-k most relevant *per criterion*	✓ Full	✓

Table 1: Comparison of different computer use trajectory verifiers' characteristics

2 Background and Related Work

CUA Verifiers Several systems have been proposed for automatically evaluating CUA, differing primarily in what inputs they consume and whether they rely on prompted LLMs or trained models. **WebVoyager** (He et al., 2024b) uses a GPT-4V-based evaluator that receives all trajectory screenshots (but not all action history) alongside the agent's stated final answer to produce a binary outcome judgment. Validated against human annotations on 300 tasks, the GPT-4V variant achieves 85.3% agreement (κ=0.70), matching human inter-annotator agreement. **WebJudge** (Xue et al., 2025) addresses two known failure modes of this approach: reliance on the agent's potentially hallucinated final answer, and token overload from passing all screenshots unfiltered. It employs a three-step pipeline that first extracts key points from the task description, scores each screenshot for relevance, and judges success using only the top-k selected screenshots and the full action history. Under the same evaluation setting, WebJudge (o4-mini) achieves 85.7% human-agreement compared to 78.7% for WebVoyager.

Shifting from outcome prediction to failure diagnosis, **AgentRx** (Barke et al., 2026) identifies the *critical failure step* and assigns it a root cause from a nine-category taxonomy.

AgentRewardBench (Lù et al., 2025) provides 1,302 expert-annotated trajectories across five benchmarks (WebArena (Zhou et al., 2023), VisualWebArena (Koh et al., 2024), Assistant-Bench (Yoran et al., 2024), WorkArena (Drouin et al., 2024), WorkArena++ (Boisvert et al., 2024)) and four agent LLMs (GPT-4o (Hurst et al., 2024), Claude 3.7 Sonnet (Anthropic, 2025), Llama-3.3-70B (Dubey et al., 2024), and Qwen2.5-VL (Bai et al., 2025b)). They introduce a Simplified Judge that, in a single LLM completion, predicts three binary labels—task success, side effects, and repetition cycles. Their key finding is that no LLM-based judge exceeds 70% precision: including NNetNav (Murty et al., 2025) and AER (Pan et al., 2024). Human inter-annotator agreement was 89.3%.

Rubrics and Process Rewards Several works debated whether process or outcome rewards are more effective for scenarios such as solving math problems (Lightman et al., 2023; Uesato et al., 2022); Wang et al. (2024) trains their own process reward model. Zhang et al. (2025c) distill lessons for building process verifiers for math. Others extend to agentic RAG domains (Zhang et al., 2025a), or using rubrics as reward signal in RL training He et al. (2026). We refer the reader to additional surveys (Zheng et al., 2025; Stuhlmüller & Byun, 2022). Additional related works are in Appendix A.

3 What is True of Good Verifiers?

We distill principles we believe are critical to the construction of a reliable verifier based on our extensive hands-on experience with CUA trajectory logs.

3.1 Good Rubrics have Specific and Non-Overlapping Criteria

The root of the pipeline is rubric generation: flawed rubrics produce errors that cascade through the pipeline and cannot be easily corrected downstream. Anecdotally, Figure 1

shows that the good rubric design *alone* accounted for roughly half of the Cohen's κ gains[2]. Through iterating, we identified four systematic failure modes and corresponding fixes:

1. **Phantom criteria.** LLM-generated rubrics frequently introduce requirements never stated in the task (e.g., in Appendix B.2, Table 6), inflating the denominator and over-penalizing agents that completed the actual task.
2. **Cascading errors.** When rubric criteria are not logically independent, a single upstream error propagates into downstream criteria, multiplying the point penalty.
3. **Separate Generation and Scoring** Generating the rubric and scoring it in a single LLM call leads the model to create criteria tailored to the agent's behavior. We separate rubric generation (from the task alone, without seeing the trajectory) from scoring.
4. **Hallucination detection.** We score the whole rubric in two passes—with and without evidence from the relevant screenshots—to surface discrepancies.
5. **Conditional Criteria**: Some criteria may not apply depending on reality (e.g., *"buy organic blueberries, or if unavailable, buy non-organic"*). Hence, at rubric-generation time, we mark some criteria as "conditional" to be updated once the task is attempted. Conditions that are not met are excluded, ensuring that mutually exclusive criteria do not interfere. See Appendix B.2, Table 7 for details and examples.

The remaining sub-sections discuss the scoring of rubrics. Sometimes, the rubric is modified for e.g. updating conditional criteria, or adding new criteria for unsolicited side-effects.

3.2 Separate Process and Outcome Rewards

In computer use settings, the environment plays an out-sized role in the success of a task, especially if an agent is blocked or can't access necessary resources. Hence, a central design principle of our verification framework is the separation of *how well the agent executed* in the context of the environment from *whether the user's goal was achieved*. These two questions have fundamentally different answers in many real-world scenarios, and conflating them leads to reward signals that are either too lenient (crediting agents for apparent effort when the user is left empty-handed) or too harsh (penalizing agents for factors outside their control). We formalize this separation through two two independent signals per trajectory: a **process reward** (a fine-grained rubric whose score reflects execution across sub-goals) and an **outcome reward** (a binary success/failure judgment on whether the goal was achieved).

Process Label (Rubric Score): This is a scored rubric of criteria, each of which is weighted by a maximum number of earnable points. It is reported as a normalized score from 0.0 to 1.0 reflecting how well the agent executed each sub-goal of the task. It is computed as:

$$r_{\text{proc}} = \frac{\sum_{i \in \mathcal{A}} \text{earned_points}_i}{\sum_{i \in \mathcal{A}} \text{max_points}_i} \tag{1}$$

where $\mathcal{A}$ is the set of *applicable* rubric criteria—those whose conditions are met (for conditional criteria) or that are unconditional. The process label evaluates the quality of the agent's execution at each step, independent of whether those steps ultimately produced a successful outcome. While it is technically a scalar score, the rubric also contains specific justifications as to why points were earned or lost based on evidence from the full action history and screenshots. An agent that, for example, navigated to the correct product, and but was blocked by a login wall before it could add-to-cart would full process credit, even though the user's goal was not achieved. Example rubrics can be seen in Figures 2, and 3.

Outcome Label (Binary Success/Failure) The outcome label is a binary yes/no judgment answering: **would a reasonable user consider the task done?** This is evaluated from the perspective of a user who issued the task and is examining the end state. This is intrinsically challenging, because users may have different notions of success under ambiguity (e.g. is it acceptable to omit NeurIPS's secondary venue in Mexico City when asked *"where is NeurIPS 2025 being hosted?"*) and different preferences as for what constraints are strict vs flexible (e.g. is ok to book a table using opentable.com when the user asked to use resy.com?).

[2]The reader can view the evolution of rubrics on WebTailBench tasks: via this visualizer

In order to make progress, we assume that the outcome label should focus on *primary intent* – if the primary intent is to book a table, then the user would be flexible on which platform it is booked unless otherwise stated. We also believe most users are forgiving of nitpicks like rounding $5.95 to $6, etc. However, we assume users would *not* be forgiving of unsolicited side-effects e.g. buying a warranty when they only wanted to buy the product itself, or hallucinations like those described in Table 9. We summarize the process and outcome rewards for computer use scenarios in Table 8 – notice they only disagree in the second row.

3.3 Discern Controllable vs. Uncontrollable Factors

Since the main difference between a trajectory being a process success but an outcome failure involves the environment, we explicitly define which of these aspects are controllable vs. uncontrollable from the perspective of the agent. Each rubric's criteria description fields attempt to anticipate these factors and give guidance on how to award partial credit.

Uncontrollable factors: Conditions beyond the agent's control; *not* penalized in *process*.

- **Platform/infrastructure issues**: CAPTCHA, login walls without credentials, etc.
- **Entity non-existence**: product discontinued, business closed, service not available.
- **Availability constraints**: out of stock, no reservations on requested date, sold out.
- **Search result limitations**: no results matching all specified criteria.

Controllable factors: Avoidable mistakes the agent *should* be penalized for in *process*.

- **Intent Mis-match**: Choosing an entirely wrong product, location, person, service, etc.
- **Reasoning Errors**: Incorrect reasoning about the task e.g. Figure 4.
- **Hallucinations**: claiming success without evidence, fabricating information.
- **Insufficient effort**: giving up after a single failed attempt.
- **Execution errors**: not using available filters, skipping required steps.

3.4 Effective Context Management of Screenshot Evidence

Our main contribution is a verifier designed to combat hallucinations[3] through better management of visual screenshot evidence. Both WebVoyager (He et al., 2024b) and WebJudge (Xue et al., 2025) assess a large amount of screenshots in one LLM context window – WebVoyager includes all screenshots, whereas WebJudge ranks the top $\approx 30 - 50$. Other verifiers only analyze the last ones (Pan et al., 2024). Too many screenshots over-exerts the LLM by forcing it to solve a needle-in-a-haystack problem, which scales poorly with longer trajectories, whereas restricting to the last few risks missing task-relevant evidence. To address these problems, our design scores each screenshot against every rubric criterion to produce a relevance matrix, grouping the top-k most relevant *per criterion* to send for further analysis, which is both more scalable to longer trajectories and more focused. We elaborate on our screenshot-scoring design in Appendix B.3.1 with an example in Figure 6.

3.5 Unsolicited Side-Effects

Extraneous actions with material side effects—such as adding unrequested items to a cart (e.g. see Figure 7) or enrolling in unrequested services—constitute often cannot be anticipated before the task is attempted because rubrics are not designed to enumerate all the ways a task can go wrong. To catch such cases, a dedicated pass over a trajectory is needed. While unsolicited side-effects almost always result in outcome failure, they only partially penalize the process score, weighted by how serious the side-effect is.

4 Universal Verifier System

We model a computer use task as a tuple $(g, \mathcal{E})$, where g is a natural language goal (e.g., *"book the cheapest available flight from Seattle to Boston on June 3rd"*) and $\mathcal{E}$ is a computer environment

[3]We define the anatomy of hallucinations Section B.5, Table 9, and give an example in Figure 5

Algorithm 1 Universal Verifier

Require: agent trajectory τ, observations $\{s_0, \ldots, s_T\}$, user goal g
Ensure: Process score r_{proc}, Outcome score r_{out}, diagnostic report d

1: **Generate Rubric** $\mathcal{C} = \{c_1, \ldots, c_N\}$ of N disjoint, meaningful criteria from g. See B.2
2: **Multimodal Relevance Scoring.** Score each screenshot against every criterion to produce relevance matrix $\mathbf{R} \in \mathbb{R}^{(T+1)\times N}$. See Appendix B.3.1 for more details.
3: **Top-k Grouping.** For each c_j, select the k most relevant $\mathcal{S}_j \subseteq \{s_0, \ldots, s_T\}$, $|\mathcal{S}_j| \leq k$.
4: **Evidence Analysis.** For each pair (c_j, s_i) with $s_i \in \mathcal{S}_j$, extract visual evidence e_{ij}.
5: **Conditional Disambiguation.** Resolve conflicts among conditional criteria using $\{e_{ij}\}$.
6: **Reality Check.** Reconcile rubric assumptions against screenshot evidence; produce interpretive reality notes and action-only score $r_{\text{proc_action_only}}$.
7: **Multimodal Rescoring.** Rescore $\mathcal{C}$ holistically using screenshot evidence (which takes precedence over agent claims) following Tables 8 and 9.
8: **Side-Effect Detection.** Detect and include unsolicited agent actions with material side effects not already penalized by $\mathcal{C}$, **return** procedural score r_{proc}. (see example Figure 7)
9: **Outcome Verification.** run and **return** outcome score r_{out}.
10: **Failure Diagnosis.** Identify and localize all failures points from Table 12 and **return** d.

with an observable graphical interface. An agent interacts with $\mathcal{E}$ over T discrete timesteps, producing a trajectory $\tau = (s_0, a_1, s_1, a_2, \ldots, a_T, s_T)$, where $s_t \in \mathcal{S}$ is a screenshot observation at time t and $a_t \in \mathcal{A}$ is an action (e.g., click, type, scroll). The length T varies across tasks from a handful of steps for form-filling to hundreds of steps for multi-stage workflows.

We define a **verifier** as a function $V : (g, \tau) \to \mathcal{R}$ that maps a goal and trajectory to a structured scoring response $r \in \mathcal{R}$. In the simplest case $\mathcal{R} = \{0, 1\}$ (binary success), but we argue and our design reflects—that $\mathcal{R}$ should be richer: a tuple $(r_{\text{proc}}, r_{\text{out}}, d)$ comprising a process score $r_{\text{proc}} \in [0, 1]$, an outcome score $r_{\text{out}} \in \{0, 1\}$, and a diagnostic report d that classifies and localizes failures within τ. The process score captures the quality of the agent's execution, while the outcome score reflects whether the goal g was ultimately satisfied.

The central challenge is that V must operate over the full observation sequence $\{s_0, \ldots, s_T\}$, which can be long, visually dense, and contain critical state changes at arbitrary timesteps. We define verifier quality as agreement with a human oracle $V^* : (g, \tau) \to \mathcal{R}$, measured by precision, recall, and Cohen's κ over a labeled set of trajectories (Artstein & Poesio, 2008). A verifier that inspects only s_T or a fixed subset $\{s_{t_1}, \ldots, s_{t_k}\} \subset \tau$ is a strict approximation of V^* and, as we show empirically, systematically underperforms on trajectories where T is large. Reliable verification therefore requires attending to all $T + 1$ observations.

The **Universal Verifier** (UV) we create incorporates the principles from Section 3 and operates in three phases: rubric creation, *multimodal scoring* incorporating screenshot evidence to ascertain r_{proc}, and produce a final outcome judgment r_{out}, and error diagnosis d as shown in Algorithm 1. The key design invariant is that no relevant screenshot evidence can go undetected in the pipeline, specifically to not miss any hallucinations. To reduce variance, Steps 7–9 in Algorithm 1 can be run as multiple parallel instances, with process score determined by median of rubric scores, and outcome by majority vote.

Finally, we conduct an error analysis on τ to categorize failure modes and identify the step t at which each failure occurred in a trajectory. We hand-crafted an error taxonomy with 7 categories and 24 subcodes as shown in Table 12, covering categories such as intent missmatches, hallucinations, critical point violations, etc.

5 Experiments

We treat the Universal Verifier as an annotator like any other human, and compute inter-annotator agreements throughout our studies: (1) agreement with human trajectory labels on two independently annotated datasets, (2) agreement between native benchmark verifiers and UV at scale, and (3) an auto-research study exploring whether an AI agent can replace or augment human expertise in verifier design. We describe each experimental setup below.

CUAVerifierBench: Human-Labeled Datasets

Since the UV's innovation of verifying both process and outcome labels is novel in the computer use domain, no existing benchmarks provide both labels.

