The AI Replication Engine: Autonomous Verification of Empirical Research in Social Sciences

Anonymous Authors

Abstract

Autonomous scientific replication is a demanding testbed for LLM agents and scientific NLP: systems must read papers, inspect code, execute analyses, and ground claims in execution traces and regenerated artifacts. We introduce the *AI Replication Engine* (AIRE), a bounded multi-agent system for empirical replication of social-science papers with shared code and data. AIRE combines LLM reasoning, deterministic execution, and structured retrieval to assess computational reproducibility, manuscript–code implementation consistency, and robustness-check opportunities.

We also contribute an evaluation framework for autonomous replication as an evidence-grounded NLP task. Drawing on large-scale replication efforts, we construct benchmarks and annotation protocols that separate recovered reference findings, duplicates, hallucinations, and valid-novel outputs under incomplete reference knowledge. On 74 economics and political-science papers, GPT-5.5 and Claude Opus 4.7 perform best on computational reproducibility, recover fewer expert-documented implementation and robustness checks, but still produce useful reference-absent findings, especially robustness checks. The results suggest that autonomous replication is a promising but supervision-dependent setting for evaluating grounded scientific agents.

1 Introduction

LLM agents are increasingly used as scientific assistants that read papers, inspect code, call tools, execute analyses, and return structured outputs (Yao et al., 2023; Miao et al., 2025; Bibal et al., 2025). For NLP, this shift raises an evaluation problem that is not captured by conventional question answering, summarization, or coding benchmarks. A scientific agent must not only produce a plausible answer; it must ground natural-language claims in heterogeneous evidence, including manuscript text, source code, execution traces, regenerated artifacts, and provenance records.

Empirical replication is a natural stress test for this class of systems. Independent replication is central to scientific credibility, but even publicly available code and data often require tracing empirical specifications across scripts, identifying undocumented analytical decisions, and distinguishing substantive discrepancies from benign implementation differences (Brodeur et al., 2024, 2026; Gundersen et al., 2025). Replication therefore turns scientific-document understanding into an evidence-grounded agent task: the system must connect claims in a paper to implemented analyses, runtime behavior, and regenerated outputs.

Existing evaluations of reproducibility agents often emphasize workflow reconstruction, execution success, or reproduction of computational artifacts (Hu et al., 2025; Zhang et al., 2026). These are necessary but incomplete criteria. In empirical replication, success also requires recovering documented findings, identifying mismatches between reported and implemented procedures, separating substantive discrepancies from technical failures, reasoning under incomplete reference knowledge, and producing outputs that experts can audit. This makes replication a demanding benchmark for language agents that combine document understanding, code reasoning, retrieval, execution, and structured prediction (Jimenez et al., 2024; Siegel et al., 2025; Chen et al., 2025; Starace et al., 2025).

Our contribution is twofold. We first introduce the *AI Replication Engine* (AIRE), a model-agnostic, bounded multi-agent system for empirical social science papers with shared code and data. AIRE combines LLM agents, deterministic code execution, retrieval over manuscripts and replication packages, and schema-constrained reporting for three linked tasks: computational reproducibility verification, analytical implementation inconsistency detection, and robustness-check proposal.

The system operates in a controlled execution environment where substantive outputs must be supported by current-run logs, regenerated artifacts, and structured provenance.

Second, we operationalize autonomous empirical replication as an evidence-grounded scientific NLP evaluation framework. This includes task definitions, benchmark construction from prior large-scale replication efforts, and evidence-linked annotation and scoring protocols that treat human replication reports as informative but incomplete reference sets rather than exhaustive ground truth (Brodeur et al., 2024, 2026). The protocols measure both reference recovery and expert validation of additional AIRE-only outputs, meaning retained system outputs that do not align with reviewed reference annotations. These outputs are scored as valid-novel findings, hallucinations, and duplicates.

The empirical setting is economics and political science, where many papers provide replication packages and replication projects supply reviewed reproduction evidence. This setting allows agents to inspect manuscripts, execute code, compare reported and regenerated results, and retrieve relevant evidence while exposing key failure modes of scientific-assistant systems, including evidence misattribution, incorrect result mapping, unsupported reasoning, duplicate findings, and overconfident interpretation under incomplete information.

Using this scaffold, we compare GPT-5.5 and Claude Opus 4.7 under identical prompts, tools, schemas, retrieval settings, and execution environments. The results support three conclusions. First, AIRE is strongest when it comes to computational reproducibility: both models reach 99% executability agreement and approximately 87 F1 on comparison and correspondence recovery. Second, implementation inconsistency detection remains difficult; GPT-5.5 recovers more reference inconsistencies and more valid-novel issues, while Opus 4.7 is more conservative and achieves higher strict precision. Third, robustness-check generation is better characterized as expert-review triage than exact recovery of reference reports: strict overlap is low, but more than 90% of model-only robustness checks are judged valid-novel and hallucination rates remain below 2%.

2 Related Work

Our work builds on tool-using language agents that interleave reasoning with actions in external environments (Yao et al., 2023; Schick et al., 2023). It also relates to agent benchmarks that evaluate models through code, tools, and execution rather than static text prediction, including software-engineering tasks, data-driven scientific discovery, and research-paper replication (Jimenez et al., 2024; Chen et al., 2025; Siegel et al., 2025; Starace et al., 2025). These benchmarks motivate the broader NLP question addressed here: how should language agents be evaluated when outputs must be grounded in executable evidence and remain auditable by human experts?

Replication studies in economics and political science show that computational reproducibility is often achievable but far from sufficient: reproduction reports document execution failures, implementation discrepancies, and robustness-sensitive conclusions (Brodeur et al., 2024, 2026). Prior work on analytical flexibility, specification searching, and selective reporting further suggests that rerunning original scripts alone is insufficient for evaluating empirical robustness or implementation validity (Gelman and Loken, 2013; Steegen et al., 2016; Brodeur et al., 2020; Askarov et al., 2024). These concerns motivate treating computational reproducibility, implementation consistency, and robustness assessment as distinct but related tasks with different evidentiary standards.

Recent work has begun using AI systems for reproducibility and replication, including systems for parsing papers and code into executable reproduction artifacts, interactive reproduction agents, LLM-assisted replication workflows, and large-scale AI-supported reanalysis pipelines (Yan et al., 2025; Bibal et al., 2025; Miao et al., 2025; Kubota et al., 2026; Xu and Yang, 2026). Recent social-science systems evaluate agentic reproducibility with access to replication packages (Hu et al., 2025; Zhang et al., 2026), while concurrent work studies reproduction from papers and data without access to original code (Kohler et al., 2026). AIRE differs from the closest systems by evaluating three evidence-grounded replication tasks rather than a single reproducibility judgment, and by scoring both recovery of reviewed reference findings and expert validation of reference-absent system outputs. This makes the benchmark sensitive to duplicate findings, hallucinations, and valid-novel outputs under incomplete reference evidence. A tabular comparison appears in Appendix A.

