BEYOND FINAL ANSWERS: EVIDENCE-GROUNDED EVALUATION OF AGENT TRAJECTORIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Evaluating an agent trajectory is difficult when correctness cannot be determined from a final string alone: relevant evidence may be distributed across the execution trace, task-time skills, generated files, and executable behavior. We study how such evidence should be exposed to an evaluator and whether a single evaluation pipeline can adapt to different diagnostic targets through a declarative output schema. Our pipeline combines a MetaAgent, a small set of task-specific micro-judges, deterministic checks, and a structured OutcomeJudge, and supports both optional evidence retrieval and direct materialization of evidence into the shared context. On the full 203-case SkillTV split, evidence-delivery policy substantially changes evaluation behavior: materializing task-time skills yields much higher pass recall than output-only context (70.21% vs. 30.43%), while jointly materializing skills and outputs gives 59.9% accuracy among the matched Gemini-2.5-Flash configurations. A vanilla Gemini-2.5-Flash judge given the same evidence reaches only 44.6% accuracy but a higher F1 score (42.0% vs. 30.8%), showing that evaluator architecture changes how the same evidence is converted into a decision. On a 52-run collection derived from agent executions on AFTER, our Gemini-2.5-Flash pipeline raises pass recall to 83.7% (90.1% F1), compared with 30.6% (46.9% F1) for our reproduction of JudgeSkill v4 with Claude Sonnet 4.6; with the same Sonnet 4.6 backbone, our pipeline reaches 53.1% (69.3% F1). Finally, we apply the same core evaluation architecture to Who&When and TRAIL using benchmark-specific output schemas. On the handcrafted Who&When split, it matches TraceElephant in agent attribution accuracy (56.9%) while improving step accuracy from 1.72% to 12.07%, although agent attribution on the algorithm-generated split remains below both baselines; on the TRAIL GAIA split, a larger micro-judge budget raises Category F1 from 15.40% to 27.75%, exceeding the native TRAIL judge's 24.16%, although fine-grained localization remains substantially harder. The results separate three factors that are often conflated in agent evaluation: which evidence is available, how it is exposed and aggregated, and how evaluation criteria are specified across tasks. Code and data are available at `https://anonymous.4open.science/r/codegen-agent-eval-6744`.

1 INTRODUCTION

Agent evaluation increasingly operates in settings where the "answer" is not a single textual string. An agent may create a presentation, spreadsheet, report, code patch, dataset, or directory of files. In other settings the run is already known to have failed, and the question is which agent or trace step caused the failure. In both cases the evaluator has to reason over the execution evidence as well as the final response. That evidence is heterogeneous: the trace records what the agent attempted, task-time skills encode the procedures and constraints that shaped the run, output files show what was actually produced, and a sandbox can test executable claims. Each source can resolve a different ambiguity, and no single one should be assumed to dominate across tasks.

This paper focuses on a less obvious design choice: *how that evidence reaches the judge*. A natural agentic evaluator exposes skills and outputs as optional tools and lets the model decide whether to call them. On SkillTV, this choice changes the outcome. With skill and output tools available to the micro-judges, pass recall is 63.83% and pass F1 is 34.48%; when the task-time skill is instead

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

materialized into the input shared by all pipeline stages, pass recall reaches 70.21% and pass F1 40.24%. Materializing output content alone gives higher raw accuracy (54.46%) but much lower pass recall (30.43%), and materializing both gives the highest raw accuracy among the matched runs (59.90%). Skills and outputs are therefore complementary, and evidence-delivery policy is itself part of evaluator design.

The two evidence types do different work. Skill evidence is especially useful for understanding what the agent was expected to do and for avoiding false rejections of successful runs. Output evidence is especially useful when the result cannot be inferred from a completion message: the evaluator can check whether a required file exists, whether its saved contents match the task, and whether the artifact contradicts the trace. An evaluator does not have to choose between "skill-aware" and "artifact-aware" evaluation: it can expose both kinds of evidence in a controlled way and let task-specific checks use them for different purposes.

We implement this idea in a modular verifier. A MetaAgent selects one to two specialized micro-judges, while deterministic validators cover cheap, recognizable failures. In tool-based runs, selected micro-judges retrieve task-time skills and produced files on demand, and a tool-less OutcomeJudge aggregates their quoted evidence together with the trace. In forced-context runs, skill documents and/or a bounded output representation are appended to the trace itself, so the evidence is visible to routing, micro-judging, and final aggregation. The two modes differ both in whether evidence must be requested and in which stages see it, and we compare them as complete system configurations.

A second goal is transferability. SkillTV and our AFTER-derived collection (Belikova et al., 2026) ask for a binary verdict, whereas Who&When asks which agent and step caused a failure (Zhang et al., 2025), TraceElephant studies attribution under fuller observability (Chen et al., 2026a), and TRAIL annotates error locations and categories (Deshpande et al., 2025). We keep the evaluation architecture fixed and specify benchmark-specific output fields and criteria through a declarative output schema, which conditions judge selection and aggregation without benchmark-specific training or prompt optimization.

We study three questions: how evidence-delivery policy and evidence type change binary judging on SkillTV (RQ1); whether the same schema-driven evaluator transfers to failure attribution and error localization (RQ2); and whether it transfers to a separate, artifact-rich collection of agent runs on AFTER (RQ3). Abstention is an exploratory extension (RQ4). Our contributions are:

- We treat *evidence-delivery policy* as an explicit evaluator design variable and show on SkillTV that optional retrieval and materialization lead to different error profiles, with complementary roles for skills (higher pass recall, F1, and balanced accuracy) and outputs (higher true-negative rate and raw accuracy).
- We build an evidence-grounded evaluator with one to two routed micro-judges, deterministic validators, scoped evidence tools, and optional E2B execution, and show that it converts the same evidence into different decisions than single-call LLM judges.
- We show schema-driven transfer to Who&When attribution and TRAIL-GAIA error localization and categorization, using the same core evaluation architecture with benchmark-specific output schemas.
- On the AFTER-derived collection, our pipeline outperforms our reproduction of JudgeSkill v4 both with a cheaper backbone and under a backbone-matched comparison.