Internal split. We sampled 140 trajectories from WebTailBench using Fara-7B(Awadallah et al., 2025b). In-house expert annotators labeled each trajectory for both process success and outcome success following the guidelines in §3. This dataset is used for tuning the workflow and prompts of the verification pipeline, and the auto-research experiments (§6).

Browserbase OM2W split. To prevent overfitting on the Internal dataset, we contracted external annotators managed by **Browserbase**[4] to label an additional 106 trajectories sampled from Fara-7B (Awadallah et al., 2025b) on Online-Mind2Web for both process and outcome success, with 2× annotator overlap per trajectory. Annotators were first calibrated on 10 practice trajectories with gold annotations. They then judged each evaluation trajectory in a two-stage process: 1) **UV-blind stage:** Annotators saw only the input task, the un-scored rubric criteria, and the agent's trajectory. They independently judged outcome and process success and provided a continuous rubric score per trajectory. 2) **UV-informed stage:** Annotators were shown the UV's outcome verdict and rubric scores, and asked whether they *agreed/disagreed* with the UV's outcome and process.

For task-level aggregation, outcome labels are computed as the majority vote of the annotators' binary judgments, and process labels are the median of the annotators' continuous rubric scores, then binarized at a ≥ 0.8 threshold. Ties are broken by a third. We report agreement metrics from both stages: *UV-blind* agreement measures how often human judgments independently align with the UV, while *UV-informed* agreement measures how often humans endorse the UV's verdict after reviewing its reasoning. We further measure inter-annotator agreement, and how often their labels flipped once seeing the UV's output.

Agreement on Canonical Benchmarks' Verifiers The human-labeled datasets above are small by design (expert annotation is expensive). To assess verifier behavior at scale, we re-score several agent trajectories across several canonical benchmarks like with Universal Verifier and compute agreement between that benchmark's "native" verifier and UV. We select three benchmarks – WebVoyager, Online-Mind2Web (OM2W), and WebTailBench – and two agent models – Fara-7B and GPT-5 as a Set-of-Marks Agent (Yang et al., 2023).

Auto-Research Study The Universal Verifier comprises approximately 3,000 lines of code and 2,000 lines of prompts—including rubric generation templates, scoring instructions, outcome verification logic, and error classification rules—all designed iteratively by a human expert (the first author). To investigate whether an AI agent can replicate or augment this human expertise, we designed an auto-research system using Claude Code v2.1.87 with Claude Opus 4.6 (1M context) on a Claude Max subscription. The system is given the same principles from Section 3, and reuses the same experimental infrastructure as the human expert (running the UV on the internal set, computing agreement metrics, and committing prompt changes to version control). We evaluate two settings:

- **From-blank prompts:** All ∼2,000 lines of prompts are replaced with `// TODO` placeholders, leaving only the code scaffold. The agent is given high-level design principles but no access to prior prompt versions, previous commits, or other branches. A separate compliance agent audits each iteration to prevent memorization of test examples into prompts. The optimization rule is: *maximize Cohen's κ without increasing FPR*; any FPR-increasing change is automatically rolled back.
- **Continuing expert work:** The agent starts from the human expert's best prompts and continues with the same optimization objective.

[4] https://www.browserbase.com/

	Internal Dataset (n=140)					Browserbase OM2W (n=106)				
	WebVoy.		WebJudge		UV	WebVoy.		WebJudge		UV
	GPT-4o	GPT-5.2	o4-mini	GPT-5.2	GPT-5.2	GPT-4o	GPT-5.2	o4-mini	GPT-5.2	GPT-5.2
Agreement with outcome human labels										
Accuracy (↑)	0.67	0.70	0.72	0.64	0.81	0.48	0.74	0.64	0.74	0.88
F1 (↑)	0.73	0.69	0.74	0.58	0.81	0.35	0.50	0.44	0.46	0.65
Cohen's κ (↑)	0.31	0.43	0.44	0.33	0.64	0.13	0.36	0.26	0.31	0.58
FNR (↓)	0.24	0.44	0.33	0.57	0.32	0.12	0.18	0.12	0.29	0.31
FPR (↓)	0.45	0.10	0.22	0.07	0.01	0.60	0.28	0.40	0.26	0.08
Agreement with process human labels										
Accuracy (↑)	0.62	0.64	0.66	0.61	0.81	0.55	0.75	0.68	0.73	0.78
F1 (↑)	0.70	0.65	0.70	0.57	0.86	0.47	0.56	0.53	0.49	0.57
Cohen's κ (↑)	0.17	0.34	0.32	0.30	0.59	0.22	0.40	0.34	0.32	0.43
FNR (↓)	0.31	0.49	0.40	0.59	0.24	0.05	0.23	0.12	0.36	0.29
FPR (↓)	0.52	0.10	0.25	0.04	0.04	0.56	0.26	0.38	0.25	0.20

Table 2: Agreement between three verifiers and humans in CUAVerifierBench. Upgrading external verifier to GPT-5.2 results in only modest improvement, confirming the UV's advantage is architectural.

6 Results

Agreement with Human Labels: UV vs. Existing Verifiers: In Table 2 we compare UV against two prominent existing trajectory judges—WebVoyager (He et al., 2024b) and WebJudge (Xue et al., 2025)—on CUAVerifierBench. The UV substantially outperforms both baselines across nearly every metric on both datasets. On outcome labels, the UV achieves a Cohen's κ of 0.64 (internal) and 0.58 (Browserbase), compared to 0.44/0.26 for WebJudge and 0.31/0.13 for WebVoyager. Strikingly, the UV achieves an FPR near zero (0.01 internal, 0.08 Browserbase) on outcome labels, meaning it almost never credits a trajectory with success when a human annotator would mark it as a failure. A version of this table with standard deviation error bars computed from three independent runs is included in Table 17.

To test whether the UV's advantage stems from simply from using a stronger backbone model, we report four additional columns in Table 2, where we upgrade WebVoyager's GPT-4o and WebJudge's o4-mini to GPT-5.2. While this does reduce FPR substantially (e.g., WebVoyager outcome FPR drops from 0.45 to 0.10 on Internal), it also dramatically increases FNR (0.24→0.44), and overall κ improves only modestly. **We conclude UV's advantage stems from its screenshot scoring design, not merely from using a stronger model**

Browserbase Annotations: Using the two-stage annotation protocol described in §5, we measure how agreement changes when annotators are shown the UV's reasoning. The UV-informed stage substantially improves agreement: outcome Cohen's κ rises from 0.39 to 0.63, and outcome FNR drops from 0.62 to 0.35, while FPR remains near zero (0.04). On process labels, FNR drops sharply from 0.32 to 0.09. Only 16.6% of annotator outcome judgements flipped after seeing the UV's reasoning, nearly all moving from success to failure after the UV identified a failure they initially missed.

We also plot a scatter plot of the rubric score the human annotators assigned to the trajectories vs what the UV assigned in Figure 11. See in Appendix C.2 including Table 15 for full results.

Inter-annotator agreement: the Browserbase split contains at least two annotations per trajectory. The UV's outcome κ with human labels (0.58, Table 2) and process κ (0.43) fall within the corresponding inter-annotator ranges (0.53–0.57 and 0.36–0.45, respectively; Table 16), **indicating that the UV agrees with humans about as well as humans agree with each other on both dimensions** (We report more details in Section C.2).

Ablations: Varying Rubric Generator and Scorer: We conduct two additional ablations of the Universal Verifier, reported in full in Appendix C.1. In Table 13 we vary the backbone

	WebVoyager		OM2W		WebTailBench	
	Fara-7B	GPT-5	Fara-7B	GPT-5	Fara-7B	GPT-5
N (tasks scored)	594	593	298	276	599	597
Unterminated (%)	4.2	3.4	5.0	7.2	17.0	7.7
Success rate (%)						
Native verifier	74.6	90.6	32.2	62.0	39.6	62.5
UV Process	49.0	79.4	25.8	64.9	41.9	66.9
UV Outcome	37.9	71.0	15.8	48.6	22.4	41.0
Native vs. UV Process†						
FNR (↓)	0.06	0.04	0.26	0.27	0.30	0.23
FPR (↓)	0.56	0.68	0.18	0.42	0.20	0.37
Accuracy (↑)	0.69	0.83	0.80	0.67	0.76	0.72
F1 (↑)	0.75	0.90	0.66	0.74	0.70	0.78
Cohen's κ (↑)	0.38	0.36	**0.52**	0.30	**0.50**	**0.40**
Native vs. UV Outcome						
FNR (↓)	0.01	0.02	0.17	0.24	0.14	0.17
FPR (↓)	0.60	0.72	0.23	0.49	0.25	0.49
Accuracy (↑)	0.63	0.78	0.78	0.63	0.77	0.64
F1 (↑)	0.68	0.86	0.55	0.67	0.64	0.65
Cohen's κ (↑)	0.33	0.33	0.42	0.27	**0.49**	0.31

Table 3: Agreement between native benchmark verifiers and the Universal Verifier (UV) across three benchmarks and two agent models. The UV is treated as the reference label.

LLMs of the UV end-to-end (each model generates and scores its own rubric), finding that GPT-5.2 achieves the lowest FPR while GPT-5 offers the best balanced agreement. In Table 14 we again vary the backbone LLM, but isolate the scoring component by fixing the rubric (generated by GPT-5.2), showing that GPT-5.2 is the most conservative scorer while GPT-5.1 achieves the highest overall κ.

Agreement Between UV and Native Benchmark Verifiers We measure agreement between the UV and the *native verifiers* shipped with each of three benchmarks: WebVoyager, Online-Mind2Web (OM2W), and WebTailBench. Table 3 shows that the native verifiers disagree substantially with the UV labels: false positive rates w.r.t UV outcome labels are consistently above 20%, with WebVoyager (GPT-4o) having the highest FPR and lowest Cohen's κ. Histograms of error taxonomies for these are shown in Figures 8, 9, and 10.

Training: Filtering SFT Data We wanted to ascertain if using the Universal Verifier as a SFT training data filter improves model performance. In Table 4 we show that, under fixed training trajectory quantity, those filtered by the Universal Verifier lead to the best downstream model, especially under data-limited scenario. We were somewhat surprised to observe that the process-filter outperforms outcome-filtered; we believe this is due to the process success threshold of 80% allowing some imperfections in the trajectory, which is ultimately beneficial to the model (Tajwar et al., 2024). These are not exactly compute-equalized in the sense that some trajectories

Experiment	Filtered by	Online Mind2Web	WebVoyager
3k traj.	Baseline (old verifier)	0.20	0.41
	UV Process	0.28	0.45
	UV Outcome	0.24	0.44
9k traj.	Baseline (old verifier)	0.25	0.46
	UV Process	0.29	0.52
	UV Outcome	0.29	0.49

Table 4: Results across data filtering variants at 3k and 9k training trajectories.

In this training regime, we sampled trajectories from the insta-150k-v3 dataset (Trabucco et al., 2025), after being re-solved by the FaraGen pipeline (Awadallah et al., 2025a). We

	SoM Agents								*Computer Use Models*					
	SoM GPT-5		SoM o3		SoM GPT-4o		GLM-4.1V 9B-Thinking		OAI Comp. Use-Prev		UI-TARS 1.5-7B		Fara 7B	
WebTailBench	pr	out	pr	out	pr	out	pr	out	pr	out	pr	out	pr	out
Shopping	64.8	38.9	60.0	36.4	37.5	17.9	27.8	9.3	44.6	16.1	53.6	30.4	58.9	39.3
Flights	51.1	27.7	23.4	8.5	3.9	2.0	6.1	0.0	17.6	7.8	5.9	2.0	44.0	30.0
Hotels	84.6	23.1	54.9	15.7	32.7	9.6	21.2	5.8	32.7	13.5	28.8	5.8	51.9	32.7
Restaurants	68.6	41.2	47.1	27.5	32.7	13.5	34.6	17.3	34.6	17.3	11.5	3.8	35.3	13.7
Activities	71.2	43.8	55.1	37.2	27.5	15.0	32.9	13.9	40.0	22.5	8.8	5.0	36.0	16.0
Ticketing	56.4	21.8	37.5	16.1	24.6	0.0	24.6	8.8	40.4	14.0	14.0	1.8	33.9	7.1
Real-Estate	58.7	43.5	19.1	17.0	8.3	4.2	21.3	12.8	13.3	6.7	6.2	0.0	32.6	15.2
Jobs/Careers	61.7	46.8	42.9	30.6	26.0	18.0	28.0	14.0	36.0	20.0	20.0	12.0	30.6	20.4
Shopping List (2 items)	72.5	52.9	48.0	28.0	9.8	5.9	16.3	8.2	35.3	25.5	27.5	11.8	55.1	32.7
Comparison Shopping	73.7	59.6	75.4	57.9	35.1	17.5	28.3	9.4	3.5	1.8	12.3	1.8	54.4	28.1
Compositional Tasks	72.0	52.0	63.0	42.6	25.5	14.5	21.8	10.9	20.0	10.9	14.5	9.1	27.8	11.1
Macro Avg.	66.9	41.0	47.9	28.9	24.0	10.7	23.9	10.0	28.9	14.2	18.5	7.6	41.9	22.4
Micro Avg.	67.3	41.2	48.9	29.7	24.5	11.0	24.5	10.2	29.5	14.5	18.2	7.6	41.8	22.2

Table 5: Breakdown of WebTailBench-v1 results for each of its 11 segments. Each model reports **pr** (process scores ≥ 0.8) and **out** (outcome scores). The first 8 segments test a single skill or objective usually on a single website, the remaining three are more difficult multi-step or cross-site tasks.

trained under largely the same settings as Fara-7B, the only difference being we initialized from Qwen-3-VL-Instruct (Bai et al., 2025a); we trained for two epochs with learning rate 5×10^{-6} with an effective batch size of 128 on a single 8xB200 node with 0.1 learning rate warmup and cosine decay. We tried to enforce compute-equivalence by fixing the number of trajectories in each experiment, but we note that process-filtered trajectories

Updated WebTailBench w/ Universal Verifier In Table 5 we re-report WebTailBench (v1), our heldout evaluation set (Awadallah et al., 2025a) under evaluation by the Universal Verifier. We first fix the rubrics for all 609 tasks in the benchmark[5], and then re-score (running step 2 onwards in Algorithm 1) all the trajectories from all the models we cached from the original Fara-7B report in Table 11 of Awadallah et al. (2025a). Note that process scores are typically higher because the rubric credits the model for best-effort attempts even if the environment fails to manifest e.g. available bookings, search results, or is blocked by logins etc. This is especially true of time-sensitive or availability-based tasks like reservations and shopping.