We frame AIRE as a bounded replication system rather than a general scientific-discovery agent.

Questions of construct validity, external validity, and conceptual replication often require substantive domain expertise, alternative research designs, or new data, and therefore fall outside the scope of the present system (Athey and Imbens, 2017; Esterling et al., 2025; Holzmeister et al., 2024; Vivalt, 2020).

3 Replication Task Operationalization

3.1 Scope

The input to AIRE is a paper, its replication package, and optional package-layout metadata. At inference time, the engine may use only these materials and evidence generated during the current run, including execution logs, regenerated artifacts, and derived outputs. Human reproduction reports are used to construct reference annotations and metrics, but are not available to AIRE during inference.

The current system is intentionally bounded to empirical replication tasks that can be evaluated from shared code and data in standardized execution environments. The reported experiments focus on papers with Stata or R replication packages and exclude cases requiring unavailable proprietary data, unsupported primary analysis languages, excessive runtime, or unusually large packages. Computational reproducibility focuses on selected main-paper tables rather than all appendix material or prose claims. By contrast, implementation consistency and robustness tasks consider the full paper and package.

3.2 Replication Tasks

We consider three replication tasks that capture distinct aspects of empirical validation and serve as targets for evaluation.

Computational reproducibility. The agent must determine whether the provided code executes successfully and whether regenerated outputs correspond to reported numerical results. We distinguish technical executability from output correspondence. Executability concerns whether the package reaches an admissible current-run state, while output correspondence concerns whether regenerated values match selected reported quantities. Comparisons are evaluated over predefined manifest metrics, including coefficients, standard errors, test statistics, and observation counts linked to specific paper items. A comparison counts only when supported by current-run evidence such as execution logs or regenerated artifacts.

Analytical implementation consistency. The agent must identify discrepancies between the empirical analysis described in the manuscript and the analysis implemented in code, as well as contradictions within the code affecting the same analysis. This task excludes pure runtime failures and other executability problems. A qualifying inconsistency must specify the manuscript expectation, implemented behavior, mechanism of divergence, and affected empirical result or design component.

Robustness-check proposal. The agent must propose feasible robustness checks that vary analytically meaningful choices while preserving the underlying estimand. Examples include alternative samples, controls, outcome definitions, treatment definitions, or inference procedures. Checks requiring external data are treated as out of scope.

4 System

Figure 1 summarizes the technical architecture. The system is organized into two layers. The inference layer contains the model-visible workflow: source binding, retrieval and indexing, bounded tool use, execution, agent stages, ledger state, and structured outputs. The evaluation layer is post-hoc: it aligns those outputs to held-out reference evidence and computes task metrics. This separation prevents reviewed replication reports or annotation sheets from leaking into inference.

4.1 Inference Layer

The inference layer begins with three inputs: a paper PDF, a replication package, and optional package-layout metadata. These inputs define the evidence boundary. AIRE may parse the paper, scan the package, execute code, and inspect current-run artifacts, but it may not use human reproduction reports, annotation sheets, or precomputed reference labels.

Source binding and retrieval. Run control first creates a run context and binds the paper to a source bundle containing the PDF, package root, and layout metadata. The paper is parsed into text, table, and section-level representations; when embedded text is unreliable, OCR recovers page text. The package scanner indexes candidate entry points, code files, data files, pre-generated outputs, build files, documentation, and language-specific dependency signals. Pre-generated or preexisting outputs can be used for orientation and file discovery, but

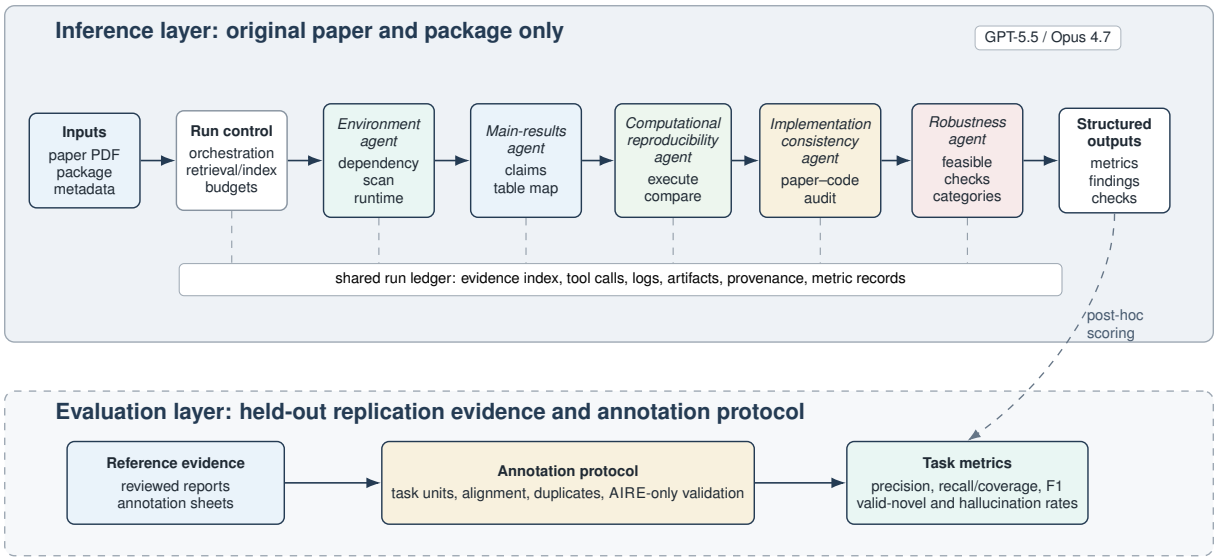

Figure 1: AIRE system and evaluation schematic. The inference layer takes the original paper and replication package through run control, five agent stages, and schema-constrained outputs. The shared run ledger stores evidence, tool calls, logs, artifacts, provenance, and metric records. The evaluation layer scores AIRE outputs against held-out reviewed replication evidence through a separate annotation protocol.

they are blocked as reproduction evidence: numerical comparisons must be supported by artifacts regenerated during the current run.