2 Related Work

2.1 Trajectory and agentic evaluation

Agent-as-a-Judge evaluates agent systems with an agentic evaluator that can inspect intermediate work and hierarchical requirements (Zhuge et al., 2025). Related designs specialize or generate the judges themselves: AutoPumpkin evaluates multi-agent systems with a collective of specialized agent-level and system-level evaluators (Zhidkovskaya et al., 2026b), while AutoJudge automatically generates LLM-based judges from execution traces (Zhidkovskaya et al., 2026a). SkillTV-Bench is the closest benchmark to our initial setting: it evaluates complete skill-augmented execu-

tions with task instructions, normalized trajectories, task-time skills, and inspectable artifacts, and proposes a reusable JudgeSkill that guides targeted inspections before a binary verdict (Han et al., 2026). We use its fixed held-out 203-case evaluation split for the main evidence-policy study.

2.2 Failure attribution and trace diagnosis

Who&When formulates automated failure attribution for LLM multi-agent systems: given a failed trace, a judge must identify the failure-responsible agent and decisive error step. The dataset contains 184 failure-annotation tasks drawn from 127 multi-agent systems, and the original work compares all-at-once, binary-search, and step-by-step attribution strategies (Zhang et al., 2025). TraceElephant argues that failure attribution should be evaluated under full execution observability and releases reproducible environments together with complete traces (Chen et al., 2026a). TRAIL instead annotates error spans and categories over 148 traces from GAIA and SWE-Bench, making the judge solve localization and diagnosis rather than only end-to-end outcome classification (Deshpande et al., 2025). We use these benchmarks to test whether our evaluator architecture can transfer across different trace schemas and output contracts.

2.3 Skill-aware and evidence-aware evaluators

Skill-centric evaluation studies the relationship between reusable procedural context and agent behavior. SkillJuror analyzes how skill organization affects runtime behavior (Chen et al., 2026c); SkillLearnBench evaluates generated skills at skill-quality, trajectory, and task-outcome levels (Zhong et al., 2026); and Skill-RM frames reward evaluation as an agentic process that selects and aggregates heterogeneous evidence (Chen et al., 2026b). Our experiments separate a related but distinct question: even when skill information is available, does the evaluator benefit from receiving it directly rather than deciding whether to retrieve it?

2.4 Abstention and selective judging

AbstentionBench studies whether LLMs recognize when they should avoid definitive answers (Kirichenko et al., 2025). Judge, Retrieve, or Abstain develops uncertainty-guarded judging with explicit risk control (Badshah et al., 2026). We keep abstention outside the main method: our unknown-enabled variant improves quality on decided cases but abstains too frequently to serve as the default evaluation rule.

3 Method

Figure 1 summarizes the evaluator. For outcome verification, a case is $x = (I, \tau, S, A)$, where I is the task instruction, τ the execution trace, S the task-time skills available to the agent, and A the produced artifacts; diagnostic benchmarks replace the binary label with annotations such as an error span, responsible agent, or error category. The trace explains the attempted process, skills provide normative context for what the agent should have done, artifacts record the delivered state, and sandbox checks provide behavioral evidence for executable tasks.

Routing, micro-judges, and aggregation. A MetaAgent receives the task, the trace, and the output schema and selects specialized judges from a fixed built-in registry, under a budget of one to two micro-judges per trajectory in the main configuration. In practice it almost always saturates the budget, so the upper bound acts as the compute budget. Each micro-judge covers one failure dimension and returns structured findings with cited evidence, severity, and confidence; deterministic validators run alongside them for recognizable failure patterns. The OutcomeJudge aggregates the findings into the schema-defined output and is deliberately tool-less. An optional Skeptic pass re-examines candidate successful trajectories for overlooked failure evidence.

3.1 Evidence-delivery policies

Optional retrieval. Selected micro-judges can call scoped tools such as `read_skill` or `read_output_file`; the latter enumerates the output tree, reads text files, and converts common

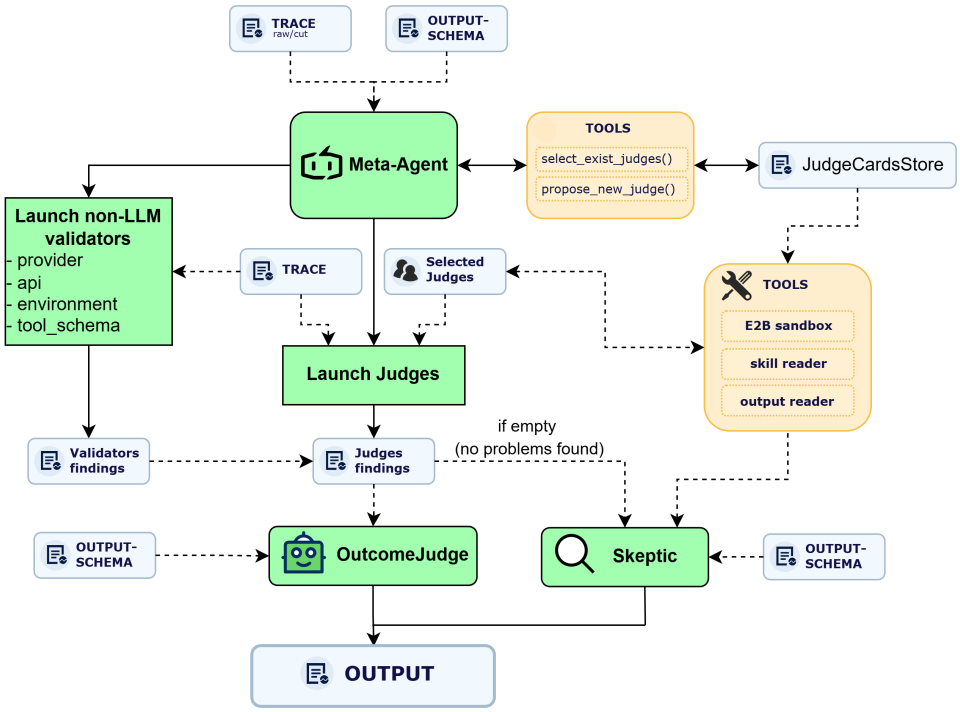


Figure 1: Overview of the evaluation pipeline. The output schema specifies the required decision fields and task-specific evaluation criteria. The MetaAgent uses the trace and schema to select specialized micro-judges, which can inspect additional evidence through scoped tools. Their findings are aggregated by the OutcomeJudge, with an optional Skeptic pass for additional verification. The `propose_new_judge()` tool is disabled in all reported experiments (Appendix A).

binary document formats (PDF, DOCX, PPTX, XLSX, Parquet) to bounded textual views. E2B execution is available only through a dedicated `code_reproducibility` micro-judge. This design is flexible but creates an acquisition failure mode: a relevant source may never be requested.