Auto-Research: Can AI Replace Human Experts in Verifier Design? A natural question is whether an AI auto-research agent can replicate—or even improve upon—the process of designing verifiers(Lu et al., 2026; Karpathy, 2026; Tie et al., 2025). Figure 1 shows outcome Cohen's κ progression across experiments for the human expert and both auto-research settings (process κ is in Figure 13), and Figures 14–15 show the corresponding FPR and FNR trajectories. The blank-prompt auto-research agent reached about 70% of the quality of the human expert in only 5% of the time, and when given the best prompts and code the human had, it could still find improvements subject to the constraint of not increasing false positive rate. Table 20 in Appendix D.1 summarizes each `continue-expert` iteration's changes.

7 Conclusion

We present the Universal Verifier, which scores computer-use trajectories at both the process and outcome level, and CUAVerifierBench, a benchmark measuring verifier agreement with gold human labels. The UV matches human–human agreement, outperforms every verifier we tested, and reduces false positives to near zero (vs. $\geq$45% for WebVoyager and $\geq$22% for WebJudge). These gains are architectural rather than model-driven: giving baselines the same backbone LLM yields only modest improvements.

[5]Rubrics for WebtailBench are: https://huggingface.co/datasets/microsoft/WebTailBench

8 Ethics Statement

We disclose that we contracted human annotators via an external firm Browserbase, and they represented to us that those annotators were paid more than minimum wage applicable under local law. We also represent that some annotators gave us express written permission to quote qualitative feedback they gave us about their experience judging the tasks. We do no disclose any personally identifiable information about the judges. We did not give the judges any psychologically harmful, offensive, or adult-natured tasks.

Additionally, we disclose that parts of this work were produced by generative AI, including but not limited to auto-research studies, results, analysis, and code. We performed our best effort to verify the results were not hallucinated.

References

Saaket Agashe, Kyle Wong, Vincent Tu, Jiachen Yang, Ang Li, and Xin Eric Wang. Agent s2: A compositional generalist-specialist framework for computer use agents, 2025. URL https://arxiv.org/abs/2504.00906.

Anthropic. The claude model spec, 2025. URL https://www.anthropic.com/news/claude-3-7-sonnet. Claude 3.7 Sonnet model card available at https://assets.anthropic.com/m/785e231869ea8b3b/original/claude-3-7-sonnet-system-card.pdf.

Ron Artstein and Massimo Poesio. Survey article: Inter-coder agreement for computational linguistics. *International Conference on Computational Logic*, 34(4):555–596, 2008. doi: 10.1162/coli.07-034-R2. URL https://aclanthology.org/J08-4004/.

Ahmed Awadallah, Yash Lara, Raghav Magazine, Hussein Mozannar, Akshay Nambi, Yash Pandya, A. Rajeswaran, Corby Rosset, Alexey Taymanov, Vibhav Vineet, et al. Fara-7B: An efficient agentic model for computer use. *arXiv.org*, 2025a. doi: 10.48550/arXiv.2511.19663. URL https://arxiv.org/abs/2511.19663.

Ahmed Awadallah, Yash Lara, Raghav Magazine, Hussein Mozannar, Akshay Nambi, Yash Pandya, Aravind Rajeswaran, Corby Rosset, Alexey Taymanov, Vibhav Vineet, et al. Fara-7b: An efficient agentic model for computer use, 2025b. URL https://arxiv.org/abs/2511.19663.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Ke-qin Chen, Xiong-Hui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Rongyao Fang, Chang Gao, et al. Qwen3-vl technical report, 2025a. URL https://arxiv.org/abs/2511.21631.

Shuai Bai, Ke-qin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, K. Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-vl technical report, 2025b. URL https://arxiv.org/abs/2502.13923.

Shraddha Barke, Arnav Goyal, Alind Khare, Avaljot Singh, Suman Nath, and Chetan Bansal. Agentrx: Diagnosing ai agent failures from execution trajectories, 2026. URL https://arxiv.org/abs/2602.02475.

Léo Boisvert, Megh Thakkar, Maxime Gasse, Massimo Caccia, Thibault Le Sellier De Chezelles, Quentin Cappart, Nicolas Chapados, Alexandre Lacoste, and Alexandre Drouin. Workarena++: Towards compositional planning and reasoning-based common knowledge work tasks, 2024. URL https://arxiv.org/abs/2407.05291.

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

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, A. Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Tanmay Gupta, Piper Wolters, Zixian Ma, Petr Sushko, Rock Yuren Pang, Diego Llanes, Yue Yang, Taira Anderson, Bo Zheng, Zhongzheng Ren, et al. MolmoWeb: Open visual web agent and open data for the open web. *arXiv.org*, 2026. doi: 10.48550/arXiv.2604.08516. URL https://arxiv.org/abs/2601.10611.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. WebVoyager: Building an end-to-end web agent with large multimodal models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6864–6890. Association for Computational Linguistics, 2024a. doi: 10.18653/v1/2024.acl-long.371. URL https://arxiv.org/abs/2401.13919.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. Webvoyager: Building an end-to-end web agent with large multimodal models, 2024b. URL https://arxiv.org/abs/2401.13919.

Yun He, Wenzhe Li, Hejia Zhang, Songlin Li, Karishma Mandyam, Sopan Khosla, Yuanhao Xiong, Nanshu Wang, Xiaoliang Peng, Beibin Li, et al. Advancedif: Rubric-based benchmarking and reinforcement learning for advancing llm instruction following, 2026. URL https://arxiv.org/abs/2511.10507.

OpenAI Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card, 2024. URL https://arxiv.org/abs/2410.21276.

Andrej Karpathy. autoresearch: AI agents running research on single-GPU nanochat training automatically. https://github.com/karpathy/autoresearch, March 2026. Accessed: 2026-03-29.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. VisualWebArena: Evaluating multimodal agents on realistic visually grounded web tasks. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 881–905. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.50. URL https://arxiv.org/abs/2401.13649.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step, 2023. URL https://arxiv.org/abs/2305.20050.

Chris Lu, Cong Lu, Robert Tjarko Lange, Yutaro Yamada, Shengran Hu, Jakob Foerster, David Ha, and Jeff Clune. Towards end-to-end automation of AI research. *Nature*, 651 (8107):914–919, 2026. doi: 10.1038/s41586-026-10265-5. URL https://www.nature.com/articles/s41586-026-10265-5.

Xing Han Lù, Amirhossein Kazemnejad, Nicholas Meade, Arkil Patel, Dongchan Shin, Alejandra Zambrano, Karolina Stańczak, Peter Shaw, Christopher J. Pal, and Siva Reddy. Agentrewardbench: Evaluating automatic evaluations of web agent trajectories, 2025. URL https://arxiv.org/abs/2504.08942.

Shikhar Murty, Dzmitry Bahdanau, and Christopher D. Manning. Nnetnav: Unsupervised learning of browser agents through environment interaction in the wild, 2025. URL https://arxiv.org/abs/2410.02907.

OpenAI. Computer-using agent. Technical Report 4, 2025. URL https://openai.com/index/computer-using-agent/.

Jiayi Pan, Yichi Zhang, Nicholas Tomlin, Yifei Zhou, Sergey Levine, and Alane Suhr. Autonomous evaluation and refinement of digital agents, 2024. URL https://arxiv.org/abs/2404.06474.

Rulin Shao, Akari Asai, et al. Dr tulu: Reinforcement learning with evolving rubrics for deep research, 2025. URL https://arxiv.org/abs/2511.19399.

Manasi Sharma, Chen Bo Calvin Zhang, et al. Researchrubrics: A benchmark of prompts and rubrics for evaluating deep research agents, 2025. URL https://arxiv.org/abs/2511.07685.

Andreas Stuhlmüller and Jungwon Byun. Supervise process, not outcomes, 2022.

Fahim Tajwar, Anikait Singh, Archit Sharma, Rafael Rafailov, Jeff Schneider, Tengyang Xie, Stefano Ermon, Chelsea Finn, and Aviral Kumar. Preference fine-tuning of llms should leverage suboptimal, on-policy data, 2024. URL https://arxiv.org/abs/2404.14367.

Guiyao Tie, Pan Zhou, and Lichao Sun. A survey of AI scientists, 2025. URL https://arxiv.org/abs/2510.23045.

Brandon Trabucco, Gunnar Sigursson, Robinson Piramuthu, and Ruslan Salakhutdinov. Insta: Towards internet-scale training for agents, 2025. URL https://arxiv.org/abs/2502.06776.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process- and outcome-based feedback, 2022. URL https://arxiv.org/abs/2211.14275.

Peiyi Wang, Lei Li, Zhihong Shao, R. X. Xu, Damai Dai, Yifei Li, Deli Chen, Y.Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.510.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, T. Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems 37*, pp. 52040–52094. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2024. doi: 10.52202/079017-1650. URL https://arxiv.org/abs/2404.07972.

Ran Xu, Tianci Liu, Zihan Dong, et al. Alternating reinforcement learning for rubric-based reward modeling in non-verifiable llm post-training, 2026. URL https://arxiv.org/abs/2602.01511.

Tianci Xue, Weijian Qi, Tianneng Shi, Chan Hee Song, Boyu Gou, Dawn Song, Huan Sun, and Yu Su. An illusion of progress? assessing the current state of web agents, 2025. URL https://arxiv.org/abs/2504.01382.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v, 2023. URL https://arxiv.org/abs/2310.11441.

Ori Yoran, Samuel Joseph Amouyal, Chaitanya Malaviya, Ben Bogin, Ofir Press, and Jonathan Berant. Assistantbench: Can web agents solve realistic and time-consuming tasks?, 2024. URL https://arxiv.org/abs/2407.15711.

Wenlin Zhang, Xiangyang Li, Kuicai Dong, Yichao Wang, Pengyue Jia, Xiaopeng Li, Yingyi Zhang, Derong Xu, Zhaocheng Du, Huifeng Guo, et al. Process vs. outcome reward: Which is better for agentic rag reinforcement learning, 2025a.

Xinchen Zhang, Xiaoying Zhang, Youbin Wu, Yanbin Cao, Renrui Zhang, Ruihang Chu, Ling Yang, and Yujiu Yang. Generative universal verifier as multimodal meta-reasoner, 2025b. URL https://arxiv.org/abs/2510.13804.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning, 2025c.

Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. GPT-4V(ision) is a generalist web agent, if grounded. In *International Conference on Machine Learning*, 2024. doi: 10.48550/arXiv.2401.01614. URL https://arxiv.org/abs/2401.01614.

Congmin Zheng, Jiacheng Zhu, Zhuoying Ou, Yuxiang Chen, Kangning Zhang, Rong Shan, Zeyu Zheng, Mengyue Yang, Jianghao Lin, Yong Yu, et al. A survey of process reward models: From outcome signals to process supervisions for large language models, 2025.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, et al. Webarena: A realistic web environment for building autonomous agents, 2023. URL https://arxiv.org/abs/2307.13854.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, et al. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, 2024. doi: 10.48550/arXiv.2307.13854. URL https://arxiv.org/abs/2307.13854.

A Extended Related Work

Rubrics and generative verification. Several works debated whether process or outcome rewards are more effective for scenarios such as solving math problems (Lightman et al., 2023; Uesato et al., 2022; Wang et al., 2024); Math-Shepherd (Wang et al., 2024) trains its own process reward model, and Zhang et al. (2025c) distill lessons for building process verifiers for math. Others extend verification to agentic RAG domains (Zhang et al., 2025a).

A parallel line of work moves beyond binary or scalar rewards toward *rubrics*—structured, criterion-level checklists—as the verification signal, motivated by domains where a single ground-truth outcome is ill-defined. This mirrors the challenge in CUA evaluation, where success is often partial, multi-faceted, and hard to reduce to one label. He et al. (2026) use rubrics directly as the reward signal during RL training. ResearchRubrics (Sharma et al., 2025) operationalizes this for deep-research agents, pairing prompts with over 2,500 expert-written rubrics that assess factual grounding, reasoning soundness, and clarity; even frontier systems fall below 68% rubric compliance, underscoring how much headroom criterion-level evaluation exposes relative to coarse outcome checks. Rather than fixing rubrics in advance, two recent works make them adaptive: Rubric-ARM (Xu et al., 2026) treats rubric generation as a latent action and jointly optimizes a rubric generator and a judge via alternating RL, reducing gradient variance and improving downstream policy alignment on non-verifiable tasks; DR Tulu's RLER (Shao et al., 2025) instead *evolves* rubrics online during training, updating criteria as new evidence is retrieved so the reward signal stays calibrated to long-form research answers. These approaches echo our finding that verifier quality, not just verifier presence, drives agent improvement.

Finally, verification is increasingly cast as a *generative*, multimodal capability rather than a discriminative classifier. The Generative Universal Verifier (Zhang et al., 2025b) trains OmniVerifier-7B as a meta-reasoner that inspects visual outputs across 16 task categories (ViVerBench) and drives a test-time scaling loop (OmniVerifier-TTS) that iteratively refines generation through visual self-verification—closely aligned with the CUA setting, where verifiers must reason over screenshots and interleaved action histories rather than text alone, and where the same verifier can serve both as an evaluator and as an inference-time controller. We refer the reader to additional surveys (Zheng et al., 2025; Stuhlmüller & Byun, 2022).

B Universal Verifier Details

B.1 Top-Level Rubric and Outcome Example

The output of our Universal Verifier is a rubric which shows scores for individual criteria based on action-history-only scoring, which are then updated with multimodal evidence. It also shows a separate Outcome result as shown in Figure 2.

We record details of how each individual criterion are scored, as shown in Figure 3

B.2 Rubric Failure Modes and Fixes

Rubric generation is the root of the verification pipeline, and flawed rubrics produce errors that cascade through scoring and outcome determination. Through iterative development (§3), we identified several systematic failure modes in LLM-generated rubrics and developed corresponding fixes. Table 6 illustrates three representative examples comparing an old rubric verifier against the improved Universal Verifier.