Tool control and guardrails. Run control is infrastructure, not a separately evaluated agent. It enforces stage order, tool-call budgets, execution timeouts, structured schemas, evidence requirements, uncertainty fields, and blocked-evidence rules. Agent access is mediated through bounded tools for file inspection, PDF extraction, code execution, artifact listing, comparison registration, and result serialization. The same parameters are used for both GPT-5.5 and Claude Opus 4.7.

Agent stages. The five agent stages operate sequentially. The *environment agent* scans dependencies, shell tools, and runtime availability so later stages can separate unsupported execution conditions from substantive failures. The *main-results agent* maps central manuscript claims to candidate headline tables, which scopes the comparison manifest before code execution. The *computational reproducibility agent* executes original scripts or narrowly scoped wrappers and compares regenerated values against the manifest. The *implementation consistency agent* inspects manuscript, code, logs, and generated artifacts for substantive paper–code discrepancies while excluding pure runtime failures. The *robustness agent* produces structured candidate checks grounded in available code, data, and current-run evidence.

Execution and numeric matching. The execution layer supports Python, R, and Stata primitives, while the reported experiments are restricted to R and Stata replication packages. Runs use fixed software environments, dependency manifests, execution timeouts, and deterministic runtime settings where supported. Substantive code edits are disallowed; if original analysis code is incomplete or internally inconsistent, the relevant item is marked blocked rather than silently repaired. The comparison manifest stores the original value, reproduced value, table or item identifier, row and column labels, provenance, evidence tier, and match status. A value is classified as matching if it matches exactly, matches after rounding to manuscript display precision, satisfies reported p-value thresholding, falls within an absolute tolerance of 0.0005, or falls within a 5% relative tolerance.

Shared run ledger and outputs. All stages write to and read from the shared run ledger shown below the agent boxes. The ledger records extracted paper text, package indexes, tool calls, execution logs, generated artifacts, comparison records, provenance metadata, and failure records. Downstream agents retrieve from this ledger rather than relying on transient conversational memory. The final inference product is a schema-constrained set of metrics, findings, proposed checks, and report artifacts.

Additional implementation details on retrieval, execution, provenance management, and bounded tool interaction are reported in Appendix B.

4.2 Evaluation Layer

The evaluation layer begins only after inference completes. Structured AIRE outputs are aligned to held-out reference evidence from reviewed reports and annotation sheets. The annotation protocol defines the scoring unit for each task, duplicate handling, granularity decisions, alignment rules, and validation of AIRE-only outputs. The soft arrow from structured outputs to task metrics denotes post-hoc scoring only; reference evidence never flows back into the inference layer.

The metrics layer consumes the aligned annotations and computes task-level indicators. For computational reproducibility, the evaluation reports coverage, equivalent to recall over required comparisons, match rate among aligned comparisons, executability agreement, and hallucination or valid-novel diagnostics for AIRE-identified mismatches absent from the reference set. For implementation consistency and robustness, the evaluation reports precision, recall/coverage, F1, valid-novel rate, and hallucination rate over grouped findings or grouped robustness ideas.

5 Data and Evaluation Design

5.1 Data Sources

We draw on three sources of replication evidence. The first is the Institute for Replication (I4R), including its Discussion Paper Series and Replication Games, where independent teams reproduce empirical studies (Brodeur et al., 2024, 2026). The second is the Robustness and Replicability in Development Economics (R2E-Dev/RWI) project, which provides reviewed replication evidence for development economics (Brodeur et al., 2026). The third is the World Bank Reproducibility Initiative, which contributes technical executability cases under the computational reproducibility task.

Applying the scope filters defined in Section 3.1 yields 119 papers.[1] The reported results use an evaluation subset of 74 papers selected in random-ID batches from the filtered benchmark (Table 1); the World Bank source contributes only technical executability annotations under computational reproducibility. Appendix C reports the evaluation-corpus construction pipeline and exclusion counts by source.

Source	Filtered benchmark	Evaluation subset
I4R	71	46
R2E-Dev/RWI	39	23
World Bank	9	5
Total	119	74

Table 1: Filtered benchmark sample and currently evaluated subset.

Task	Reference units	Mean / paper
Reproducibility	comparison	89.79
Implementation	coding error	0.52
Robustness	robustness check	4.37

Table 2: Reference annotation density by replication task.

5.2 Reference Annotations and Output Scoring

Six co-authors contributed to the annotation workflow across the three replication tasks: 3 postdocs, 1 professor, 1 master's student and 1 senior researcher. Unclear cases were flagged and resolved collaboratively. The workflow separated reference construction from AIRE output scoring. Reference data from prior human reproduction efforts were treated as structured but imperfect benchmarks rather than exhaustive ground truth. Annotators reviewed raw reference entries and revised, merged, split, deduplicated, reclassified, or excluded entries when necessary. Appendix D summarizes the annotation and alignment protocol used across tasks.

Reference annotation density by task is summarized in Table 2. For computational reproducibility, annotators reviewed executability evidence for 74 papers and comparison-level alignments against 6,838 reference comparisons from papers' selected main tables. Annotators reviewed these reference entries against 63 AIRE-flagged implementation inconsistencies and 544 AIRE-proposed robustness checks. Among these, 54 implementation-consistency findings and 504 robustness checks were model-only outputs absent from the reference annotations and were further assessed for valid-novel status, hallucination, infeasibility, and duplication.

For computational reproducibility, output correspondence was evaluated at the level of compar-

[1] One additional case was excluded by a dependency-retrieval filter because it relied on an idiosyncratic user-written Stata package with a non-standard installation scheme.

isons in selected main tables. Comparisons had to refer to the same reported quantity, table, and model specification; incorrect table, row, column, model, or statistic matches were treated as identification errors rather than numerical mismatches.

For implementation inconsistencies and robustness checks, scoring was based on underlying issues or robustness ideas rather than broad categories. Outputs absent from the reviewed reference set were not automatically treated as false positives. Instead, annotators evaluated whether such outputs constituted valid-novel findings or checks, defined as evidence-grounded, methodologically defensible, feasible with the available materials, and non-duplicative of existing author analyses. Duplicate outputs and granularity problems were recorded separately as diagnostics.

5.3 Model Comparison

Our goal in the present experiments is not to attribute performance gains to individual architectural components, but to evaluate whether a fixed evidence-grounded replication scaffold can support bounded replication tasks across frontier backbone models.

We compare GPT-5.5 and Claude Opus 4.7 as backbone models. The agent scaffold, prompts, retrieval settings, tool interfaces, execution environment, evidence policy, and per-paper compute budgets are held fixed. No model weights are fine-tuned. The run ledger records model identifiers, provider settings, prompt versions, and run metadata.