Forced materialization. The runner appends the full task-time skill documents and/or a bounded listing and text representation of the output package directly to the trace. The materialized evidence is therefore visible to the MetaAgent, the micro-judges, and the OutcomeJudge, and the read tools are disabled. The forced-output serializer inlines readable text files up to fixed limits and lists binary files without converting them. The two policies therefore differ in whether evidence must be requested, which stages see it, and, for binary artifacts, its representation; we compare them as complete system configurations. Artifact evidence matters most for tasks with no canonical final string: a trace can claim that a file was created, but only the saved artifact can establish whether the deliverable exists and contains the required content. The artifact, in turn, may not reveal procedural constraints, which is why skills and traces remain complementary.

3.2 Trace representation and schema-driven adaptation

Long traces contain repeated tool observations and duplicated context. We compare the raw trace, a lossy head–tail truncation, and lossless citation-based deduplication, in which repeated content is replaced by compact references (Appendix A). Deduplication performs best (Table 2): it improves accuracy by 4.44 points over the raw trace and by 4.93 points over head–tail truncation, and gives the best balanced accuracy and pass F1. Since keeping only the beginning and end of a trace does not recover the same performance, evidence from the middle of a trajectory appears to matter. Deduplication is used in all main SkillTV experiments.

Table 1: Datasets and roles in our study. For the AFTER-derived collection, we use the 52 of 53 packages for which all evaluators returned a valid judgment (Section 5.3).

Dataset	Cases	Labels	Role
SkillTV held-out eval	203	47 pass / 156 fail	main evidence-policy study
AFTER-derived collection	52	49 pass / 3 fail	agent-run transfer
Who&When	184	responsible agent, decisive step	attribution transfer
TRAIL (GAIA split)	GAIA split	native TRAIL annotations	localization/category transfer

Table 2: Matched trace-preprocessing ablation on the 203-case SkillTV split (Gemini-2.5-Flash, one-to-two micro-judges, deterministic validators, trace-only evidence). F1/precision/recall are for the pass class.

Trace representation	N	Acc.	Bal. Acc.	Prec.	Recall	F1	TP	TN	FP	FN
Raw	203	38.42	48.79	22.54	68.09	33.86	32	46	110	15
Head–tail	203	37.93	46.24	21.17	61.70	31.52	29	48	108	18
Deduplicated	203	**42.86**	**53.16**	**24.82**	**72.34**	**36.96**	**34**	**53**	**103**	**13**

Evaluation requirements are supplied as a declarative output schema that specifies target fields, their types, and the criteria used to produce them. The schema is exposed to the MetaAgent and the OutcomeJudge, so checks are selected with the requested target in view, while routing, evidence access, micro-judges, and aggregation are unchanged across benchmarks. For SkillTV and AFTER the schema requests a binary verdict; for Who&When, the responsible agent, decisive step, and reason; for TRAIL, benchmark-native error-localization and error-category fields.

4 Experimental Setup

Datasets. SkillTV provides the main controlled setting: its full 203-case held-out split contains the task, trace, skill, and output channels needed for evidence-policy ablations, and all SkillTV results except the exploratory abstention study use this full split. AFTER is an agent benchmark rather than a benchmark for judges; we use 53 session/log packages produced by running agents on AFTER tasks, each with the execution log, task-time skills, and generated outputs, 52 of which are scored by all evaluators (Section 5.3). The binary labels are the pass/fail verdicts assigned to each run by the AFTER benchmark itself; the logs, skills, outputs, and labels are included in our anonymous repository. Who&When and TRAIL test transfer from outcome verification to trace diagnosis (Table 1); for TRAIL we report only the GAIA split, because the SWE-Bench split is too small for a meaningful standalone comparison.

Metrics and configuration. For SkillTV we report accuracy, balanced accuracy, pass-class precision/recall/F1, and the full confusion matrix; because the split is failure-heavy, raw accuracy is never interpreted alone. For AFTER we report accuracy with pass-class F1 and recall; for Who&When, responsible-agent and decisive-step accuracy; for TRAIL-GAIA, category F1 and localization accuracy. When some cases could not be scored because of evaluation or system errors, metrics are computed over the scored cases and the denominator is stated. The main SkillTV configuration uses deduplicated traces, the fixed judge registry, a one-to-two micro-judge budget, deterministic validators, binary aggregation, and Gemini-2.5-Flash; the evidence policy is the experimental variable. Skeptic and confidence rules are ablations. Cost is a secondary statistic (Appendix D). Extended experimental details, full Who&When results, and a summary of findings are given in Appendices B and C.

Table 3: SkillTV held-out results on the 203-case split (one–two judge budget). F1/precision/recall are for the pass class; N is the number of scored cases. †Accuracy and F1 as reported by SkillTV-Bench; on this split (47 pass, 156 fail) they determine the confusion matrix uniquely, and the other cells are derived from them.

Configuration	N	Acc.	Bal. Acc.	Prec.	Recall	F1	TP	TN	FP	FN
Trace only	203	42.86	53.16	24.82	72.34	36.96	34	53	103	13
Trace + optional skill/output tools	203	43.84	50.83	23.62	63.83	34.48	30	59	97	17
Trace + optional skill/output tools + Skeptic	203	51.72	51.49	24.24	51.06	32.88	24	81	75	23
Trace only; Gemini judges + Sonnet 4.6 Outcome	199	33.67	48.50	22.93	76.60	35.29	36	31	121	11
Trace only; Gemini judges + Sonnet 4.6 Outcome; conf.=0.9	199	64.32	49.45	22.73	21.28	21.98	10	118	34	37
Trace + forced skill	203	51.72	58.18	28.21	70.21	40.24	33	72	84	14
Trace + forced skill; conf.=0.9	203	53.20	59.14	28.95	70.21	40.99	33	75	81	14
Trace + forced skill + Skeptic	203	51.72	58.18	28.21	70.21	40.24	33	72	84	14
Trace + forced output	202	54.46	45.99	18.92	30.43	23.33	14	96	60	32
Trace + forced skill + forced output	202	59.90	52.58	25.35	39.13	30.77	18	103	53	28
+ deterministic skill check + Skeptic	202	58.42	51.62	24.32	39.13	30.00	18	100	56	28
Refined JudgeSkill (Sonnet 4.6)†	203	58.60	63.41	32.38	72.34	44.70	34	85	71	13

Table 4: Architecture ablation on SkillTV. All evaluators use Gemini-2.5-Flash and receive the same materialized trace, skill, and output evidence. Metrics are computed over successfully scored outputs; coverage may differ across evaluators.