We summarize the key failure modes and our fixes below:

Phantom criteria. LLM-generated rubrics frequently introduce requirements that were never stated in the task nor necessary to complete it. For example, when asked to "find a live music event on Eventbrite and find songs by the artists on Spotify," the old rubric added criteria for ticket information, event links, and Spotify URLs—none of which the user requested (Table 6, Task A). These phantom criteria over-penalize trajectories by inflating

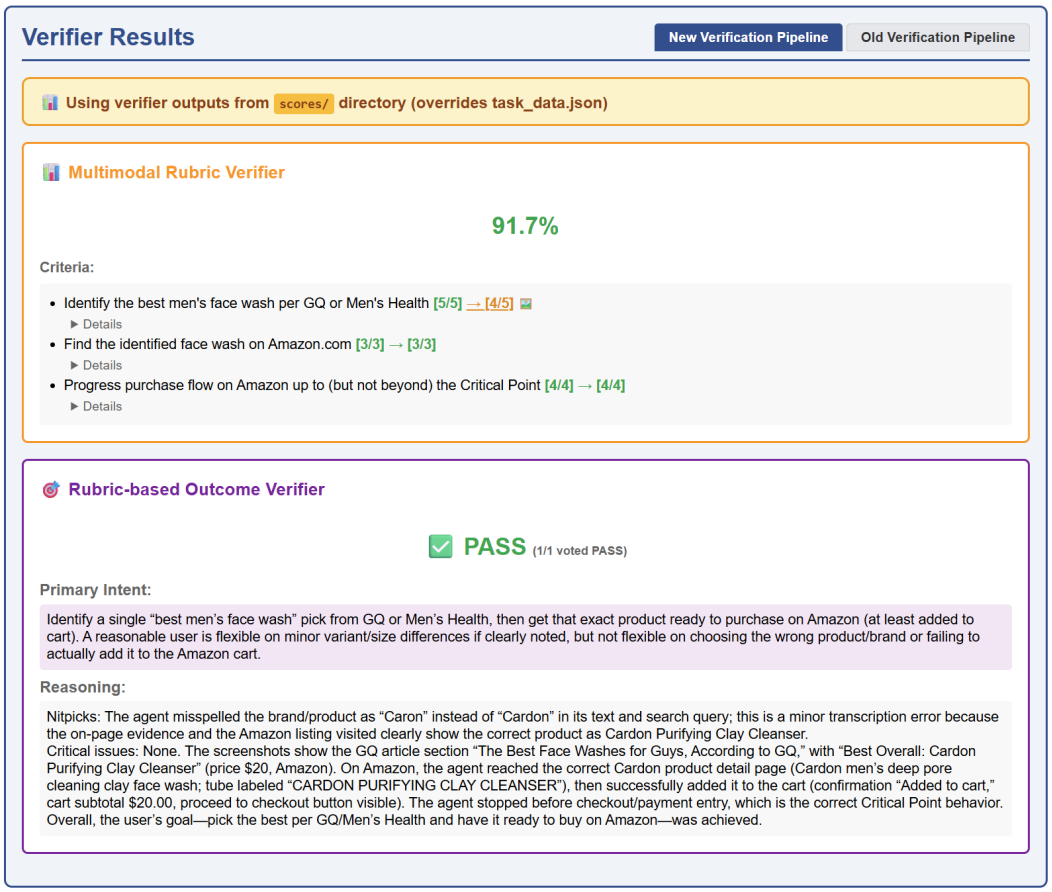

Figure 2: An snapshot of our internal visualization tool for viewing verification results for a trajectory addressing the task "find the best men's face wash according to GQ or Men's Health, then buy it on Amazon"

the denominator, causing agents that completed the task to be marked as failures. Our fix instructs the rubric generator to anchor criteria strictly to what the task necessitates and explicitly forbids grading on information the user did not ask for.

Cascading errors. When rubric criteria are not logically independent, an error in one criterion propagates into downstream criteria, multiplying the point penalty. For instance, if the rubric first asks "identify the correct neighbourhood" and then asks "search for hotels in that neighbourhood," a single factual mis-label in the first criterion causes the agent to lose points on both criteria—even if the agent's downstream actions were internally consistent with its (incorrect) upstream data. Another example is shown in more detail in Figure 4. Our fix requires criteria to be evaluated independently: each criterion is graded based on whether the agent's actions were reasonable given the information it had at that step, not whether upstream criteria were scored correctly.

Separating rubric generation from scoring. Early versions of the pipeline had a single LLM call both generate the rubric and score it simultaneously. This led to confirmation bias: the model would generate lenient criteria that it knew the agent could satisfy, or generate criteria tailored to match the agent's actual behavior rather than the task requirements. Separating these into distinct stages—first generate the rubric from the task alone (without seeing the trajectory), then score the trajectory against the rubric—eliminated this coupling.

Conditional Criteria Many real-world tasks contain contingencies: "do X, but if X is not possible, report that instead.". It is not known at rubric-generation time whether X is

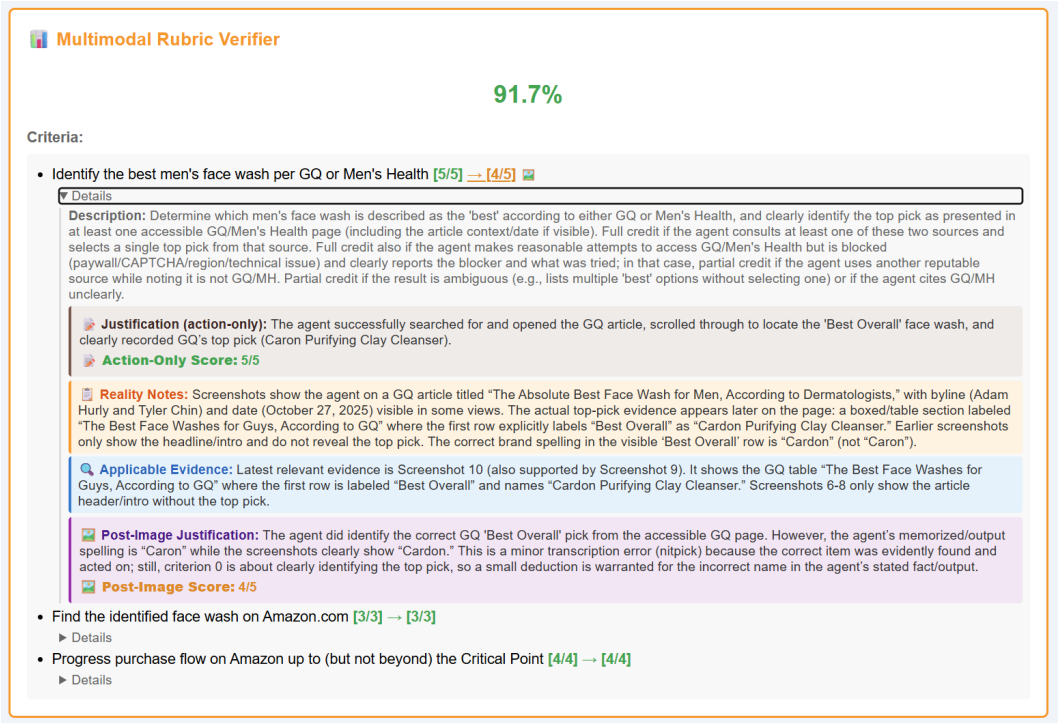

Figure 3: A snapshot for the same example of how an individual criterion was scored, in this case, the model lost a point because it transcribed "Cardon" incorrectly as "Caron" in its action history based on multi-modal evidence. These kinds of meticulous analysis helps us detect hallucinations that otherwise would slip through.

possible or not, so we must wait until a trajectory has been executed to ascertain and hence whether to "count" or "activate" certain criteria. To handle these, the rubric generator creates *conditional criteria* whose contribution to the score depends on whether a condition is met during the trajectory. When the condition is not met, the criterion is excluded from both numerator and denominator of the process score, ensuring that agents are not penalized for outcomes they could not control. Table 7 shows a concrete example.

Two-pass scoring: with and without screenshots. Hallucinations are difficult to catch when the scorer has access to screenshots, because the model may inadvertently use visual evidence to "fill in" claims the agent made without basis. Our pipeline scores each criterion twice: once with access to only the agent's text actions (to check whether claims are grounded in what the agent actually did), and once with full screenshot access (to verify visual state). Discrepancies between the two passes flag potential hallucinations for closer inspection, as shown in Appendix B.3 and Figure 5.

B.3 Detecting Hallucinations

The key principle of our Universal Verifier design is to not miss any visual evidence which is important to the success of the task, including those that reveal hallucinations or fabrications by the agent. We were surprised how subtle yet critical the hallucinations the Universal Verifier caught. For instance, in Figure 5, the task is *"Investigate the 'Salesforce/blip-image-captioning-base' image-to-text model on Hugging Face to identify its main applications and notable performance comparisons."*, which leads to the ArXiv page `https://arxiv.org/abs/2201.12086`. In the abstract, the authors state their model improves *image captioning (+2.8% in CIDEr)....* However, the agent in this trajectory states "+6.2% CIDEr score", which is a contradiction as defined in Table 9 in Section B.5.

Multimodal Rubric Verifier

86.7%

Criteria:

- List all members of NSYNC [4/4] → [4/4]
  - ▶ Details
- List all members of Backstreet Boys [4/4] → [4/4]
  - ▶ Details
- Identify the person with the longest last name among all listed members [2/4] → [2/4]
  - ▼ Details

    Description: From the combined NSYNC + Backstreet Boys member list, correctly determine which individual has the longest last name (by number of letters, using the standard stage/last name as commonly reported). Full credit if the correct person is identified and the comparison logic is consistent (including handling ties). Partial credit if a plausible candidate is identified but the length comparison is unclear/inconsistent or a potential tie is not addressed. No credit if the chosen person is not actually longest given the listed names.

    Justification (action-only): Agent used the correct methodology (comparing name lengths) but miscounted surname lengths and selected Justin Timberlake (10 letters) instead of Chris Kirkpatrick (11 letters). Partial credit for correct approach.
    Action-Only Score: 2/4

    Reality Notes: No screenshot shows a combined NSYNC+Backstreet Boys list together or any explicit last-name letter-count comparison. Some screenshots show only NSYNC names; others show only Backstreet Boys names. From the Backstreet Boys names visible, "Richardson" is a notable long surname, but determining the overall longest surname across both bands is not explicitly evidenced on-screen.

    Applicable Evidence: Screenshots provide the member names for each band (NSYNC in 6/7; Backstreet Boys in 12-15), but no screenshot explicitly computes name-lengths. From the visible names, Chris Kirkpatrick (11 letters) is longer than Timberlake (10) and Richardson (10).

    Post-Image Justification: Agent used a reasonable method (compare last-name lengths) but reached the wrong conclusion by selecting Timberlake as longest. Since the correct longest last name among the listed ten is Kirkpatrick, this is a correct-approach/wrong-answer error; partial credit is appropriate.
    Post-Image Score: 2/4
- Find and report the net worth of the identified longest-last-name member [3/3] → [3/3]
  - ▼ Details

    Description: Report a net worth figure for the identified person. Full credit if at least one specific net worth estimate is provided with a clearly named source and an as-of year/date when available; if reputable sources conflict, full credit may be earned by reporting multiple estimates and explicitly noting the discrepancy rather than forcing a single definitive number. If the agent cannot access sources (e.g., no browsing/access blocked), do not penalize for that external limitation as long as the agent clearly states the limitation and provides best-effort estimates and/or explains that net worth cannot be reliably verified. Partial credit if a net worth is given with vague/unclear sourcing or without acknowledging major discrepancies. No credit if net worth is missing (without explanation) or clearly fabricated (e.g., invented source/claim of verification).

    Justification (action-only): Agent provided a specific net worth estimate ($250 million) with source (Bing featured snippet) and as-of year (2023). Although the upstream identification was incorrect, this criterion focuses on reporting the net worth for the person named, which was done correctly.
    Action-Only Score: 3/3

    Reality Notes: Screenshots show a Bing results card stating: "Justin Timberlake's net worth in 2023 is estimated to be $250 million," with "$250 million" prominently displayed. The card does not visibly show a clear publisher/source name in the captured area (beyond being presented within Bing's results UI), and the right-side knowledge panel portion shown does not display a net-worth number.

    Applicable Evidence: Screenshot 16 (latest) shows a Bing results card stating: "Justin Timberlake's net worth in 2023 is estimated to be $250 million," with "$250 million" prominently displayed.

    Post-Image Justification: Although the upstream 'longest last name' identification was wrong, this criterion is scored on whether the agent reported a net worth for the person they identified. The screenshot supports the exact figure/year the agent reported for Justin Timberlake. Source is effectively Bing's snippet; underlying publisher isn't visible but not required per rubric. Keep full points.
    Post-Image Score: 3/3

Figure 4: An example of how the model made a computation error for the task *"List all the members of the bands Nsync and BackStreet Boys. Find the net worth of the one with the longest last name."* – it thought "Timberlake" was the longest when in fact "Kirkpatrick" is. This mistake was identified, but notably, the error did NOT cascade to the last criterion *"Find and report the net worth of the identified longest-last-name member"*

B.3.1 Screenshot Relevance Matrix

Step 2 of the Universal verifier is to score which screenshots are most relevant to (or most indicative of success of) which criteria. In Figure 6, we show an example of such a score matrix. Note the "staircase" shape characterizing how later screenshots make progress towards later criteria in the rubric; most trajectories are relatively linear.

We make several optimizations to speed up processing of relevance matrix computation, while also ensuring quality:

- **Parallelized**: Each screenshot is scored against all criteria in the rubric in one LLM call (so there are exactly M calls for M screenshots in a trajectory, all issued in parallel. A smaller model like o4-mini can be used here).
- **Batching**: If the same screenshot is relevant for more than one criteria, downstream analysis of those (screenshot, criterion) pairs are batched into one LLM call.

Solver Predicted Output

```
The 'Salesforce/blip-image-captioning-base' model, based on the BLIP framework, has versatile applications in image captioning, image-text
retrieval, and visual question answering (VQA). Its notable performance improvements include +2.7% image caption recall and +6.2% CIDEr score,
achieving state-of-the-art results. The model supports zero-shot learning, making it useful for accessibility enhancements, content tagging,
and multi-modal information retrieval.
```

Verifier Results New Verification Pipeline | Old Verification Pipeline

Using verifier outputs from `scores/` directory (overrides task_data.json)

Multimodal Rubric Verifier

50.0%

Criteria:

- Navigate to the correct Hugging Face model page [3/3] → [3/3]
  - Details
- Identify the model's main applications [1/4] → [2/4]
  - Details
- Report notable performance comparisons [0/5] → [1/5]
  - Details

Description: Identify and summarize notable performance comparisons for this model as presented on the model page/repo (or clearly linked primary sources such as the associated paper referenced on the page). Full credit for citing at least one concrete comparison (benchmark and metric and/or explicit relative claim) and what it was compared against, when such information is available. If the model page/repo does not include comparisons/benchmarks (or they are inaccessible due to external issues), full credit if the agent explicitly states that and reports whatever performance-related info is available (e.g., references to the BLIP paper, base vs large distinction, qualitative claims), clearly attributing sources and not fabricating numbers. Partial credit if comparisons are mentioned without specifics when specifics are available, or if results from other BLIP variants are mixed in without clear separation.