The scaffold was calibrated to stabilize execution and retrieval behavior across heterogeneous replication packages. Calibration addressed infrastructure-level issues such as package layouts, path assumptions, dependency installation, OCR failures, and output extraction. These changes targeted common package-handling problems rather than paper-specific substantive judgments.

6 Metrics

The main paper reports task-specific *precision*, *recall*, and *F1* as conventional measures of reference alignment. For implementation consistency and robustness-check generation, we also report the *valid-novel rate* and *hallucination rate* and *cross-model agreement* to characterize outputs beyond reference recovery. Secondary diagnostics are reported in the appendix.

Recall measures the share of reference items recovered by AIRE, while precision measures the share of non-duplicate AIRE outputs aligned with the reference set. The valid-novel rate measures the share of reference-absent, non-duplicate outputs that expert annotators judged evidence-grounded and methodologically defensible. The hallucination rate measures the share of non-duplicate outputs judged unsupported, invalid, or based on misinterpreted evidence.

Computational reproducibility is evaluated in two stages. First, executability agreement measures whether AIRE and the reference agree that a package reaches an admissible current-run state. Second, comparison metrics evaluate whether AIRE correctly aligns reported quantities with regenerated outputs. Correspondence metrics additionally require the aligned comparison to be correctly classified as a match or mismatch under the numeric correspondence rule. Match share reports the fraction of aligned comparisons whose regenerated values match the reported values.

For implementation consistency and robustness, we distinguish between strict and weak alignment. The main text reports strict indicators, while Appendix E provides additional metric definitions and denominator conventions.

Metrics are evaluated at task-specific units: papers for executability agreement, comparison-level quantities for computational reproducibility, inconsistency findings for implementation consistency, and robustness ideas for robustness evaluation.

7 Results

All main results are reported as percentages. The aggregate pattern is strongly task-dependent. Both backbone models perform well on execution-grounded computational reproducibility, where the system can bind paper quantities to regenerated artifacts and evaluate numeric correspondence. Performance is much weaker under strict recovery of implementation inconsistencies and robustness checks from reviewed reference reports. These strict scores should not be read as ordinary false-positive rates because the reference reports are informative but incomplete; the evaluation therefore separately validates model-only outputs. This distinction is especially important for robustness checks, where most model-only proposals are judged valid-novel rather than hallucinated.

The experimental setup combines local execu-

Metric	GPT-5.5	Opus 4.7
Executability agreement	98.65	98.65
Match share *(ref.: 98.79)*	98.22	97.70
Comparison precision	93.47	92.81
Comparison recall	81.43	81.22
Comparison F1	87.03	86.63
Correspondence precision	92.77	92.34
Correspondence recall	81.04	80.89
Correspondence F1	86.51	86.24

Table 3: Computational reproducibility results. Executability agreement is evaluated at the paper level; comparison and correspondence metrics are evaluated over reported–regenerated quantity pairs. Denominator details are reported in Appendix E.

Metric	GPT-5.5	Opus 4.7
Strict precision	6.98	10.53
Strict recall	8.33	5.56
Strict F1	7.59	7.27
Valid-novel rate	48.72	33.33
Hallucination rate	16.28	36.84
Cross-model agreement	11.63	26.32

Table 4: Implementation-consistency results. Metrics are evaluated over retained non-duplicate model-flagged inconsistencies, reviewed reference inconsistencies, and model-only inconsistencies as appropriate. Denominator details are reported in Appendix E.

tion with remote model inference. The agent system and replication-code execution run locally on a Mac M4 Max with 16 CPU cores, 40 GPU cores, 128 GB of RAM, and 4 TB of storage. Language model inference uses proprietary foundation models accessed through the OpenAI and Anthropic APIs with a budget of $15000 USD equally divided among the two.

The system was implemented as a LangGraph and LangChain application. LangGraph represented the replication workflow as a stateful graph of agent nodes, conditional transitions, retry paths, and ledger updates, while LangChain provided model interfaces, tool bindings, prompt management, and structured-output utilities (LangChain, 2026b,a). All code will be released under CC-BY 4.0.

7.1 Computational Reproducibility

Table 3 shows that computational reproducibility is the strongest setting for AIRE. Both models agree with the effective reference executability labels for 99% of evaluated packages. Conditional on successful execution, the main limitation is comparison coverage rather than numeric correspondence classification. Comparison recall is 81% for GPT-5.5 and 81% for Opus 4.7, while comparison precision remains above 92% for both models. Correspondence F1 is similarly high, reaching 86.58 for GPT-5.5 and 86.27 for Opus 4.7.

Metric	GPT-5.5	Opus 4.7
Strict precision	7.72	6.62
Strict recall	7.07	6.06
Strict F1	7.38	6.33
Valid-novel rate	92.80	91.73
Hallucination rate	1.10	1.84
Cross-model agreement	26.84	26.84

Table 5: Robustness-check results. Metrics are evaluated over retained non-duplicate model-flagged checks, reviewed reference checks, and model-only checks as appropriate. Denominator details are reported in Appendix E.

The match-share results further indicate that once AIRE correctly binds the relevant reported and regenerated quantities, it usually classifies numerical correspondence correctly. The reference match share is 98.79%, compared with 98.22% for GPT-5.5 and 97.70% for Opus 4.7. This pattern suggests that residual errors arise primarily from incomplete recovery or misbinding of table comparisons rather than systematic failures to execute analyses or recognize numeric agreement. Model differences are modest overall, but GPT-5.5 achieves slightly higher precision, recall, and correspondence scores across most metrics. Supplementary evaluation results are reported in Appendix F.

7.2 Implementation Consistency

Table 4 reports strict-alignment implementation-consistency results. Both models achieved low strict recall, indicating limited recovery of the reviewed reference inconsistencies. GPT-5.5 generated substantially more candidate inconsistencies and achieved higher recall, but at the cost of lower precision and more hallucinated claims. Opus 4.7 was more conservative, producing fewer retained findings with somewhat higher precision but lower coverage.

At the same time, both models produced valid-novel inconsistencies absent from the reviewed reference reports, suggesting that AIRE was able to surface some genuine implementation concerns beyond the existing replication evidence. However, hallucination rates remained non-trivial, highlighting the difficulty of distinguishing substantive analytical inconsistencies from manuscript ambiguity,

code complexity, or unsupported causal interpretations. Unsupported implementation-error claims are particularly costly because they require expert adjudication.