Evaluator	Acc.	F1
Vanilla Gemini-2.5-Flash	44.6	**42.0**
PydanticAI LLMJudge	29.6	38.1
Our pipeline	**59.9**	30.8

Table 5: SkillTV judge-budget comparison with skill- and output-reading tools. One–two costs are measured; the others are our estimates. The one–ten Skeptic run invoked the Skeptic 80 times.

Budget	Skeptic	Acc.	F1	Cost
1–2	no	43.84	34.48	$4.69
1–2	yes	51.72	32.88	$5.20
1–10	no	59.61	35.94	$43.5
1–10	yes	61.08	32.48	$44.5
JudgeSkill v4	–	58.60	44.70	$40–45

5 EXPERIMENTS

5.1 RQ1: HOW DO EVIDENCE DELIVERY AND EVALUATOR ARCHITECTURE AFFECT SKILLTV JUDGING?

We evaluate the 203-case split with deduplicated traces while changing how skills and outputs reach the judge (Table 3). The optional-tool rows keep the one–two judge budget, which isolates evidence delivery from the judge-budget effect (Table 5). The mixed-backbone rows use Gemini-2.5-Flash micro-judges and a Claude Sonnet 4.6 OutcomeJudge. The two mixed-backbone rows were scored on 199 cases and the three forced-output rows on 202; we do not impute missing predictions.

The trace-only baseline makes the effect of each evidence channel easier to interpret: with trace alone, the evaluator retains high pass recall (72.34%) but produces 103 false positives and only 53 true negatives. When skill and output evidence are only available through optional tools under the same one–two-judge budget, the pipeline reaches 43.84% accuracy and 34.48% F1; adding the Skeptic increases accuracy to 51.72% while F1 is 32.88%. Forced skill preserves nearly the same pass recall (70.21%) while reducing false positives to 84, raising accuracy from 42.86% to 51.72% and balanced accuracy from 53.16% to 58.18%, which is the clearest evidence in our runs that task-time skills add information beyond the trace. Forced output changes the operating point differently: it raises true negatives to 96 but cuts pass recall to 30.43%. Combining both yields the highest raw accuracy among the matched Gemini rows (59.90%) and the fewest false positives (53), but does not dominate the trace-only or skill-only conditions on balanced accuracy or pass F1. The Skeptic leaves the forced-skill run unchanged and lowers accuracy in the combined configuration (59.90% to 58.42%).

Architecture ablation. To separate evidence availability from evaluator architecture, we give the same materialized evidence to a single vanilla Gemini-2.5-Flash call and to the `LLMJudge` evaluator from PydanticAI Evals[1], neither of which uses routing, micro-judges, validators, or OutcomeJudge

[1] `https://ai.pydantic.dev/evals/`

Table 6: Failure attribution on Who&When (Agent/Step accuracy). All evaluators use Gemini-2.5-Flash. Missing predictions count as wrong; all runs completed except one TraceElephant case on the algorithm-generated subset.

Evaluator	Method	Agent	Step
Handcrafted (58)			
TraceElephant	all-at-once	56.90	1.72
Who&When judge	all-at-once	58.62	3.45
Ours	schema-driven	56.90	12.07
Ours	agents-findings	74.14	39.66
Algorithm-generated (126)			
TraceElephant	all-at-once	62.70	9.52
Who&When judge	all-at-once	66.67	28.57
Ours	schema-driven	46.03	26.98
Ours	agents-findings	51.59	41.27

Table 7: TRAIL results on the GAIA split. All evaluators use Gemini-2.5-Flash.

Evaluator	Cat. F1	Loc. Acc.
TRAIL native judge	24.16	**17.05**
Ours, 1 - 2 judges	15.40	8.80
Ours, 3 - 7 judges	**27.75**	9.50

Table 8: Reference-span coverage on TRAIL (internal diagnostic, not the native metric), out of 585 reference spans.

Judges	Raw	Final	Conv.
1 - 2	13.3	8.4	62.8
3 - 7	**22.4**	**9.1**	40.5

aggregation (Table 4). Our pipeline reaches 59.9% accuracy, compared with 44.6% and 29.6%, while both single-call judges achieve higher F1 (42.0% and 38.1% vs. 30.8%). Access to the same evidence therefore does not by itself determine the judgment: the architecture that processes it materially changes the result.

Comparison to JudgeSkill. Refined JudgeSkill reaches 58.6% accuracy and 44.7% pass F1. Our forced skill+output row reaches slightly higher raw accuracy but lower pass F1 (30.77%) and balanced accuracy (52.58% vs. 63.41%). The mixed-backbone run with a 0.9 confidence rule reaches 64.32% accuracy with only 21.98% pass F1, illustrating why raw accuracy alone is misleading on this imbalanced split; the direct LLM judges reported by SkillTV-Bench show the same pattern (Appendix E). These are different operating points, and none is a blanket improvement over JudgeSkill.

Judge budget. With optional tools, raising the routing range from one–two to one–ten (in practice about two versus ten judges) raises accuracy from 43.84% to 59.61% without the Skeptic and from 51.72% to 61.08% with it (Table 5). The cost grows from $4.69–$5.20 to roughly $43.5–$44.5, in the range of our $40–$45 estimate for JudgeSkill v4 with Sonnet 4.6. We therefore keep the one–two budget as the default, so that the evidence-policy study does not conflate evidence delivery with scaling the number of judges.

5.2 RQ2: Does the evaluator transfer beyond binary outcome verification?

We apply the same core evaluation architecture to Who&When and TRAIL-GAIA using benchmark-specific output schemas. For Who&When, *schema-driven* scores the final OutcomeJudge prediction, while *agents-findings* measures whether the correct agent or step appears anywhere in the intermediate micro-judge findings. For TRAIL, we compare budgets of one to two and three to seven micro-judges with Gemini-2.5-Flash, and additionally measure whether the reference span appears in the intermediate findings.

On the handcrafted Who&When subset, our final prediction matches TraceElephant in agent accuracy (56.90%), slightly below the native judge (58.62%), and substantially improves step accuracy over both (12.07% vs. 1.72% and 3.45%). On the algorithm-generated subset, our agent accuracy (46.03%) is lower than that of both baselines, while our step accuracy (26.98%) exceeds TraceElephant (9.52%) and remains close to the native judge (28.57%). On TRAIL, the larger budget raises Category F1 from 15.40% to 27.75%, exceeding the native judge's 24.16%, while localization accuracy improves only from 8.80% to 9.50% and remains below the native judge. In both benchmarks the intermediate findings contain much more correct evidence than the final output: the agents-findings scores are far higher on Who&When, and on TRAIL the larger budget raises span coverage in the findings from 13.3% to 22.4% but in the final output only from 8.4% to 9.1%. The pipeline transfers across output schemas, while fine-grained attribution and localization are limited partly by final aggregation.