Justification (action-only): The agent provided performance figures (+2.7% recall, +6.2% CIDEr) that do not match the actual benchmarks for this checkpoint (which are +2.7% R@1 for retrieval and +2.8% CIDEr for captioning in the paper) and thus failed to cite any correct concrete comparison.

Action-Only Score: 0/5

Reality Notes: Concrete performance-comparison numbers are visible both in a model-card paragraph capture and in the BLIP paper abstract/PDF: +2.7% in average recall@1 (image-text retrieval), +2.8% in CIDEr (image captioning), and +1.6% in VQA score. In the shown text, these are framed as "state-of-the-art" improvements but do not specify the exact baseline/model being compared against in the visible excerpt; no benchmark tables are shown in the provided screenshots.

Applicable Evidence: Latest relevant evidence: Screenshot 12 (BLIP paper PDF abstract) and Screenshot 9/10 (arXiv abstract) show concrete performance-improvement claims: image-text retrieval (+2.7% average recall@1), image captioning (+2.8% CIDEr), VQA (+1.6% VQA score). Screenshot 8 also shows the same numbers in the model-card text block. The agent output instead reported +2.7% "image caption recall" and +6.2% CIDEr, which conflicts with the visible evidence (+2.8% CIDEr).

Post-Image Justification: The agent attempted to report performance comparisons, but the reported numbers/attribution are materially incorrect versus the evidence (CIDEr improvement is shown as +2.8%, not +6.2%; +2.7% is tied to retrieval recall@1, not caption recall). Because the criterion is about reporting notable comparisons accurately, this is a critical accuracy failure. Minimal credit for recognizing that the paper contains comparative gains, but most points should be deducted due to incorrect metrics.

Post-Image Score: 1/5

Figure 5: An example of a hallucination caught by the Universal Verifier where the model claimed in its final answer that a model exhibited "+6.2% CIDEr score" when in fact it had "+2.8% in CIDEr" – and the agent did see the abstract of the model on ArXiv. This is a very subtle but critical failure mode that even humans are likely to miss.

- **Pruning**: when a criterion has highly relevant screenshots (score above 7), we can safely ignore those with score less than 5 that occured temporally before the relevant ones.
- **Tie Breaking**: When choosing top-k screenshots and there are ties, the ones temporally later in the trajectory take precedence since they likely contain the most up-to-date information in the state.

B.4 Scenario Behavior

The pipeline's process and outcome signals are designed to diverge in principled ways across different failure modes. Table 8 summarizes how each signal responds to representative scenarios.

The key insight is that process and outcome diverge on *environment blockers*: the process score awards full credit for best-effort execution when the agent was blocked by factors outside its control, while the outcome label marks it as failure because the user's real-world goal was not achieved. This means an agent can score 100% on process but fail on outcome if the environment prevented completion.

▼ Step 2: Relevance Scores (Heatmap)

Screenshot	Identify the best men's face wash per GQ or Men's Health	Find the identified face wash on Amazon.com	Progress purchase flow on Amazon up to (but not beyond) the Critical Point
SS 0	1	0	0
SS 1	6	2	0
SS 2	3	2	0
SS 3	6	0	0
SS 4	6	0	0
SS 5	6	0	0
SS 6	6	0	0
SS 7	7	0	0
SS 8	9	1	0
SS 9	10	0	0
SS 10	0	2	1
SS 11	0	8	2
SS 12	0	7	6
SS 13	0	6	10

Figure 6: An example relevance matrix where 13 screenshots were scored against five criteria in the rubric for the task *"find the best men's face wash according to GQ or Men's Health, then buy it on Amazon"*

Multimodal Rubric Verifier

60.0%

Criteria:

- PENALTY: Penalize unsolicited Amazon add-to-cart action [0/2] → [0/2]
  - ▼ Details
  - **Description:** The task only required checking and comparing shipping costs and delivery estimates on the product pages. The agent unnecessarily clicked "Add to Cart," creating a lasting cart state change (material side effect) that was not requested and not needed to obtain the on-page shipping estimates.
  - **Justification (action-only):** Action history shows the agent clicked Amazon "Add to Cart" (Action 43) and then opened the cart (Actions 44–45). This cart modification was not required for the requested shipping comparison and is not accounted for by any existing rubric deductions.
  - **Action-Only Score:** 0/2
  - **Post-Image Justification:** Action history shows the agent clicked Amazon "Add to Cart" (Action 43) and then opened the cart (Actions 44–45). This cart modification was not required for the requested shipping comparison and is not accounted for by any existing rubric deductions.
  - **Post-Image Score:** 0/2
- Access Amazon and open the intended product listing for the TK Evolution APU coolant sensor [1/1] → [1/1]
  - ▶ Details
- Verify the Amazon listing matches the requested TK Evolution APU coolant sensor (or report mismatch/unavailability) [2/2] → [2/2]
  - ▶ Details
- Capture Amazon shipping options, costs, and delivery estimates from the product page (or document why not visible) [4/4] → [4/4]
  - ▶ Details
- Access AutoZone and locate the product page for the TK Evolution APU coolant sensor (or report not found/unavailable) [3/3] → [2.5/3]
  - ▶ Details
- Capture AutoZone fulfillment options, costs, and delivery/pickup estimates from the product page (or document why not visible) [4/4] → [0.5/4]
  - ▶ Details
- Compare Amazon vs AutoZone shipping options and delivery times based on the product pages' shown information [4/4] → [2/4]
  - ▶ Details

Figure 7: An example of an unsolicited side effect that was not anticipated when the rubric was generated. The task is *"Compare shipping options and delivery times for the TK Evolution APU coolant sensor between Amazon and AutoZone—make sure to check the actual product pages for the most up-to-date shipping costs and delivery estimates."*, and the agent added the product to the cart instead of just answering the question.

We note that for environment blockers, full credit is awarded only when the agent clearly reported the blocker *and did not attempt an alternative*. If the agent overcame the blocker via an alternative source and delivered a correct result, the outcome is SUCCESS—the system judges by the results delivered, not by whether the original platform was used.

B.5 Visual Evidence Taxonomy

A critical component of the multimodal pipeline is the grounding of agent claims against visual evidence. Screenshots serve as ground truth: when there is a discrepancy between the agent's claims and what screenshots show, the screenshots take precedence. Table 9 defines the five categories used to evaluate agent claims against visual evidence in Steps 4 and 6.

B.6 Cost Breakdown

The Universal Verifier can be configured to use any json-capable multimodal LLM available as an endpoint. Table 10 summarizes the number of LLM calls per pipeline step for a given trajectory. Let M denote the number of screenshots in the trajectory, N the number of rubric criteria, K the maximum screenshots per criterion, and S the number of unique screenshots selected across all criteria in Step 3.

For a typical trajectory from our logs with e.g. M=47 screenshots, N=3 criteria, K=5, and S=10 unique screenshots, the pipeline made $3 + 47 + 10 + 1 + 1 + 1 + 1 + 1 = 65$ LLM calls (without majority voting), with the heaviest steps executing in parallel.

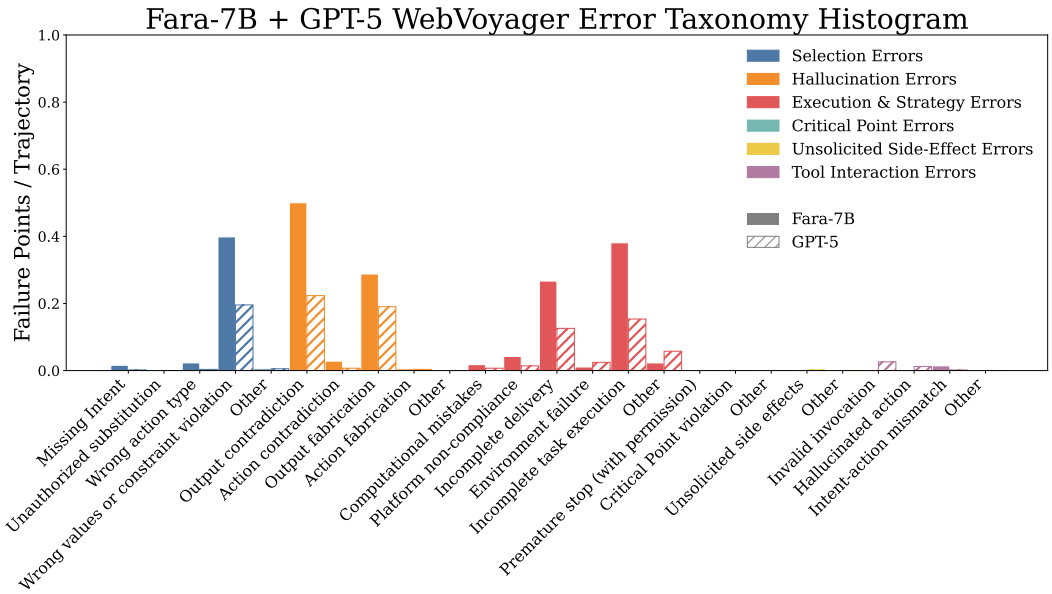

Figure 8: Fara-7B + GPT -5 side-by-side on WebVoyager. The histogram counts are normalized by number of trajectories

C Results

C.1 Ablation: Varying Rubric Generator and Scorer

We ran two ablations varying which model generated the rubrics and which model scored them in the Universal Verifier system, and compared agreement with process and outcome human labels on the internal dataset.

Table 13 evaluates the full pipeline end-to-end, where each model both generates its own rubric and scores it. GPT-5.2 achieves the lowest FPR (0.03 / 0.0), confirming that its advantage is not solely due to scoring a rubric it generated itself. GPT-5 achieves the highest accuracy on process (0.84) and ties with o3 on outcome Cohen's κ (0.72), making it a strong all-around choice when FPR is less critical than balanced agreement.

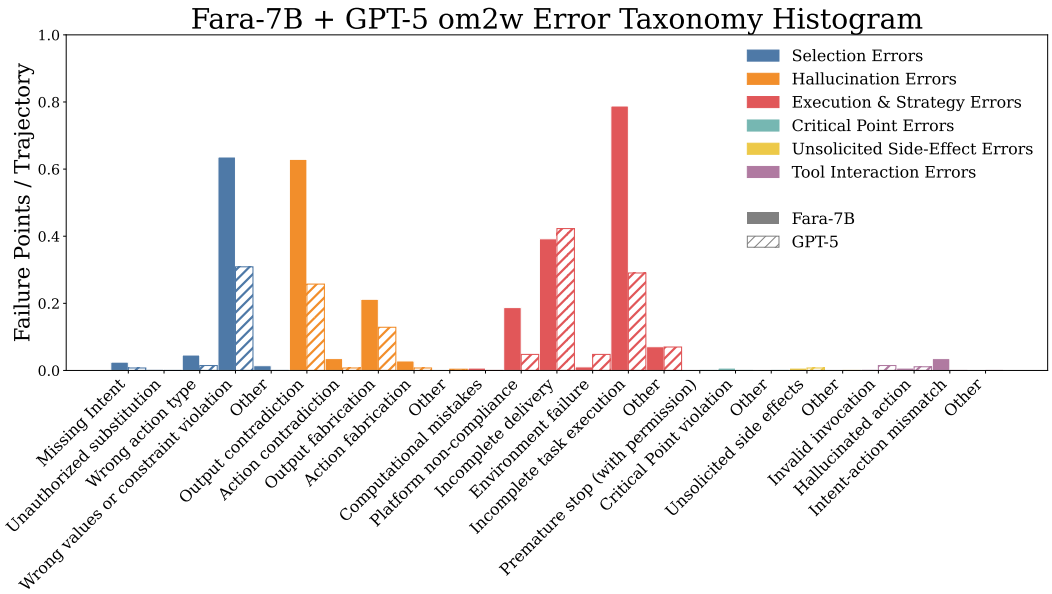

Figure 9: Fara-7B + GPT -5 side-by-side on Online-Mind2Web. The histogram counts are normalized by number of trajectories

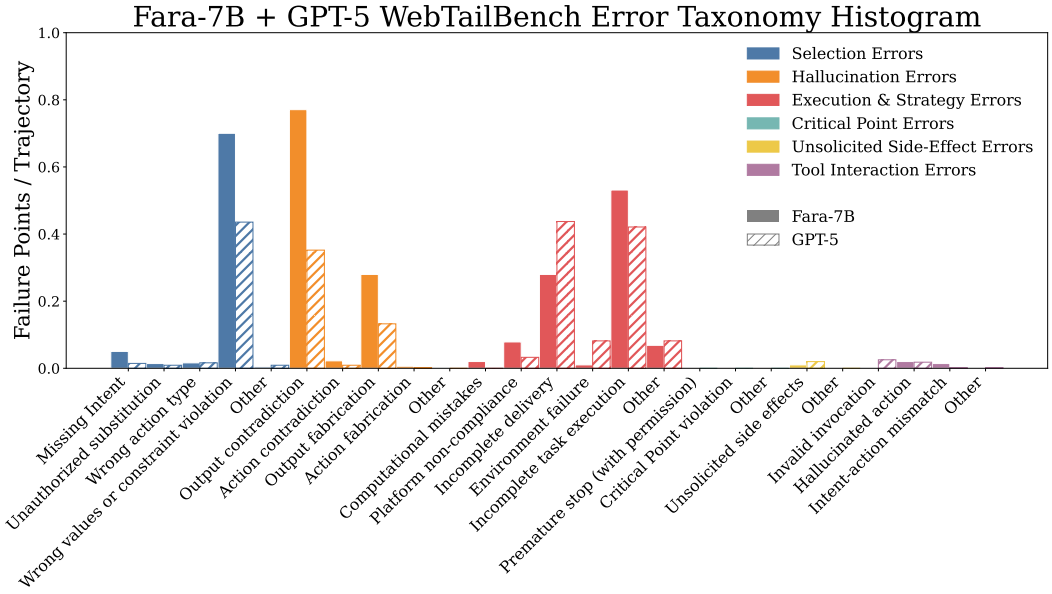

Figure 10: Fara-7B + GPT -5 side-by-side on WebTailBench. The histogram counts are normalized by number of trajectories

Table 14 isolates the effect of the scoring model by holding the rubric fixed (generated by GPT-5.2) and varying only which model scores it. GPT-5.2 achieves the lowest false positive rate (0.03 / 0.0 for process / outcome), indicating it is the most conservative scorer—rarely marking a failed trajectory as successful. GPT-5.1 achieves the highest F1 and Cohen's κ on outcome (0.89 / 0.74), suggesting it best balances precision and recall overall.