7.3 Robustness Checks

Table 5 reports strict-alignment robustness-check results. Strict precision, recall, and F1 are low for both backbone models, indicating limited overlap with the robustness checks documented in the reviewed reference reports. This overlap should be interpreted cautiously: AIRE was restricted to at most four robustness checks per paper, which is close to the reference-report average of 4.37 checks (Table 2), but neither set should be treated as an exhaustive enumeration of all reasonable alternatives. The high valid-novel rates therefore matter substantively. Most retained AIRE proposals absent from the reference annotations were judged feasible, methodologically defensible, and non-duplicative, while hallucination rates remained low. This suggests that AIRE often surfaced useful robustness ideas beyond the reviewed reports rather than merely failing to recover reference checks.

A recurring boundary case concerned sample-changing robustness checks. Many were judged valid and useful, but annotators had to assess when changes in sample size altered statistical power or shifted the effective estimand enough to make the check less comparable to the original claim.

8 Limitations and Future Directions

AIRE is a bounded replication system. The current evaluation focuses on economics and political science papers with shared R or Stata replication packages and evaluates computational reproducibility primarily on selected main-paper tables. Future work should test whether the same annotation and agent scaffolds transfer to domains with different software ecosystems, proprietary data, simulation-heavy workflows, or claims that cannot be reduced to table-level numerical comparisons.

AIRE is currently diagnostic rather than interventionist. The system identifies execution failures, numerical mismatches, implementation inconsistencies, and candidate robustness checks, but does not perform automated repair, substantive code rewriting, or re-estimation of alternative specifications. Extending the system from diagnosis to intervention would require additional safeguards because repairs and alternative estimations can alter the scientific object being evaluated.

A third direction is human-in-the-loop replication. AIRE is not a substitute for expert review: implementation inconsistency detection and robustness-check proposal often require substantive domain judgment. Future systems should therefore study how agents can best support expert replicators, for example by prioritizing candidate findings, exposing uncertainty, and linking claims to executable evidence.

The present evaluation also holds the agent scaffold fixed and compares frontier backbone models within that scaffold. Future work could evaluate alternative orchestration designs, including single-agent variants, retrieval-free variants, relaxed evidence constraints, and simpler non-agentic baselines. At the same time, exhaustive ablation studies may have limited shelf life in a rapidly changing model and tooling ecosystem. Such analyses are likely most useful when they test durable design principles, including evidence constraints, provenance tracking, and task decomposition. Future versions of the system could also strengthen execution isolation and reproducibility guarantees through fully containerized runtime environments and stricter dependency sandboxing.

9 Conclusion

We reframe autonomous replication as both a systems problem and an evaluation-design problem. AIRE shows how LLM agents can be coupled with deterministic execution, retrieval, evidence ledgers, and schema-constrained outputs to perform bounded replication tasks. The benchmark contribution is equally important: structured annotation protocols make it possible to evaluate not only whether code runs, but whether AIRE outputs recover reviewed replication findings, avoid unsupported claims, and surface valid candidate findings for expert review beyond an incomplete reference set.

More broadly, we argue that AI-for-science systems should be evaluated against the epistemic structure of the scientific task, not only against end-to-end task completion. For empirical replication, that structure includes executable evidence, provenance, reference incompleteness, duplicate handling and human-reviewable uncertainty. AIRE provides a concrete testbed for studying these challenges in autonomous scientific agents.

References

Zohid Askarov, Anthony Doucouliagos, Hristos Doucouliagos, and TD Stanley. 2024. Selective and (mis) leading economics journals: Meta-research evidence. *Journal of Economic Surveys*, 38(5):1567–1592.

Susan Athey and Guido W Imbens. 2017. The state of applied econometrics: Causality and policy evaluation. *Journal of Economic Perspectives*, 31(2):3–32.

Adrien Bibal, Steven N Minton, Deborah Khider, and Yolanda Gil. 2025. AI copilots for reproducibility in science: A case study. *arXiv preprint*, arXiv:2506.20130.

Abel Brodeur, Nikolai Cook, and Anthony Heyes. 2020. Methods matter: P-hacking and publication bias in causal analysis in economics. *American Economic Review*, 110(11):3634–3660.

Abel Brodeur, Kevin Esterling, Jörg Ankel-Peters, Natália S Bueno, Scott Desposato, Anna Dreber, Federica Genovese, Donald P Green, Matthew Hepplewhite, Fernando Hoces de la Guardia, and 1 others. 2024. Promoting reproducibility and replicability in political science. *Research & Politics*, 11(1):20531680241233439.

Abel Brodeur, Derek Mikola, Nikolai Cook, Thomas Brailey, Ryan Briggs, Alexandra de Gendre, Yannick Dupraz, Lenka Fiala, Jacopo Gabani, Romain Gauriot, Joanne Haddad, Ryan McWay, Joel Levin, Magnus Johannesson, Lennard Metson, Jonas Minet Kinge, Wenjie Tian, Timo Wochner, Sumit Mishra, and 328 others. 2026. Reproducibility and robustness of economics and political science research. *Nature*, 652(8108):151–156.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, Vishal Dey, Mingyi Xue, Frazier N. Baker, Benjamin Burns, Daniel Adu-Ampratwum, Xuhui Huang, Xia Ning, Song Gao, Yu Su, and Huan Sun. 2025. Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery. In *International Conference on Learning Representations*.

Cheng Cui, Ting Sun, Suyin Liang, Tingquan Gao, Zelun Zhang, Jiaxuan Liu, Xueqing Wang, Changda Zhou, Hongen Liu, Manhui Lin, Yue Zhang, Yubo Zhang, Handong Zheng, Jing Zhang, Jun Zhang, Yi Liu, Dianhai Yu, and Yanjun Ma. 2025. PaddleOCR-VL: Boosting multilingual document parsing via a 0.9b ultra-compact vision-language model. *Preprint*, arXiv:2510.14528.

Kevin M Esterling, David Brady, and Eric Schwitzgebel. 2025. The necessity of construct and external validity for deductive causal inference. *Journal of Causal Inference*, 13(1):20240002.

Andrew Gelman and Eric Loken. 2013. The garden of forking paths: Why multiple comparisons can be a problem, even when there is no "fishing expedition" or "p-hacking" and the research hypothesis was posited ahead of time. *Department of Statistics, Columbia University*, 348(1-17):3.

Odd Erik Gundersen, Øystein Cappelen, Morten Mølna, and Nils Gjerstad Nilsen. 2025. The unreasonable effectiveness of open science in AI: A replication study. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 26211–26219.