Table 9: Binary judging on the AFTER-derived collection (52 packages). F1 and recall are for the pass class. Best evaluator in bold; the constant predictor is a reference.

Evaluator	Backbone	Acc.	F1	Rec.
Always "pass"	–	94.2	97.0	100.0
JudgeSkill v4 (repr.)	Sonnet 4.6	34.6	46.9	30.6
Ours (tools)	Sonnet 4.6	55.8	69.3	53.1
Jev	Jev 1.13	63.5	75.9	61.2
Ours (tools)	Gemini 2.5 Flash	**82.7**	**90.1**	**83.7**

Table 10: Exploratory abstention study on the 57-case SkillTV subset with artifacts.

Rule	Cov.	Acc.	Bal.	Unk.	FP
Binary	1.000	31.58	43.77	0	35
Unknown	0.368	61.90	63.89	36	7

5.3 RQ3: Evaluation on an AFTER-derived agent-run collection

In the AFTER-derived collection, task success often cannot be determined from the execution log alone: the evaluator may need to inspect the skills available to the agent or the files actually produced. We compare our pipeline, with skills and outputs available through optional tools, against our own reproduction of JudgeSkill using the authors' released final v4 skill and Claude Sonnet 4.6; this is not an author-reported AFTER result. We also compare against Jev 1.13 (Almeida, 2026), a model for structured decisions that returns probabilities for predefined questions, which we run ourselves with its default configuration. Its context window is limited to 32k tokens, which does not accommodate SkillTV traces, so Jev is evaluated on the AFTER-derived collection only. We run our pipeline both with the same Sonnet 4.6 backbone and with the cheaper Gemini-2.5-Flash backbone used in our main experiments. Both Sonnet runs returned valid judgments for 52 of the 53 packages; to keep the comparison on identical inputs, all evaluators are scored on these 52 packages.

The collection is heavily skewed toward passing runs (49 of the 52 packages pass), so a constant "pass" predictor reaches 94.2% accuracy; we report it as a reference and focus on pass recall, since nearly all errors of every evaluator are false negatives.

With the same Sonnet 4.6 backbone, our pipeline improves accuracy from 34.6% to 55.8%, F1 from 46.9% to 69.3%, and pass recall from 30.6% to 53.1% (Table 9), with 100% precision (26 TP, 3 TN, 0 FP, 23 FN). Since the backbone is identical, this gain is attributable to the pipeline rather than to the judge model. Jev is stronger than both Sonnet-based evaluators (63.5% accuracy, 75.9% F1, 61.2% pass recall; 30 TP, 3 TN, 0 FP, 19 FN). With the cheaper Gemini-2.5-Flash backbone, our pipeline reaches 82.7% accuracy, 90.1% F1, and 83.7% pass recall (41 TP, 2 TN, 1 FP, 8 FN), outperforming Jev without requiring a stronger judge model.

5.4 RQ4 (exploratory): Is an **unknown** verdict useful?

We compare a binary run with an unknown-enabled run on stripped traces of the 57 SkillTV cases with artifacts (Gemini-2.5-Flash, one-to-two micro-judges, validators, trace and output evidence; Table 10). Enabling abstention reduces coverage to 36.8% while raising accuracy on decided cases from 31.6% to 61.9% and balanced accuracy from 43.8% to 63.9%; false passes drop from 35 to 7.

Two observations follow. First, abstention converts errors into abstentions rather than into correct verdicts: most of the coverage given up by the unknown-enabled variant consists of cases the binary pipeline was deciding incorrectly, which is why selective accuracy rises as coverage falls. Second, the abstentions are concentrated where the available evidence cannot settle the verdict: confirming a pass requires a comparison against ground truth that neither the trace nor the produced artifacts contain, while fail verdicts survive only when a violation is directly observable. We therefore view **unknown** not as a competing decision rule but as a routing signal: it identifies the subset of cases that require stronger evidence (executable checks or reference answers) than the evaluator was given.

6 Discussion and Limitations

Skills and outputs have different error profiles. The SkillTV results do not support a universal ranking of evidence channels. Skill context has stronger pass recall, F1, and balanced accuracy than

output-only context, whereas output-only context has more true negatives and higher raw accuracy. This may reflect genuinely complementary information or a conservatism bias induced by artifact representations.

Artifact access is not full artifact understanding. The optional `read_output_file` path exposes semantic content, tables, slide text, file existence, and basic structure for common document formats, but does not assess visual layout, chart rendering, formatting fidelity, or interactive behavior. The forced-inline path is weaker for binary documents because it lists them without conversion. Moreover, forced and optional conditions differ both in artifact serialization and in which pipeline stages see the evidence; a hybrid condition that keeps the read tools alongside forced context, and a variant that equalizes stage visibility, would isolate these factors and are not covered by our experiments.

Cross-benchmark transfer changes the target. Who&When and TRAIL-GAIA are not binary success benchmarks; their inclusion tests architectural portability, not metric comparability with SkillTV. Apart from the diagnostic span-coverage analysis on TRAIL, we report benchmark-native metrics.

Class imbalance. SkillTV has more than three times as many failures as passes (156 vs. 47), and the combined skill+output configuration raises raw accuracy primarily through true negatives. The AFTER-derived collection is strongly pass-skewed (49 pass, 3 fail among the 52 scored packages), so its pass-class metrics are dominated by the majority class and a constant "pass" predictor reaches 94.2% accuracy. We therefore report accuracy together with class-aware metrics and confusion counts.

Skeptic and abstention. With optional tools, the Skeptic raises accuracy but lowers pass F1; it leaves the forced-skill run unchanged and is slightly negative in the combined configuration. The unknown-enabled variant trades a large loss of coverage for higher quality among decided cases. Both therefore remain optional components rather than part of the main method.

Cost figures are partly estimated. Only the one–two-judge costs are measured; the one–ten and JudgeSkill figures are our estimates (Appendix D).

7 Conclusion

We study agent evaluation as a problem of acquiring and combining execution evidence rather than scoring a final answer in isolation. Materializing task-time skills produces much stronger pass recall than output-only context, output artifacts increase the rejection of failed runs, and combining both yields the highest raw accuracy in our system, so evaluator design should specify how, and to which stages, evidence is delivered, not only which evidence is available. The same core evaluation architecture, used with benchmark-specific output schemas, transfers to agent runs on AFTER and to failure attribution and error localization. The study thus separates evidence usefulness, evidence delivery, and evaluator portability – three dimensions that are easy to conflate in monolithic LLM-as-a-judge systems.