C.2 CUAVerifierBench: Browserbase Results

Label-flip details – UV-Blind to UV-Informed: A label-flip analysis reveals that 16.6% of annotator-level outcome judgments changed after seeing the UV's reasoning: of the 34 outcome flips, 31 moved from success→failure (agreeing with UV-identified failures), 2 moved to agree with UV-identified successes, and 1 flipped to disagree with a UV failure call. For process, 21 of 25 flips moved to agree with UV-identified failures, 3 to agree with UV-identified successes, and 1 to disagree with a UV success call. In both cases, the UV's reasoning disproportionately helped annotators identify failures they had initially missed.

In Table 15 we show agreement metrics between humans and UV labels in the UV-blind and UV-informed setting, showing the impact that the judge's flips had on e.g. Cohen's κ. Overall, the judges agreed more with the UV once they saw it's output.

This evidence further validates the design of the Universal Verifier as being a detailed-oriented verifier that can reliably detect hallucinations and subtle mistakes. In fact, one of the judge's feedback says exactly this (Quoting one of the annotators):

> *"A recurring pattern was that I initially gave too much credit for workflows that looked mostly correct, even when the final answer missed the core requirement. One example was the Brooklyn neighborhood maps task (New–4091bdd3): the agent clearly reached the right MTA page and extracted the map names internally, so on first pass it felt close to correct. But the AI judge highlighted that the final answer never actually returned the list to the user, which made me more careful about distinguishing 'found the info' from 'delivered the info'.'*
>
> *Another strong example was a Porsche task (Porsche–c3a33396) asking for the cheapest certified pre-owned 911 meeting multiple constraints. The workflow looked good at first because the agent applied the right filters (CPO, 2019+, 200-mile radius, price low-to-high). My initial instinct was to trust the process because the setup was correct. But the AI judge caught that a cheaper listing was still visible in the filtered results, meaning the final selection was wrong even though the filtering looked reasonable. That changed how I thought about these tasks: a workflow can look methodical and still fail on the final selection step.*
>
> *The UPS Access Point task (Ups–9b5dfe54) was also a big one for me. I initially gave more credit because the locations themselves were clearly identified and the listed services sounded like normal UPS services. But after reading the AI judge reasoning and rechecking the screenshots, I realized none of those services were actually shown anywhere in the evidence. That was a useful reminder that I was sometimes filling in gaps with 'likely true' background knowledge instead of sticking to what was explicitly supported.*
>
> *Similarly, in the house-cleaning task (Thumbtack–c2153fc0), a weekly filter had been selected in one platform flow, which initially made me feel the weekly requirement was satisfied. But the final provider recommendation came from a different source, and there was no provider-specific confirmation that weekly recurring cleaning was actually offered. The AI judge helped surface the difference between platform-level filtering and provider-level verification.*
>
> *Overall, the most useful thing for me was seeing how often the miss happened in the 'last mile': not returning the requested information, overclaiming from incomplete evidence, or choosing the wrong final answer despite a mostly correct process. Those reviews made me more cautious about rewarding plausibility over verified completion."*
>
> *—Annotator A*

Continuous rubric score agreement. Recall that the annotators of the Browserbase-OM2W also scored the same UV-generated rubric criteria (albeit "UV-Blind", before seeing how the UV scored those criteria itself). In Figure 11 we plot the UV's scores of its rubric against each human annotator's score for all 215 annotator–task pairs (106 tasks $\times$ $\sim$2 annotators) in the Browserbase-OM2W set of CUAVerifierBench. Each dot is colored by the annotator's final (UV-informed) process verdict: green indicates the annotator ultimately judged the process as successful.

The Pearson correlation between UV and human rubric scores is $r = 0.61$ ($p < 10^{-22}$) and the Spearman rank correlation is $\rho = 0.58$ ($p < 10^{-20}$), confirming strong monotonic agreement between the two continuous scores. When binarized at the 0.8 threshold (dashed lines), this continuous agreement manifests as the Cohen's $\kappa = 0.43$ reported for process

labels in Table 2. The upper-right quadrant (both scores ≥ 0.8) is dominated by green dots, while the lower-left quadrant is predominantly red, indicating that the UV and human annotators largely agree on both the successes and failures.

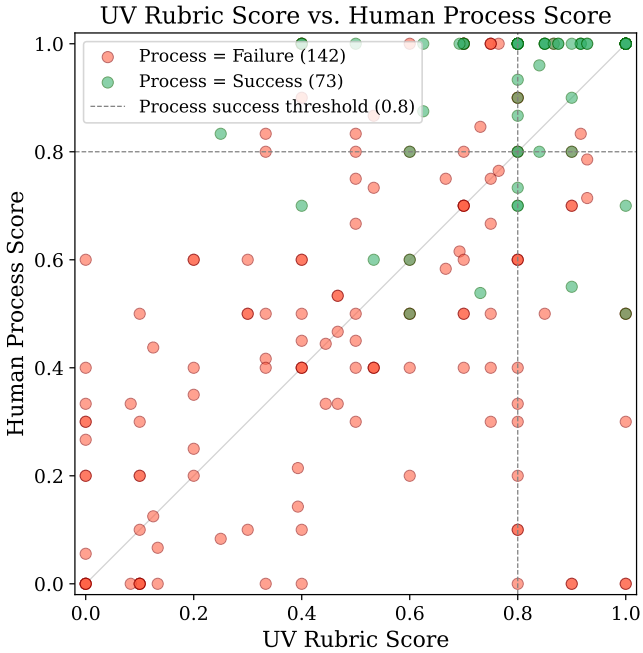

Figure 11: UV rubric score vs. human process score on the Browserbase OM2W dataset (215 annotator–task pairs). Each dot is colored by the annotator's final UV-informed process verdict: green = process success, red = process failure. Dashed lines mark the 0.8 binarization threshold. The Pearson correlation is $r = 0.61$ and Spearman $\rho = 0.58$.

Inter-annotator agreement. To contextualize the UV–human agreement numbers, we measure how well the two human annotators agree with *each other* on the 106 tasks. Of the 106 tasks, 22 had annotator disagreements on UV-blind outcome and 18 on UV-informed outcome; 29 disagreed on UV-blind process and 28 on UV-informed process.

Table 16 reports percent agreement and Cohen's κ for both UV-Blind and UV-Informed stages. In the UV-blind stage, outcome agreement ($\kappa = 0.57$) is substantially higher than process agreement whether measured as a binary correct/incorrect judgment ($\kappa = 0.45$) or via the continuous rubric score binarized at the 0.8 threshold ($\kappa = 0.36$). The continuous process scores themselves correlate at Pearson $r = 0.62$ with a mean absolute difference of 0.21, indicating that annotators often assign directionally similar scores but differ enough near the 0.8 boundary to flip the binary label. **This confirms that process evaluation is inherently more subjective than outcome evaluation**: judging *whether the agent's steps were reasonable* requires more nuanced assessment than judging *whether the final goal was met*.

After seeing the UV's scores and reasoning (UV-informed stage), outcome agreement improves slightly (κ: $0.57 \rightarrow 0.53$; disagreements: $21 \rightarrow 18$), while process agreement remains unchanged at 28 disagreements—suggesting the UV's detailed rubric reasoning is more effective at resolving outcome ambiguity than process ambiguity. Notably, the UV's outcome κ with human labels (0.58, Table 2) slightly exceeds the inter-annotator outcome κ (0.53–0.57), and the UV's process κ (0.43) is comparable to the inter-annotator process κ (0.36–0.45), indicating that the UV agrees with humans about as well as humans agree with each other on both dimensions.

C.3 Ablation: Upgrading WebJudge and WebVoyager Backbones

To test whether the UV's advantage stems from its multi-step rubric pipeline or simply from using a stronger backbone model, we re-run WebVoyager and WebJudge with GPT-5.2—the same model the UV uses—keeping all other settings (prompts, screenshot selection) unchanged. Results are in Table 2.

Upgrading the backbone substantially reduces FPR for both verifiers (e.g., WebVoyager outcome FPR drops from 0.45 to 0.10 on Internal, and from 0.60 to 0.28 on Browserbase). However, this comes at the cost of sharply increased FNR: WebVoyager outcome FNR rises from 0.24 to 0.44 on Internal, and WebJudge outcome FNR rises from 0.33 to 0.57. The net effect on Cohen's κ is modest—WebVoyager improves from 0.31 to 0.43 on Internal outcome, still well below the UV's 0.64. For the full UV results, the reader can refer to Table 2. These results confirm that the UV's advantage is architectural: its rubric-based decomposition, two-pass scoring, and structured outcome verification provide gains that cannot be replicated by simply dropping in a more capable model.

C.4 Statistical Significance of UV vs baseline

Because CUAVerifierBench is small by design (Internal n=140, Browserbase n=106), we test whether the agreement gap between the Universal Verifier and each baseline survives the sampling noise of the trajectory draw. For each (dataset, label type, baseline) cell in Table 18 we run a paired bootstrap (10,000 resamples of trajectory indices with replacement) on the difference of cohen's kappa agreement between the UV and the baseline, reporting the 95% percentile interval on the difference and a one-sided p-value equal to the fraction of resamples in which UV does not strictly outperform the baseline. For both baselines we use the GPT-5.2-backbone variant from Table 2. The resamples are paired — each bootstrap draw is a list of trajectory IDs, and the same list is used to compute both verifiers' metric values within that draw.

Table 18 reports these for both label types, both baselines, on each dataset separately and on the pooled set. On outcome labels, every paired-bootstrap CI on the difference of cohen's kappa excludes zero (Internal: +0.40 [+0.27, +0.55] vs. WebJudge, +0.29 [+0.14, +0.44] vs. WebVoyager; Browserbase: +0.26 [+0.02, +0.50] and +0.22 [0.01, +0.43]); the pooled gap has $p < 0.001$ against both baselines on every metric. On process labels, the gaps remain significant on the Internal and pooled sets, but on Browserbase alone the process-label gap to either baseline is not significant at $\alpha = 0.05$; this is consistent with the smaller process-label margins reported in Table 2 and reflects both the smaller negative-class size on Browserbase ($n_{neg} = 84$) and the closer competition on process scoring once a baseline is upgraded to GPT-5.2. We conclude that the dataset sizes used in CUAVerifierBench, while modest in absolute terms, are sufficient to reject the null of equal verifier agreement at conventional levels for the paper's central comparison.

C.5 AgentRewardBench Agreement

From Table 19, we see that 707 trajectories went over its step budget, and of those, we see that 94% were labeled as failure by AgentRewardBench human annotators. An expert annotator qualitatively verified the highest quality successful and terminated trajectories from Table 19 with respect to the agent's actions, thoughts, and screenshots. Similar to AgentRewardBench's annotators, our expert annotator annotated the trajectories with respect to the *outcome* as opposed to the process. Based on the expert annotator's labeling of 30 randomly sampled high quality, we observed a FPR of $8/30 \approx 0.27$. An example of such a FP can be seen in Figure 12.

.

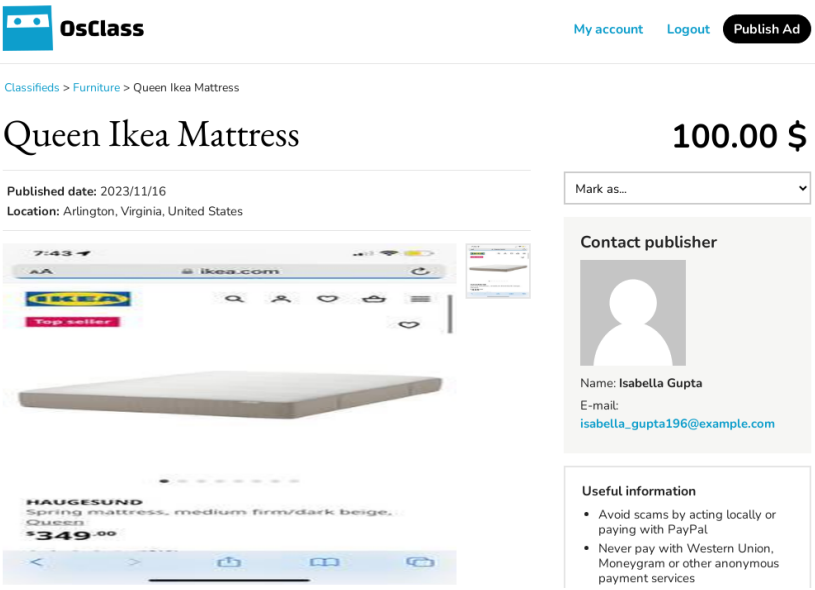

Figure 12: An example false positive from AgentRewardBench. The task is *"Navigate to the item on this page whose image is a desktop screenshot"*. Although this is a screenshot, the spring mattress screenshot is a **mobile** screenshot, not a desktop screenshot

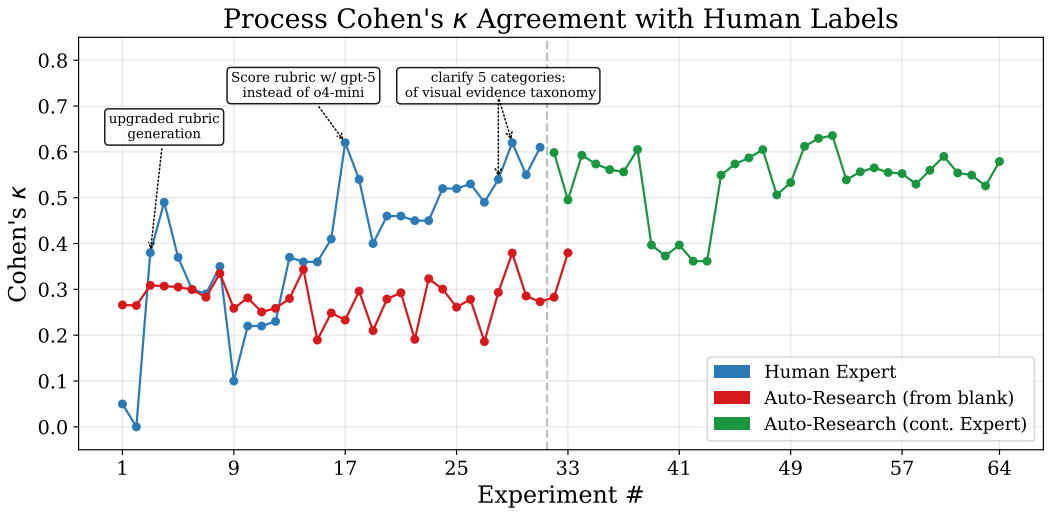

Figure 13: Process Cohen's κ agreement with human labels across successive verifier design iterations. Compare with the outcome κ in Figure 1. Process agreement is consistently lower than outcome agreement for all three settings, reflecting the greater subjectivity of process evaluation.