Felix Holzmeister, Magnus Johannesson, Robert Böhm, Anna Dreber, Jürgen Huber, and Michael Kirchler. 2024. Heterogeneity in effect size estimates. *Proceedings of the National Academy of Sciences*, 121(32):e2403490121.

Chuxuan Hu, Liyun Zhang, Yeji Lim, Aum Wadhwani, Austin Peters, and Daniel Kang. 2025. Repro-bench: Can agentic AI systems assess the reproducibility of social science research? In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 23616–23626.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R. Narasimhan. 2024. SWE-bench: Can language models resolve real-world github issues? In *International Conference on Learning Representations*.

Benjamin Kohler, David Zollikofer, Johanna Einsiedler, Alexander Hoyle, and Elliott Ash. 2026. Read the paper, write the code: Agentic reproduction of social-science results. *arXiv preprint arXiv:2604.21965*.

So Kubota, Hiromu Yakura, Samuel Coavoux, Sho Yamada, and Yuki Nakamura. 2026. Llm-assisted replication for quantitative social science. *arXiv preprint*, arXiv:2602.18453.

LangChain. 2026a. Langchain documentation. Accessed 2026-05-24.

LangChain. 2026b. Langgraph documentation. Accessed 2026-05-24.

Jiacheng Miao, Joe R Davis, Yaohui Zhang, Jonathan K Pritchard, and James Zou. 2025. Paper2agent: Reimagining research papers as interactive and reliable AI agents. *arXiv preprint*, arXiv:2509.06917.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36.

Zachary S. Siegel, Sayash Kapoor, Nitya Nagdir, Benedikt Stroebl, and Arvind Narayanan. 2025. CORE-bench: Fostering the credibility of published research through a computational reproducibility agent benchmark. *Transactions on Machine Learning Research*.

Giulio Starace, Oliver Jaffe, Dane Sherburn, James Ha Tun Aung, Jun Shern Chan, Leon Maksin, Rachel Dias, Evan Mays, Benjamin Kinsella, Wyatt Thompson, Johannes Heidecke, Amelia Glaese, and Tejal Patwardhan. 2025. Paperbench: Evaluating AI's ability to replicate AI research. *arXiv preprint arXiv:2504.01848*.

Sara Steegen, Francis Tuerlinckx, Andrew Gelman, and Wolf Vanpaemel. 2016. Increasing transparency through a multiverse analysis. *Perspectives on Psychological Science*, 11(5):702–712.

Eva Vivalt. 2020. How much can we generalize from impact evaluations? *Journal of the European Economic Association*, 18(6):3045–3089.

Yiqing Xu and Leo Yang Yang. 2026. Scaling reproducibility: An AI-assisted workflow for large-scale reanalysis. *arXiv preprint arXiv:2602.16733v2*.

Shuo Yan, Ruochen Li, Ziming Luo, Zimu Wang, Daoyang Li, Liqiang Jing, Kaiyu He, Peilin Wu, Juntong Ni, George Michalopoulos, Yue Zhang, Ziyang Zhang, Mian Zhang, Zhiyu Chen, and Xinya Du. 2025. LMR-BENCH: Evaluating LLM agent's ability on reproducing language modeling research. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 6164–6186, Suzhou, China. Association for Computational Linguistics.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*.

Linhao Zhang, Tong Xia, Jinghua Piao, Lizhen Cui, and Yong Li. 2026. Paperrepro: Automated computational reproducibility assessment for social science papers. *arXiv preprint arXiv:2603.00058*.

Appendix

A Closest Related Work Comparison

This appendix compares AIRE with the two closest lines of recent work: systems that evaluate agentic reproducibility using replication packages and systems that attempt paper-level reproduction from manuscripts and data. The comparison is intended to clarify scope rather than rank systems. AIRE differs most directly in treating empirical replication as three evidence-grounded NLP tasks, combining execution traces with manuscript–code alignment, and evaluating both recovery of reviewed reference findings and validation of AIRE-only outputs. This distinction is important because expert replication reports are informative but not exhaustive, so a benchmark for autonomous replication must separately account for duplicates, hallucinations, granularity errors, and valid-novel findings.

B Retrieval, Execution, and Tool Interaction Details

This appendix summarizes implementation details that support the interpretation and reproducibility of the reported experiments. The focus is on evidence retrieval, execution control, provenance tracking, and bounded tool interaction.

B.1 Retrieval and Evidence Management

Each benchmark entry is bound to a *source bundle* consisting of a paper PDF, a replication-package root, and package-layout metadata. The source-discovery stage recursively scans candidate directories and records code files, data files, documentation files, likely outputs, and candidate entry points using package-layout heuristics rather than fixed directory assumptions.

Paper retrieval uses a hybrid PDF extraction strategy. The system first attempts embedded-text extraction and falls back to OCR when extracted text is sparse or unreliable. We use PaddleOCR-VL as our OCR, a compact vision-language document parser designed to recognize complex document elements such as tables, formulas, and charts (Cui et al., 2025). Extracted text and table-like regions are retained for manuscript search, table discovery, headline-result selection, and evidence citation.

Package retrieval separates source inputs from outputs shipped by the original authors. Code files, data files, documentation files, and likely output artifacts are indexed separately. Shipped tables, figures, logs, and derived files may be used for orientation or file discovery, but they are blocked as reproduction evidence for numeric correspondence. A numeric comparison counts only when supported by current-run evidence such as execution logs, regenerated outputs, or verified derived artifacts.

Retrieved materials and generated evidence are organized through a shared run ledger that records source paths, generated artifacts, logs, summaries, metric records, and failure records together with provenance metadata. This design allows downstream agents to query prior tool outputs and generated evidence without relying on conversational memory.

B.2 Execution Environment

Execution is designed to reproduce original analyses while minimizing substantive intervention. Each run creates an isolated working directory with standardized subdirectories for adapted inputs, generated wrappers, derived outputs, figures, diagnostics, and reports. The execution layer supports Python, R, and Stata primitives, while the reported benchmark is restricted to R and Stata packages.

Execution uses fixed software versions, dependency manifests, execution timeouts, and deterministic runtime settings where supported by the underlying software stack. Runtime availability is checked before execution, and failed or timed-out processes are terminated rather than reused across later calls.

The system constructs planned execution steps from candidate scripts, child-script calls, input-file references, output-file references, and paper-item hints. Wrappers adapt paths and working directories so that scripts can execute within the sandboxed environment. The system may rewrite path declarations, placeholder `cd` commands, and project-root macros, but it does not rewrite substantive model specifications or repair analysis logic.