AI use statement

We used generative AI assistants, such as Codex and Claude Code, during the research and writing process. They were used to assist with brainstorming and refining experimental design, inspecting and debugging code, interpreting experimental results, searching for and summarizing related work, and drafting, restructuring, and editing parts of the manuscript. Generative AI was also used for formatting tables and improving the clarity of technical descriptions.

All AI-assisted research decisions, code changes, numerical results, citations, and manuscript claims were reviewed by the authors. Experimental results reported in the paper were obtained from the implemented evaluation pipelines and benchmark runs rather than generated by the writing assistant. Citations and factual claims suggested by AI tools were independently checked against their original

sources. The authors take full responsibility for the content of the paper and for all AI-assisted material included in the submission.

The use of LLMs as judges, micro-judges, or evaluated components in our experiments is part of the proposed methodology and is described separately in the corresponding experimental sections; it is distinct from the use of generative AI as a research and writing assistant.

Reproducibility statement

The code is in an anonymous repository at `https://anonymous.4open.science/r/codegen-agent-eval-6744`; Appendix F lists what it contains, including the benchmark data with traces, task-time skills, and output files, and the code and configurations for every reported run, including the SkillTV preprocessing ablation, the main SkillTV configurations, and both AFTER comparisons. The evidence-delivery policies and trace representations are specified in Section 3 and Appendix A, and cost figures in Appendix D. Results from hosted LLMs can change across model versions; the backbone for each result is stated in the corresponding table or section.

References

Diogo Almeida. Introducing system one models & jev, September 2026. URL `https://typesafe.ai/blog/introducing-system-one-models-and-jev`. TypeSafe AI.

Sher Badshah, Ali Emami, and Hassan Sajjad. Judge, retrieve, or abstain: Uncertainty-guarded LLM judging with provable risk guarantees. In *Conference on Language Modeling (COLM)*, 2026.

Julia Belikova, Rauf Parchiev, Evgeny Egorov, Grigorii Davydenko, Gleb Gusev, Andrey Savchenko, and Maksim Makarenko. Managing procedural memory in LLM agents: Control, adaptation, and evaluation. *arXiv preprint arXiv:2606.23127*, 2026.

Mengzhuo Chen, Junjie Wang, Fangwen Mu, Yawen Wang, Zhe Liu, Huanxiang Feng, and Qing Wang. Seeing the whole elephant: A benchmark for failure attribution in LLM-based multi-agent systems. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2026a.

Tao Chen, Gangwei Jiang, Pengyu Cheng, Siyuan Huang, Yihao Liu, Jingwei Ni, Jiaqi Guo, Mengyu Zhou, Kai Tang, Junling Liu, Qinliang Su, Xiaoxi Jiang, and Guanjun Jiang. Skill-RM: Unifying heterogeneous evaluation criteria via agent skill. *arXiv preprint arXiv:2606.03980*, 2026b.

Zhiyu Chen, Zihan Guo, Bo Huang, Bingwei Lu, Jianghao Lin, Yuanjian Zhou, and Weinan Zhang. SkillJuror: Measuring how agent skill organization changes runtime behavior. *arXiv preprint arXiv:2606.11543*, 2026c.

Darshan Deshpande, Varun Gangal, Hersh Mehta, Jitin Krishnan, Anand Kannappan, and Rebecca Qian. TRAIL: Trace reasoning and agentic issue localization. *arXiv preprint arXiv:2505.08638*, 2025.

Zhi Han, Chenxi Zeng, Liuhaichen Yang, Zihan Guo, Ming Zhou, and Yang Li. SkillTV-Bench: Benchmarking how well judges perform on skill-augmented agentic execution. *arXiv preprint arXiv:2608.05573*, 2026.

Polina Kirichenko, Mark Ibrahim, Kamalika Chaudhuri, and Samuel J. Bell. AbstentionBench: Reasoning LLMs fail on unanswerable questions. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2025.

Shaokun Zhang, Ming Yin, Jieyu Zhang, Jiale Liu, Zhiguang Han, Jingyang Zhang, Beibin Li, Chi Wang, Huazheng Wang, Yiran Chen, and Qingyun Wu. Which agent causes task failures and when? on automated failure attribution of LLM multi-agent systems. In *International Conference on Machine Learning (ICML)*, 2025. Spotlight.

Alina Zhidkovskaya, Kirill Rapatskikh, Grigorii Barakhsin, Jerzy Kaminski, Alexander Serebryakov, Roman Dolgoshei, Julia Belikova, Maksim Makarenko, Andrey Savchenko, Anna V. Kalyuzhnaya, and Nikolay O. Nikitin. AutoJudge: Automatic generation of LLM-based judges from execution traces. In *Failure Modes in Agentic AI (FAGEN) Workshop at the International Conference on Machine Learning (ICML)*, 2026a.

Alina Zhidkovskaya, Kirill Rapatskikh, Jerzy Kaminski, Grigorii Barakhsin, Anna V. Kalyuzhnaya, Alexey Druzhinin, Andrey Savchenko, Julia Belikova, Konstantin Polev, and Nikolay O. Nikitin. Towards design of an automated judge for multi-agent systems. In *Trustworthy Agentic AI Workshop at the AAAI Conference on Artificial Intelligence*, 2026b.

Shanshan Zhong, Yi Lu, Jingjie Ning, Yibing Wan, Lihan Feng, Yuyi Ao, Leonardo F. R. Ribeiro, Markus Dreyer, Sean Ammirati, and Chenyan Xiong. SkillLearnBench: Benchmarking continual learning methods for agent skill generation on real-world tasks. *arXiv preprint arXiv:2604.20087*, 2026. Accepted at COLM 2026.

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

A Additional method details

Trace representations. *Raw* keeps the original serialized trace. *Head–tail* is a lossy truncation baseline that retains a prefix and suffix of the trajectory while dropping the middle. *Dedup* applies lossless citation-based deduplication: repeated material is replaced by compact references while the original content remains reconstructable. *Stripped* is a reduced trace variant used in the exploratory abstention study; its full-dataset runs differ in downstream model configuration and scored denominator, so we do not include it in the matched preprocessing ablation.

Routing and micro-judges. With a one–two range the MetaAgent typically selects two judges, and with a one–ten range it typically selects ten. Generated judges are disabled in all experiments, so that the study measures routing and evidence exposure rather than prompt synthesis. Each micro-judge returns a structured `FindingsResult`, and selected LLM micro-judges execute in parallel. The MetaAgent is informed through the judge registry that evidence tools exist, but does not read their contents.