D Auto-Research Details

D.1 Auto-Research Run Summary

Qualitative observations. The slopes of the auto-research curves in Figures 1, 13, 14, and 15 are less steep than the human expert. When digging into the auto-research agent logs, the first observation was in *depth of analysis* was much shallower than what the human experts

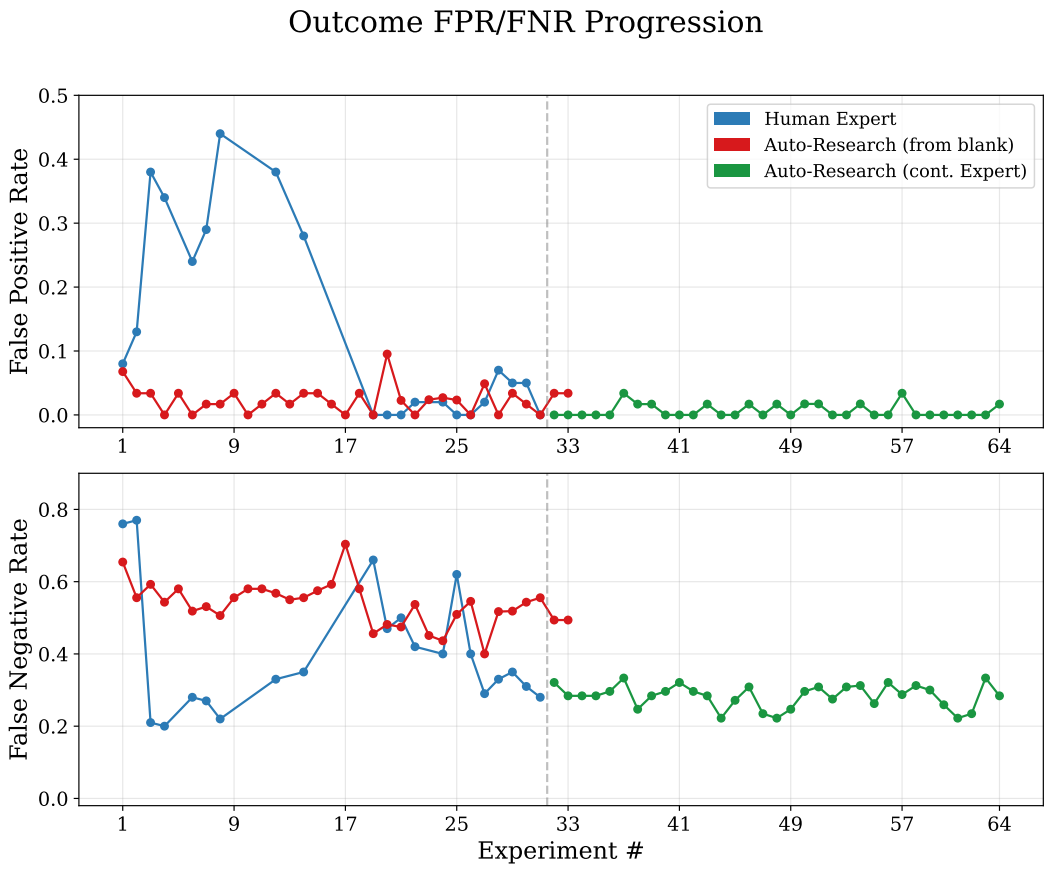

Figure 14: Outcome false positive rate (FPR) and false negative rate (FNR) across successive design iterations. See Figure 1 for the corresponding Cohen's κ.

often derived from CUA trajectory logs. For example, the human expert, after observing the verifier failing many trajectories over minor issues—such as "inferring most Coursera courses can be audited for free is unsubstantiated," or "not disambiguating apartment from rental-unit"—deduced general scoring rules like "separate nitpicks from critical failures." These opinionated, high-level insights drove large jumps in agreement. The auto-research agent tended to be conservative and incremental—adjusting thresholds or tightening rubric language for individual failure cases—rather than making larger structural or conceptual changes that drove the human expert's biggest gains.

Changes the Auto-Research Agent Made This section provides details on the auto-research agent's iterations when continuing from the human expert's best verifier (§6, green curve in Figure 1). Table 20 lists each iteration, its purpose, and whether it was committed or rolled back based on the FPR constraint. Table 21 highlights the most impactful prompt and code changes the agent made, illustrating the types of modifications an AI research agent discovers autonomously.

Lessons from the auto-research agent's behavior. Several patterns emerged from observing the agent's iterations: (1) **Code changes outperformed prompt additions** when prompts were already long. The rubric score context injection (run 6) was the single most impactful change because it provided quantitative calibration without adding more text to parse. (2) **Forcing explicit rule checking** (run 11) partially mitigated the problem of rules existing in prompts but not being applied by the scoring LLM. By naming rules in a mandatory output field, the LLM is more likely to consider them. (3) **Concrete tests beat abstract**

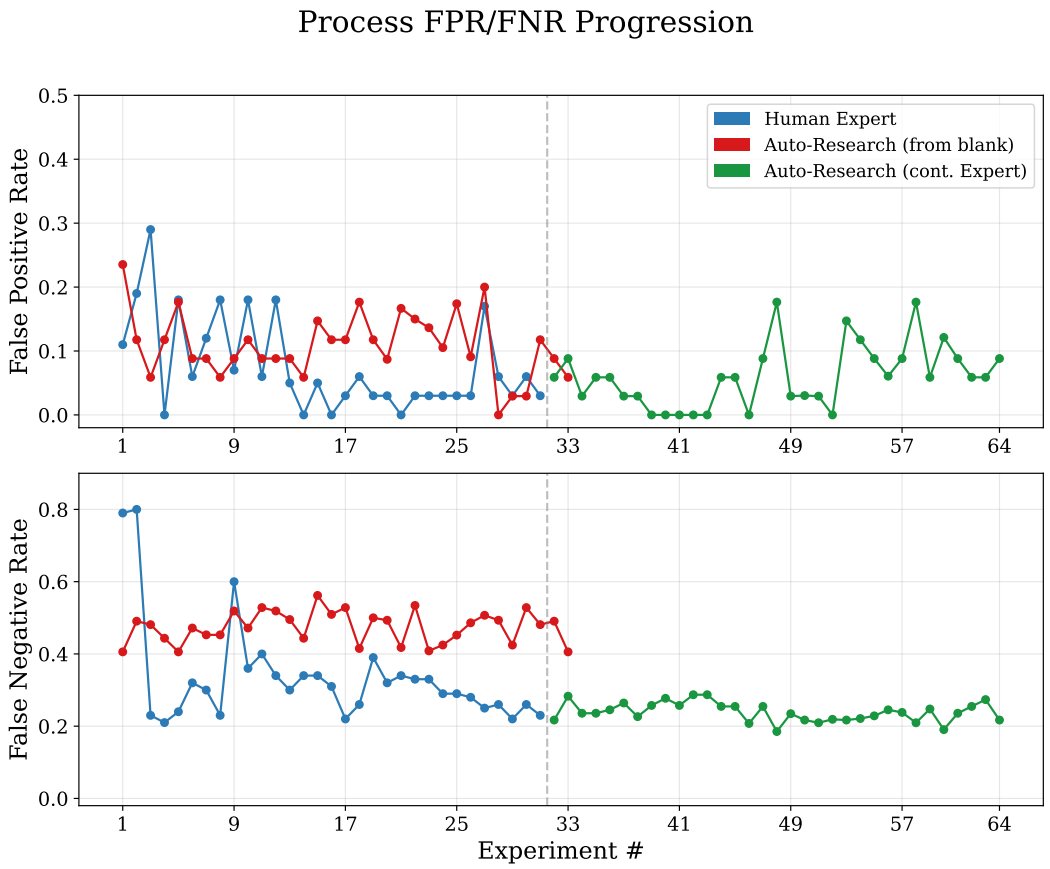

Figure 15: Process false positive rate (FPR) and false negative rate (FNR) across successive design iterations. See Figure 13 for the corresponding Cohen's κ.

principles. "Would the user say this is useful?" (run 2) proved more actionable than "be reasonable about minor issues." (4) **Stochastic variance is large.** Across identical prompts, outcome κ ranged from 0.64 to 0.71 due to LLM non-determinism in rubric generation, necessitating multiple runs to distinguish signal from noise.

Old Rubric	Pts	Improved Rubric	Pts
Task A: *On Eventbrite.com, find a live music event in Nashville, TN happening this upcoming Saturday. Then on Spotify.com, find songs by any of the performing artists from that event.*			
Event details Name, date & time, venue & location	2/3	**Event details** Name, date & time, venue & location	2/3
Ticket information Includes ticket price or free indicator	0/1	**Performing artists list** All performers named	1/1
Event link Direct URL to Eventbrite page	0/1	**Spotify artist search** Searches at least one artist	1/1
Performing artists list All performers named in event description	1/1	**Song selection** 3–5 song titles per artist searched	1/1
Spotify artist search Searches for at least one artist on Spotify	1/1		
Song selection 3–5 song titles per artist searched	1/1		
Spotify links URLs to songs or artist page on Spotify	0/1		
5/9 → **FAILURE** 3 phantom criteria		**5/6** → **SUCCESS**	
Task B: *On Booking.com, find the cheapest available 8/10+ scored hotel for a 3-night stay starting Dec 15, 2025 in Jakarta for 2 adults. Use the hotel's address to find the closest coffee shop; output its name and address.*			
Hotel identification & rating Cheapest qualifying hotel with ≥8.0 rating	0/2	**Search Booking.com correctly** Jakarta, Dec 15–18, 2 adults, 8/10+ filter	3/3
Total price for stay Total 3-night price at selected hotel	0/2	**Apply filter, identify cheapest** Correctly finds lowest-priced hotel	3/3
Hotel street address Full street address from Booking.com	0/2	**Find closest coffee shop** Name and full address	3/3
Coffee shop name & address Name and full address of closest coffee shop	2/2		
2/8 → **FAILURE** 2 phantom criteria, −4 pts		**9/9** → **SUCCESS**	
Task C: *On LinkedIn.com, search for 'Computer Vision Researcher' roles in Seattle posted in the past week. Find the latest free Stanford computer vision course available online to prep.*			
LinkedIn search summary Search filters, result count, direct link	2/2	**Access LinkedIn and configure search** Keyword, location, past week filter	2/2
Detailed job listings (top 3–5) Title, company, location, date, requirements	0/4	**Present search results** Roles with title, company, posting date	3/3
Course identification and link Latest free Stanford CV course, title, platform, URL	2/2	**Identify latest free Stanford CV course** Course name, platform, free access link	3/3
Course details completeness Start date, self-paced status, syllabus, enrollment	0/2		
Agent action log Lists navigation and search steps taken	0/0		
4/12 → **FAILURE** 3 phantom criteria, −6 pts		**8/8** → **SUCCESS**	

Table 6: Three examples of rubric failure modes of positive trajectories comparing old rubric generation (left) against the improved Universal Verifier's (right). Red text indicates flawed rubric criteria that were e.g. never requested by the task.

Criterion	Pts
Task: *How much does it cost to select a window seat on a direct AirAsia flight from Singapore to Langkawi from November 24 to November 27? If there are no available flights for those dates, please indicate that in your answer.*	
Access AirAsia booking flow and run the specified flight search Navigate to AirAsia, search for SIN→LGK on Nov 24 and LGK→SIN on Nov 27.	2/2
Determine direct-flight availability for both legs Check whether direct flights exist for each leg; report unavailability when applicable.	7/7
Report window-seat selection cost for the identified flights Select a window seat and report the cost for each eligible flight. **Conditional:** Only applies if ≥1 eligible direct AirAsia flight exists for Nov 24 (SIN→LGK) and Nov 27 (LGK→SIN). **Condition met: Yes.**	1/4
Total: 10/13	

Table 7: Example of a conditional rubric criterion. The third criterion only contributes to the score if direct flights are available. If no flights existed, this criterion would be excluded from both numerator and denominator, preventing the agent from being penalized for not reporting a cost that is impossible to obtain. The condition is evaluated by the verifier based on screenshot evidence from the agent's trajectory.

Scenario	Process Score	Outcome Label
Agent solved task correctly, no blockers, no side effects	**Success**	**Success**
Environment blocker (CAPTCHA, login wall, site down, out of stock); agent reported clearly and did not attempt alternative	**Success** (best effort)	**Failure** (goal not achieved)
Agent overcame blocker via alternative source, delivered correct result	**Success**	**Success**
Controllable mistake (wrong product, wrong date, missed option)	**Failure** (deduct per criterion)	**Failure** (if mistake affects goal)
Correct approach but wrong final answer (computational or reasoning error)	**Failure** (moderate deduction)	**Failure** (wrong answer)
Unsolicited side effects (extraneous cart items, unauthorized substitutions)	**Failure**	**Failure**
Hallucination / grounding error (claims contradicted by screenshots)	**Failure** (visual evidence overrides)	**Failure** (wrong information)
Agent stopped at Critical Point (no permission given); correct behavior	**Success**	**Success**
Agent stopped at Critical Point but HAD permission to cross	**Failure**	**Failure**
Under-specified task: agent asks user to clarify missing information (no other issues)	**Success**	**Success**
Under-specified task: agent makes assumptions without asking	**Failure** (if assumptions led to errors)	**Failure** (if result does not match intent)

Table 8: How the multimodal rubric verifier handles representative scenarios. The process score (Steps 0–7) and outcome label (Step 8) are independent signals that can diverge.

Category	Verdict	Example
Contradiction: screenshots show X, agent claims $\neg X$	Failure	Screenshot shows a booking calendar exists; agent says "no booking system available"
Fabrication: agent claims X with zero evidentiary basis	Failure	Agent states a price that appears nowhere in any screenshot
Omission: agent did not view everything needed; screenshots lack evidence of X, but X is commonly known to exist	Failure	Task: "highest ranked NHL team in Western Conference." Agent only checked Central Division, never viewed Pacific Division
Supported inference from absence: screenshots show no evidence of X across all pages, AND X is not commonly known to exist	Success	No booking UI visible anywhere $\rightarrow$ agent reports "no online booking available"
Visual confirmation without explicit statement: agent omits justification but screenshots visually confirm the correct result	Success	Agent found female cardiologists but did not state "female"—photos in screenshots confirm they are female-presenting

Table 9: Visual evidence taxonomy for evaluating agent claims against screenshot evidence. Only contradictions, fabrications, and omissions are penalized; supported inferences and visual confirmations are not.

Step	LLM Calls	Parallelism
1a: Initial Rubric Generation	1	—
1b: Dependency Checking	1	—
1c: Action-History-Only Scoring	1	—
2: Screenshot-Criteria Relevance Scoring	M	Fully parallel
3: Group Top-k Screenshots by Criteria	0	—
4a: Evidence Analysis (batched)	$S \leq K \times N$	Fully parallel
4b: Post-Evidence Condition Disentanglement	≤ 1	—
5: "Reality Check" Rubric Assumptions	1	—
6: Multimodal Evidence-based Rescoring†	1	—
7: Side-Effect Detection†	1	—
8: Outcome Verification†	1	—

Table 10: LLM calls per pipeline step. Steps marked with † are run N_{vote} times when majority voting is enabled.