Each execution attempt records the script, wrapper, working directory, generated artifacts, execution status, and failure classification. Generated outputs are indexed after execution and linked to paper items when file names, script context, or extracted labels provide sufficient evidence. If execution fails because data are missing, dependencies are unavailable, scripts are incomplete, or the original analysis code is internally inconsistent, the corresponding item is marked blocked rather than silently repaired.

Work	Benchmark setting	Task scope	Evaluation target
AIRE	Economics and political-science papers with reviewed replication evidence.	Computational reproducibility, implementation consistency, and robustness-check proposal.	Reference recovery plus valid-novel, duplicate, and hallucination rates.
Kohler et al. (2026)	Social-science papers with verified reproducibility and structured paper-derived methods extraction.	Reproduction from methods descriptions and original data without access to author code.	Deterministic comparison between reproduced and original numerical outputs.
Zhang et al. (2026)	REPRO-Bench and a refined difficulty-stratified variant.	Computational reproducibility assessment.	Accuracy of reproduced-result assessment after execution and evaluation.

Table A.1: Closest agentic reproducibility systems compared along benchmark, task, and evaluation dimensions.

B.3 Bounded Tool Interaction

Agents interact with the system through bounded structured tools rather than unrestricted shell access. Tool schemas define the arguments for file reading, PDF extraction, directory listing, code execution, dataset inspection, metric registration, output extraction, and result saving. Available tools are stage-dependent: retrieval stages expose reading and extraction tools, while execution tools are enabled only after the relevant target inventory has been established.

The orchestration layer enforces tool-call budgets, stage constraints, runtime limits, and schema validation. Tool outputs are truncated before being returned to the model so that agents receive relevant evidence without full source trees or unrestricted logs.

Tool interaction is also separated from evaluation. During inference, agents can access only the original paper, replication package, current-run logs, regenerated artifacts, and derived run summaries. Human reproduction reports and annotation sheets are not exposed to the agents during inference and are used only during post hoc evaluation and annotation.

C Evaluation Corpus Construction

C.1 Corpus Sources and Filtering

Table C.2 summarizes the benchmark-construction pipeline and exclusion counts by source. We begin from an initial pool of candidate studies from three replication initiatives: the Institute for Replication (I4R), the R2E-Dev/RWI replication project, and the World Bank Reproducibility Initiative.

We first apply three source-specific filters to define the interim sample labelled *quality-approved reproduction evidence available*. First, we exclude cases for which the reproduction report or reproducibility evidence is unavailable or incomplete. Second, for I4R only, we exclude reports that were not publicly released; retained I4R reports are those released through the I4R Discussion Paper Series, OSF, or other discussion-paper series. This public-availability filter is specific to I4R and does not apply to R2E-Dev/RWI or the World Bank source. For R2E-Dev/RWI, all reports were internally reviewed by a team of five senior researchers but are not publicly available. For the World Bank source, reproducibility evidence derives from internal review of replication packages by the World Bank Reproducibility Initiative team. Third, for the World Bank source, which contributes only to the computational reproducibility task, we exclude cases that reproduced successfully on the first run as recorded by the initiative team. We do so to introduce benchmark cases with known executability problems, since the other data sources primarily contain packages that execute successfully. For retained World Bank cases, we use the replication packages from that first run together with metadata provided by the initiative team.

The resulting interim sample thus contains quality-approved reproduction evidence in a source-specific sense: publicly released reproduction reports for I4R, internally reviewed reports for R2E-Dev/RWI, and initiative-reviewed reproducibility evidence for the World Bank source. Subsequent rows report exclusions from this interim sample based on the operational scope criteria used for AIRE v1 as described in Section 3.1.

The largest reductions from the interim sample arise from practical reproducibility constraints rather than substantive study characteristics. Common exclusions include unsupported analysis languages, unavailable or restricted-access data, replication packages exceeding the storage limit, and excessive runtime requirements. Runtime filtering was applied only where runtime information

was available from the source project. One additional case was excluded because it depended on a user-written Stata package that could not be reliably retrieved or installed under the standardized execution workflow.

The final AIRE v1 benchmark contains 119 papers. The reported evaluation subset contains 74 papers selected in random-ID batches from the filtered benchmark. Hence, this subset reflects the staged evaluation workflow, and not ex post filtering on successful execution, model performance, or annotation outcomes.

C.2 Package Language Composition

Table C.1 reports the analysis-language composition of the evaluated benchmark. Most replication packages are implemented in Stata, with a smaller share of R and mixed Stata–R workflows. This distribution reflects the software ecosystems commonly used in empirical economics and political science.

Package language configuration	Share of papers
Stata	74.3%
R	18.9%
Stata + R	6.8%

Table C.1: Analysis-language composition of the evaluated benchmark.

D Annotation and Alignment Protocol

The annotation protocol has three goals: constructing effective reference sets, scoring AIRE outputs against those references, and separately evaluating outputs absent from the reference set. This appendix summarizes the evaluation and alignment logic.

Across tasks, annotation follows a common four-stage workflow: (i) review and refinement of the reference set, (ii) construction of the retained AIRE output set, including duplicate handling and granularity review, (iii) alignment between AIRE outputs and reviewed reference findings, including cross-model alignment where relevant, and (iv) validation of retained AIRE-only outputs absent from the reference annotations.

Across implementation-consistency and robustness-check tasks, duplicate outputs are outputs that repeat the same underlying finding or robustness idea within a model's retained output set. Granularity diagnostics capture outputs stated at an inappropriate level of specificity for alignment: overly broad outputs collapse multiple distinct findings into one claim, while overly narrow outputs split a single underlying issue or robustness idea into multiple minimally different claims. These diagnostics are recorded separately from strict and weak alignment scores.

The task-specific procedures below adapt this general workflow to executability assessment, output correspondence, implementation consistency, and robustness-check evaluation.

Computational reproducibility. Computational reproducibility consists of two linked subtasks: executability assessment and output correspondence assessment. Reference executability is taken from reproduction reports or reproduction metadata and then amended with additional annotation information where needed. Annotators do not revise the original reference reports themselves; instead, they construct an effective executability indicator for evaluation. For cases reported as unsuccessful, annotators code whether the package is non-executable even under allowable automated repair by AIRE, such as path changes, or executable after such repair. Conversely, when AIRE flags non-executability for a package without a reported execution failure, annotators rerun the package to assess whether an execution problem was present but not explicitly documented in the reference report. Executability agreement is then evaluated between AIRE and this effective reference indicator.