Evidence tools and forced materialization. In optional-retrieval runs, `read_output_file` converts PDF pages, DOCX paragraphs, PPTX slide text, XLSX/XLSM sheet values, and Parquet schemas with row previews to bounded textual views. In forced-context runs, the runner rewrites the trace with the materialized evidence and then sets the evidence preset back to `trace`, so `read_skill` and `read_output_file` are not granted to the LLM judges. Because the forced-output serializer does not apply the document converters to binary files, the forced-output result changes both the delivery policy and the accessible representation of binary artifacts.

B Extended experimental details

Datasets. Table 11 extends Table 1 with the evidence available in each dataset. AFTER itself is an agent benchmark, not a benchmark for judges; the value of the AFTER-derived collection for this paper is that each execution package preserves the concrete deliverables and task-time evidence needed for artifact-level analysis.

Table 11: Datasets and roles in our study, with the evidence available in each. "Cases" denotes the units scored by our pipeline; Who&When and the TRAIL GAIA split use their native diagnostic labels rather than binary pass/fail.

Dataset	Cases	Label statistics	Available evidence	Role
SkillTV held-out eval	203	47 pass / 156 fail	trace, skills, outputs	main evidence-policy study
AFTER-derived session/log packages	52	49 pass / 3 fail	log, skills, outputs	agent-run trajectory collection
Who&When	184 tasks	failed trajectories with who/when labels	failure logs + system info	attribution transfer
TRAIL (GAIA split)	GAIA subset	native TRAIL annotations	OpenTelemetry traces	localization/category transfer

Metrics. For SkillTV we report accuracy, balanced accuracy, pass-class precision/recall/F1, and the complete confusion matrix. Because SkillTV is failure-heavy, raw accuracy is never interpreted alone. For the strongly pass-skewed AFTER-derived collection, the main comparison reports accuracy together with pass-class F1 and recall. Where some cases could not be scored because of evaluation or system errors, metrics are computed over the successfully scored cases and the denominator is stated.

For Who&When we report benchmark-native responsible-agent accuracy and decisive-step accuracy. For TRAIL we report category F1 and localization accuracy on the GAIA split. These are not collapsed into a single cross-benchmark score because the targets differ.

Inference cost is recorded as a secondary deployment statistic. The exploratory unknown study reports coverage, accuracy and balanced accuracy on decided cases, and false passes, rather than treating unknown as a third source label.

Comparison to JudgeSkill on SkillTV. The comparison to refined JudgeSkill must be equally careful. On the 203-case SkillTV split, JudgeSkill reaches 58.6% accuracy and 44.7% pass F1. Our forced skill+output row reaches a slightly higher raw accuracy among its scorable outputs, but lower pass F1 (30.77% vs. 44.7%). The mixed-backbone trace-only run reaches 64.32% raw accuracy, but does so with only 21.28% pass recall and 21.98% pass F1, illustrating why raw accuracy alone is misleading on this imbalanced split. These configurations are different operating points, and none is a blanket improvement over JudgeSkill. The direct LLM judges reported by SkillTV-Bench (Appendix E) show the same pattern: GPT-5.2 has the highest accuracy among them (48.3%) and the lowest pass F1 (29.5%).

Who&When. Table 12 reports the full Who&When results, including the number of completed and failed runs for each evaluator.

Table 12: Full Who&When results, including completed and failed runs. *schema-driven* evaluates the final prediction; *agents-findings* is a softer diagnostic of whether the correct attribution appears in any intermediate finding. Missing predictions count as wrong.

Dataset	Evaluator	Method	Agent Acc.	Step Acc.	Completed	Failed
Handcrafted subset						
Handcrafted	TraceElephant	all-at-once	56.90	1.72	58	0
Handcrafted	Who&When native judge	all-at-once	58.62	3.45	58	0
Handcrafted	Our pipeline	schema-driven	56.90	12.07	58	0
Handcrafted	Our pipeline	agents-findings	74.14	39.66	58	0
Algorithm-generated subset						
Algorithm	TraceElephant	all-at-once	62.70	9.52	125	1
Algorithm	Who&When native judge	all-at-once	66.67	28.57	126	0
Algorithm	Our pipeline	schema-driven	46.03	26.98	126	0
Algorithm	Our pipeline	agents-findings	51.59	41.27	126	0

Binary transfer against JudgeSkill v4. We compare our pipeline against a reproduction of the SkillTV JudgeSkill baseline on the same AFTER-derived collection. This is not an author-reported result on AFTER: we run JudgeSkill ourselves using the authors' released final JudgeSkill v4 and Claude Sonnet 4.6.

To control for backbone choice, we evaluate our pipeline with the same Claude Sonnet 4.6 backbone. In this configuration, skills and generated outputs are available through optional tools. We also report the corresponding Gemini-2.5-Flash configuration, which uses the substantially cheaper backbone employed in our main experiments. The two Claude Sonnet 4.6 evaluations produced valid judgments for 52 of the 53 session packages; all evaluators, including the Gemini-2.5-Flash run and Jev, are scored on these 52 packages (49 pass, 3 fail).

The model-matched comparison indicates that the improvement is not explained solely by backbone choice. With Claude Sonnet 4.6, our pipeline increases accuracy from 34.6% to 55.8%, F1 from 46.9% to 69.3%, and pass recall from 30.6% to 53.1% relative to the reproduced JudgeSkill v4

baseline. The Sonnet configuration of our pipeline also achieves 100% precision on the scored examples (26/26 predicted positives), with 26 true positives, 3 true negatives, 0 false positives, and 23 false negatives.

Using Gemini-2.5-Flash further increases accuracy to 82.7%, F1 to 90.1%, and recall to 83.7% (41 TP, 2 TN, 1 FP, 8 FN). This result is notable because Gemini-2.5-Flash is also the cheaper backbone used in our main experiments. Together, the two comparisons suggest that the gain on the AFTER-derived collection is associated with the evaluation pipeline and its access to heterogeneous execution evidence, rather than requiring a stronger judge model. Because the collection is strongly pass-skewed, a constant "pass" predictor reaches 94.2% accuracy, so these results should be read through pass recall rather than raw accuracy.

C SUMMARY OF FINDINGS

The central finding is that the way evidence is delivered changes the decision profile. Compared with forced output alone, forced skill context more than doubles pass recall (70.21% versus 30.43%) and yields much higher pass F1 (40.24% versus 23.33%). Forced output produces a higher true-negative rate, which raises raw accuracy despite the loss in pass recall. The two channels appear to carry different information.