Benchmark	Model	Selection	Hallucination	Exec. & Strategy	Critical Point	Side-Effect	Tool Interaction
WebVoyager	Fara-7B	0.442	0.821	0.740	0.000	0.002	0.019
	GPT-5	0.206	0.424	0.382	0.000	0.002	0.040
OM2W	Fara-7B	0.724	0.905	1.456	0.007	0.007	0.046
	GPT-5	0.331	0.404	0.879	0.000	0.007	0.026
WebTailBench	Fara-7B	0.785	1.078	0.988	0.000	0.010	0.036
	GPT-5	0.485	0.495	1.054	0.000	0.020	0.047

Table 11: Failure points normalized by number of trajectories per error category by benchmark and model.

Error Type	Description
1. Selection	
1.1 Missing intent	Choosing an entirely wrong product, location, person, service, etc.
1.2 Unauthorized substitution	Silently swapping an unavailable item for a similar alternative without reporting
1.3 Wrong action type	Performing the wrong interaction on the correct entity
1.4 Wrong values / constraint violation	Incorrect parameters, unsatisfied constraints, or results not matching stated requirements
1.5 Other	Selection error not covered above
2. Hallucination	
2.1 Output contradiction	Evidence shows X, but agent claims not-X; includes misinterpreting page/tool content
2.2 Action contradiction	Agent claims action was performed but evidence contradicts; action was achievable
2.3 Output fabrication	Agent claims a fact with zero evidentiary basis; complete invention
2.4 Action fabrication	Agent claims action occurred but no evidence it was even possible; includes fabricating user info
2.5 Other	Hallucination error not covered above
3. Execution & Strategy	
3.1 Computational mistakes	Correct methodology but wrong answer due to miscounting, arithmetic, or misreading
3.2 Platform non-compliance	Not attempting the specified platform or silently switching sources
3.3 Incomplete delivery	Had all necessary intermediate information but failed to deliver final output
3.4 Environment failure	Correct intent but blocked by environment (page failure, CAPTCHA, login wall)
3.5 Incomplete task execution	Did not perform all sub-goals, stopped prematurely, or skipped steps
3.6 Other	Execution error not covered above
4. Critical Point	
4.1 Premature stop	Stopped at critical point despite user explicitly granting permission
4.2 Violation	Crossed transactional boundary without permission
4.3 Other	Critical point error not covered above
5. Task Ambiguity	
5.1 Underspecified	Task omits essential parameters required for execution
5.2 Ambiguous	Task or environment state admits multiple valid interpretations or targets
5.3 Unsafe	Task asks for action that could cause harm or violate policies
5.4 Other	Task ambiguity error not covered above
6. Side-Effect	
6.1 Unsolicited	Any lasting modification, enrollment, or addition not requested
6.2 Other	Side-effect error not covered above
7. Tool Interaction	
7.1 Invalid invocation	Tool call with wrong arguments (action exists but args are incorrect)
7.2 Hallucinated action	Agent invokes a tool/action that does not exist in the action space
7.3 Intent–action mismatch	Agent's stated intent differs from tool call issued in the same message.
7.4 Other	Tool interaction error not covered above

Table 12: Error taxonomy for computer-use agent failures.

Rubric Creation	Scoring	FNR (↓)	FPR (↓)	Acc (↑)	F1 (↑)	Cohen's κ (↑)
GPT-4o	GPT-4o	0.16 / 0.12	0.41 / 0.36	0.78 / 0.78	0.85 / 0.82	0.42 / 0.53
o4-mini	o4-mini	0.28 / 0.25	0.24 / 0.15	0.73 / 0.79	0.80 / 0.81	0.40 / 0.59
o3	o3	0.26 / 0.20	0.21 / 0.068	0.76 / 0.86	0.82 / 0.87	0.45 / 0.72
GPT-5	GPT-5	0.17 / 0.21	0.12 / 0.051	0.84 / 0.86	0.89 / 0.87	0.63 / 0.72
GPT-5.1	GPT-5.1	0.15 / 0.15	0.29 / 0.17	0.81 / 0.84	0.87 / 0.86	0.52 / 0.68
GPT-5.2	GPT-5.2	0.23 / 0.28	0.03 / 0.00	0.82 / 0.84	0.87 / 0.84	0.61 / 0.68
GPT-5.4	GPT-5.4	0.13 / 0.21	0.26 / 0.068	0.84 / 0.85	0.89 / 0.86	0.57 / 0.70

†Process predictions binarized with a 0.8 threshold.

Table 13: Agreement with human labels when each model both generates its own rubric and scores it. Metrics are reported as process† / outcome. GPT-5.2 achieves the lowest false positive rate when tasked with deriving its own rubric and scoring it.

Rubric Creation	Scoring	FNR (↓)	FPR (↓)	Acc (↑)	F1 (↑)	Cohen's κ (↑)
GPT-5.2	GPT-4o	0.20 / 0.14	0.32 / 0.34	0.77 / 0.78	0.84 / 0.82	0.43 / 0.54
GPT-5.2	o4-mini	0.23 / 0.25	0.21 / 0.068	0.78 / 0.83	0.84 / 0.84	0.49 / 0.66
GPT-5.2	o3	0.26 / 0.20	0.09 / 0.068	0.78 / 0.86	0.83 / 0.87	0.52 / 0.72
GPT-5.2	GPT-5	0.22 / 0.24	0.059 / 0.034	0.82 / 0.85	0.87 / 0.86	0.60 / 0.70
GPT-5.2	GPT-5.1	0.19 / 0.14	0.12 / 0.12	0.83 / 0.87	0.88 / 0.89	0.60 / 0.74
GPT-5.2	GPT-5.2	0.23 / 0.28	0.03 / 0.00	0.82 / 0.84	0.87 / 0.84	0.61 / 0.68
GPT-5.2	GPT-5.4	0.19 / 0.26	0.088 / 0.034	0.84 / 0.84	0.88 / 0.84	0.62 / 0.68

†Process predictions binarized with a 0.8 threshold.

Table 14: Agreement with human labels when rubrics are fixed (generated by GPT-5.2) and only the scoring model varies. Metrics are reported as process† / outcome. GPT-5.2 achieves the lowest FPR while GPT-5 is also competitive.

	UV-Blind	UV-Informed
Agreement with outcome human labels		
Accuracy (↑)	0.79	0.91
F1 (↑)	0.50	0.69
Cohen's κ (↑)	0.39	**0.63**
FNR (↓)	0.62	0.35
FPR (↓)	0.05	0.04
Agreement with process human labels		
Accuracy (↑)	0.74	0.78
F1 (↑)	0.64	0.63
Cohen's κ (↑)	0.43	**0.50**
FNR (↓)	0.32	0.09
FPR (↓)	0.23	0.25

Table 15: Universal Verifier's agreement with human labels on the `Browserbase-OM2W` dataset (n=106) trajectories of Fara-7B on Online-Mind2Web, each labeled by two independent annotators). In the *UV-Blind* stage, annotators judged outcome and process success without seeing the UV's output; in the *UV-Informed* stage, annotators were shown the UV's verdict and asked whether they agreed. Outcome human labels are aggregated as majority vote; process human labels are the median rubric score binarized at ≥ 0.8.

	UV-Blind		UV-Informed	
	% Agree	κ	% Agree	κ
Outcome (binary)	79.6	0.57	82.5	0.53
Process (binary)	72.8	0.45	72.8	0.40
Process (score ≥ 0.8)	68.9	0.36	—	—

Table 16: Inter-annotator agreement on 103 Browserbase-OM2W tasks with two raters. Process labels show consistently lower agreement than outcome labels, reflecting the greater subjectivity of process evaluation. The continuous process scores have Pearson $r = 0.62$ and MAE $= 0.21$.

	Internal Dataset (n=140)			**Browserbase OM2W** (n=106)		
	WebVoy. (GPT-4o)	**WebJudge** (o4-mini)	**UV** (GPT-5.2)	**WebVoy.** (GPT-4o)	**WebJudge** (o4-mini)	**UV** (GPT-5.2)
Agreement with outcome human labels						
Accuracy (↑)	0.67 ± 0.01	0.72 ± 0.01	0.81 ± 0.02	0.48 ± 0.01	0.64 ± 0.02	0.88 ± 0.00
F1 (↑)	0.73 ± 0.01	0.74 ± 0.00	0.81 ± 0.02	0.35 ± 0.00	0.44 ± 0.02	0.65 ± 0.03
Cohen's κ (↑)	0.31 ± 0.01	0.44 ± 0.01	0.64 ± 0.03	0.13 ± 0.01	0.26 ± 0.03	0.58 ± 0.04
FNR (↓)	0.24 ± 0.01	0.33 ± 0.01	0.32 ± 0.03	0.12 ± 0.00	0.12 ± 0.05	0.31 ± 0.07
FPR (↓)	0.45 ± 0.01	0.22 ± 0.02	0.01 ± 0.01	0.60 ± 0.01	0.40 ± 0.02	0.08 ± 0.01
Agreement with process human labels						
Accuracy (↑)	0.62 ± 0.01	0.66 ± 0.01	0.81 ± 0.01	0.55 ± 0.01	0.68 ± 0.01	0.78 ± 0.01
F1 (↑)	0.70 ± 0.01	0.70 ± 0.01	0.86 ± 0.01	0.47 ± 0.00	0.53 ± 0.02	0.57 ± 0.02
Cohen's κ (↑)	0.17 ± 0.01	0.32 ± 0.02	0.59 ± 0.03	0.22 ± 0.01	0.34 ± 0.02	0.43 ± 0.03
FNR (↓)	0.31 ± 0.01	0.40 ± 0.01	0.24 ± 0.01	0.05 ± 0.00	0.12 ± 0.04	0.29 ± 0.02
FPR (↓)	0.52 ± 0.00	0.25 ± 0.03	0.04 ± 0.01	0.56 ± 0.01	0.38 ± 0.01	0.20 ± 0.01

Table 17: Agreement between three verifiers and human labels on two datasets. All values are mean $\pm$ std over 3 independent runs. *Internal Dataset* are internally annotated trajectories while *Browserbase OM2W* are Fara-7B trajectories from Online-Mind2Web with UV-informed human labels aggregated across two annotators per task by majority vote (outcome) or median rubric score binarized at ≥ 0.8 (process); see §5. For the UV, outcome uses the binary outcome signal and process uses the rubric score binarized at ≥ 0.8. WebVoyager and WebJudge each produce a single binary prediction compared against both label types.

Comparison	$\Delta\kappa$ (UV − base)	95% CI	bootstrap p
Internal: UV vs WJ	+0.40	$[+0.27, +0.55]$	< 0.001
Internal: UV vs WV	+0.29	$[+0.14, +0.44]$	< 0.001
BB: UV vs WJ	+0.26	$[+0.02, +0.50]$	0.020
BB: UV vs WV	+0.22	$[-0.01, +0.43]$	0.028
Pooled: UV vs WJ	+0.40	$[+0.27, +0.54]$	< 0.001
Pooled: UV vs WV	+0.29	$[+0.16, +0.43]$	< 0.001

Table 18: Agreement gains ($\Delta\kappa$) of UV over both baseline judges (each configued with GPT-5.2 backbone), with bootstrap confidence intervals and significance tests. Pooled is both subsets combined (N = 106 + 140).

	Success	**Fail**	**Total**	**Success Rate**
Over-budget (truncated)	44	663	707	6.2%
Terminated (agent stopped)	312	283	595	52.4%

Table 19: Partitions AgentRewardBench's 1302 human annotated trajectories based on its relation to the step budget and human-annotated success

Run	Purpose	Decision
0	Baseline	BASELINE
1	Outcome verification fixes	ROLLED BACK (process FPR 8.82%)
2	Semantic precision + entity non-existence + nitpick calibration	COMMITTED
3	Variant/tier + binding examples + CP rule	ROLLED BACK (process FPR 5.88%)
4	Similar to run 3, different approach	ROLLED BACK (process FPR 5.88%)
5	Binding example matching	ROLLED BACK (outcome FPs=2)
6	Rubric score context code change	COMMITTED
7	CP output + multi-item cart + info non-existence + superlative check	ROLLED BACK (kappa worse)
8	CP output + multi-item cart + info non-existence (no superlative)	ROLLED BACK (kappa worse)
9	No changes — stochastic baseline measurement	Confirmed run 6 was lucky ($\kappa = 0.6407$)
10	Same as run 8 (re-applied after baseline calibration)	COMMITTED
11	Rubric consistency + expanded example_match_check + lower cart threshold + colloquial terms	COMMITTED

Table 20: Summary of auto-research agent iterations continuing from the human expert's best verifier. Each run represents a single prompt modification cycle. Green rows were committed (improved κ without increasing FPR), red rows were rolled back, and yellow indicates a stochastic baseline check.

Run	Change Type	What the Agent Did	Why It Helped
2	Nitpick calibration (prompt)	Added explicit test: "Would a reasonable user say this output is useful?" Enumerated always-nitpick scenarios (approximate walk times, price tier symbols, common knowledge inferences).	Fixed 10+ false negatives where minor issues were treated as critical failures.
2	Semantic precision in rubric generation (prompt)	Added rule: criteria must test the *exact* concept the task asks about, not a related one. E.g., "how many people work remotely" $\neq$ "how many remote job postings."	Fixed false positives from rubrics testing the wrong quantity.
6	Rubric score context (code)	Computed normalized rubric score and appended calibration guidance to the outcome prompt. If rubric $\geq$ 95%, verifier must identify a *specific* critical issue to override.	Most impactful single change: provided quantitative signal instead of adding more text to an already-long prompt.
10	Critical point output rule (prompt)	When screenshots confirm the agent reached a transaction boundary (checkout, passenger info page) with correct selections, a brief output message is a nitpick, not grounds for failure.	Fixed persistent false negatives on booking/flight tasks where the agent correctly stopped but didn't restate details.
11	Forced rule checking (prompt)	Expanded the mandatory `example_match_check` JSON field to require the LLM to also check named rules (Entity Non-Existence, Multi-Item Cart, Critical Point Output, etc.) before making its verdict.	Mitigated the "rules exist but aren't applied" problem in $\sim$1,800-line prompts.

Table 21: Representative prompt and code changes made by the auto-research agent across its iterations. Changes span prompt engineering (calibration rules, forced structured checking) and code modifications (injecting rubric scores as quantitative context).