Combining forced skill and output context yields the best raw accuracy among the matched Gemini evidence-policy configurations, 59.90%, with TP=18, TN=103, FP=53, and FN=28 over 202 scored cases. This is close to the 58.6% accuracy of refined JudgeSkill, while JudgeSkill has higher pass F1 (44.7% versus 30.77%). Separately, the mixed-backbone trace-only run reaches 64.32% raw accuracy, but only 21.98% pass F1. Raw accuracy alone does not show that our judge outperforms JudgeSkill. What the numbers show is competitive raw accuracy at a different precision/recall operating point.

The Skeptic adds little in these runs. With forced skill context, adding the Skeptic leaves TP/TN/FP/FN unchanged. In the combined configuration, adding the deterministic skill check and Skeptic lowers raw accuracy from 59.90% to 58.42%. We therefore treat the Skeptic as an optional component rather than a contribution.

Finally, the evaluator carries over to agent runs outside SkillTV. On the 52 scored packages of the AFTER-derived log collection, our Gemini-2.5-Flash pipeline reaches 82.7% accuracy, 90.1% F1, and 83.7% recall, compared with 34.6% accuracy, 46.9% F1, and 30.6% recall for our reproduction of the released JudgeSkill v4 with Claude Sonnet 4.6. With the same Sonnet 4.6 backbone, our pipeline reaches 55.8% accuracy, 69.3% F1, and 53.1% recall. Together with the SkillTV ablations, this supports a broader *evidence-grounded evaluation* view: artifact access is important when success lives in generated files, while skills provide a distinct source of task semantics and constraints.

D COST ACCOUNTING

With the one–two micro-judge budget, our measured total cost for the 203-case SkillTV split is approximately $4.69 without the Skeptic and $5.20 with it, or about $0.0231 and $0.0256 per case. SkillTV-Bench does not report a dollar cost for refined JudgeSkill; using the released Sonnet 4.6 backbone and our reconstruction of the released JudgeSkill v4 procedure, we estimate $40–$45 for the same split, or about $0.197–$0.222 per case, which is roughly 7.7–9.6 times higher than our one–two runs. Our estimates for the one–ten setting are $43.5 without the Skeptic and $44.5 with it (about $0.214 and $0.219 per case). Because the MetaAgent usually saturates its budget, this estimate reflects close to ten micro-judges per case rather than a loose upper bound. These ratios are arithmetic comparisons, not evidence that the architectures would retain the same quality under a shared model or token budget.

E DIRECT LLM BASELINES AND BACKBONE SCREENING

Table 13 lists the direct LLM judges reported by SkillTV-Bench on the 203-case split; we did not reproduce them. Table 14 reports our own backbone screening.

Table 13: Direct LLM judge results on the 203-case SkillTV held-out evaluation split, as reported by the SkillTV-Bench authors (not reproduced by us), converted to percentages. Precision, recall, and F1 are for the pass class. GPT-5.2 has the highest accuracy among them and the lowest pass F1.

Judge	Backbone	Acc.	Bal. Acc.	Prec.	Recall	F1
Direct LLM	GPT-5.2	48.3	47.8	21.6	46.8	29.5
Direct LLM	Claude Sonnet 4.6	37.0	56.0	25.8	91.5	40.2
Direct LLM	Gemini 3.1 Pro Preview	44.3	58.6	27.4	85.1	41.5

Table 14: Our backbone screening next to the baselines above: our pipeline with different backbone models on the 57-case artifact-bearing subset of SkillTV, using the stripped-trace configuration. N is the number of scored cases: runs marked * were aborted early because the backbone could not reliably produce the structured outputs the pipeline requires.

Backbone	N	Acc.	Bal. Acc.	Prec.	Recall	F1	TP	TN	FP	FN
DeepSeek-v4-Flash*	5	40.0	62.5	25.0	100.0	40.0	1	1	3	0
Qwen3.6-Flash*	17	35.3	60.7	21.4	100.0	35.3	3	3	11	0
Gemini-3.5-Flash-Lite	57	19.3	50.0	19.3	100.0	32.4	11	0	46	0
GPT-5.6-Luna	57	42.1	50.3	19.4	63.6	29.8	7	17	29	4
Gemini-2.5-Flash	57	38.6	41.2	14.7	45.5	22.2	5	17	29	6

Backbone screening. Structured-output reliability, not headline accuracy, dominated this comparison (Table 14). DeepSeek-v4-Flash could not consistently satisfy the pipeline's structured-output contract and its run was aborted after a handful of cases, while Qwen3.6-Flash returned API errors unless the literal token "json" appeared in the request and was stopped after 17 cases. Gemini-3.5-Flash-Lite approved every trajectory, making it unusable as a judge. GPT-5.6-Luna and Gemini-2.5-Flash both completed the subset with dependable structured outputs; we selected Gemini-2.5-Flash for its substantially lower cost, and its later configurations with materialized evidence (Table 3) are the ones reported in the main paper.

F Repository contents

The repository contains the evaluation pipeline (MetaAgent routing, the built-in micro-judge registry, deterministic validators, the OutcomeJudge, the optional Skeptic pass, and the E2B-backed `code_reproducibility` judge), the scoped evidence tools `read_skill` and `read_output_file` with their document converters, the schema-driven output interface, unit tests, and a README with setup instructions; the backbone model is set in `.env.example`.

For SkillTV, one script downloads the 203-case evaluation split with its task-time skills and output files, another builds the deduplicated traces, and a third builds the head–tail variant (first and last ten steps, with every tool-input payload replaced by a placeholder). The runner exposes evidence preset, trace representation, micro-judge budget, and deterministic validators as command-line options. The forced-inline runner and the single-call Gemini baseline of Table 4 have their own scripts. Who&When has run and scoring scripts, and the output schemas for Who&When and TRAIL are in `output_formats/`. Every run writes a prediction file per trace, with each micro-judge's findings, and a `summary.json` with the run configuration, token usage, and accuracy.

The repository includes the code and configurations for every run reported in the paper, so each configuration can be rerun from the command line; the per-case predictions of our evaluator are not included. The JudgeSkill v4 judgments on the AFTER-derived collection are included with their labels, so that row can be rescored directly. Table 9 scores all evaluators on the 52 AFTER-derived packages judged by both Sonnet runs. The AFTER-derived collection contains, for each session/log package, the agent execution log, the task-time skills, the generated output files, and the binary success label: the run's own test result for the Hermes runs, and the result of running the task's tests against the produced outputs for the Claude Code runs.