Output correspondence is evaluated on selected main-paper tables for executable packages. Descriptive tables, such as baseline statistics, balance checks, randomization checks, and similar non-estimand tables, are excluded from the comparison set. Each comparison links a reported numeric quantity to a regenerated quantity. Evaluation distinguishes between comparison alignment and correspondence classification. Comparison alignment concerns whether AIRE correctly identifies the relevant table item, including the table, row, column, model specification, and statistic. Incorrect bindings are treated as comparison-identification errors even when the numeric values are similar. Correspondence classification concerns whether an aligned comparison is correctly classified as a match or mismatch under the numeric correspondence rule.

Retained AIRE mismatch claims absent from the reference annotations are separately validated

Filter step	I4R	R2E-Dev/RWI	World Bank
Original source sample	252	67	67
WB packages reproducible on first run	–	–	48
Replication reports unavailable or incomplete	7	6	1
I4R reports not publicly available	72	–	–
Quality-approved reproduction evidence available	173	61	18
Unsupported primary analysis languages	24	2	4
Restricted or unavailable data	2	6	1
Replication package exceeds 0.5 GB	36	13	2
Runtime exceeds two hours	22	–	2
Dependency-retrieval failure	1	0	0
No qualifying robustness checks in reference reports	17	1	0
Final benchmark sample (AIRE v1)	71	39	9
Reported evaluation subset	46	23	5

Table C.2: Benchmark construction and evaluation subset by source.

because the references do not exhaustively enumerate all possible comparison-level mismatches. For these cases, annotators rerun the relevant code and inspect the regenerated evidence to determine whether the mismatch reflects a valid-novel discrepancy or instead results from hallucination, misreading, or comparison-identification error.

Implementation consistency. Reference inconsistencies are first reviewed and refined following the general workflow above. Annotators may revise, split, merge, deduplicate, reclassify, or exclude entries that are not genuine inconsistencies or are not discoverable from the materials available to AIRE. AIRE findings are aligned with reference findings when they address the same underlying issue. Duplicate findings are removed from the retained prediction set but recorded separately as diagnostics. Retained AIRE-only findings are then evaluated as valid-novel, vague, invalid or hallucinated, or unclear.

Robustness checks. Reference robustness checks are similarly reviewed before scoring. Alignment is evaluated at the level of robustness ideas rather than broad analytical decision dimensions such as "controls" or "sample." A robustness idea refers to the specific concern or alternative specification that a check is intended to assess. For example, changes to the control set are grouped together only when they address the same concern, such as adding baseline covariates, excluding post-treatment controls, replacing one socioeconomic control set with another, or adding location fixed effects. Similarly, sample-restriction checks are grouped only when they impose the same substantive type of restriction, such as excluding outliers, restricting to a pre-treatment-balanced sample, or excluding observations with missing baseline variables.

Full alignment requires the same robustness idea and substantively equivalent analytical choices. Partial alignment captures the same underlying robustness concern implemented through different specific analytical choices, such as different covariate sets, treatment definitions, or trimming rules. Retained AIRE-only checks are evaluated based on validity, feasibility, duplication with existing author analyses, unsupported data requirements, and hallucinated or invalid reasoning.

E Metric Definitions and Evaluation Conventions

This appendix defines the weak-alignment indicators, prediction sets, denominator conventions, and zero-denominator rules used for the secondary evaluation diagnostics.

Alignment conventions. For implementation consistency and robustness, strict alignment requires full agreement at the level of the underlying inconsistency finding or robustness idea. Weak alignment additionally counts partial matches that capture the same underlying concern but differ in implementation details.

Prediction sets and denominators. Precision is evaluated over retained non-duplicate AIRE outputs, while recall is evaluated over reviewed reference groups. Computational reproducibility correspondence metrics are evaluated at the comparison level, implementation-consistency metrics at the level of inconsistency findings, and robustness metrics at the level of robustness ideas.

Table E.1 reports the denominators used for the main evaluation metrics. Denominators differ across tasks and metrics because AIRE outputs are evaluated at task-specific units and because model-identified, reference, and model-only outputs serve different roles in the scoring protocol. For paired entries, values are reported as GPT-5.5 / Opus 4.7.

Zero-denominator conventions. When no retained predictions exist despite positive reference cases, precision is set to zero by evaluation convention because the system failed to retrieve any relevant items. When neither retained predictions nor positive reference cases exist, precision is reported as missing because the metric is not evaluable. Diagnostic rates such as duplicate share are reported as zero when the system produces no candidate outputs.

F Additional Evaluation Results

This appendix reports additional evaluation indicators that complement the main-paper results. In particular, we report weak-alignment metrics, duplicate-rate diagnostics, granularity diagnostics, and unresolved-case shares for implementation consistency and robustness evaluation. Duplicate rate is computed before duplicate findings are removed from the retained prediction set.

Task	Metric	Unit	N
T1	Executability agreement	paper	74
T1	Match share	comp. (model)	5,325 / 5,663
T1	Match share	comp. (ref.)	6,143
T1	Comparison precision/F1	comp. (model)	5,220 / 5,631
T1	Comparison recall	comp. (ref.)	6,041 / 6,434
T1	Correspondence precision/F1	comp. (model)	5,325 / 5,663
T1	Correspondence recall	comp. (ref.)	6,143 / 6,464
T2	Strict precision	incons. (model)	43 / 19
T2	Strict recall	incons. (ref.)	36
T2	Valid-novel rate	incons. (model-only)	39 / 15
T2	Hallucination rate	incons. (model)	43 / 19
T2	Cross-model agreement	incons. (model)	43 / 19
T3	Strict precision	rob. (model)	272 / 272
T3	Strict recall	rob. (ref.)	297
T3	Valid-novel rate	rob. (model-only)	250 / 254
T3	Hallucination rate	rob. (model)	272 / 272
T3	Cross-model agreement	rob. (model)	272 / 272

Table E.1: Evaluation denominators. T1 = computational reproducibility, T2 = implementation consistency, and T3 = robustness-check generation. "Model" refers to retained, non-duplicate AIRE outputs; "ref." refers to reviewed reference annotations. For paired entries, values are reported as GPT-5.5 / Opus 4.7.

Metric	GPT-5.5	Opus 4.7
Weak precision	6.98	10.53
Weak recall	8.33	5.56
Weak F1	7.59	7.27
Duplicate rate	0.00	0.48

Table F.1: Additional implementation-consistency evaluation indicators. Weak precision and weak F1 are evaluated over retained, non-duplicate model-flagged inconsistencies: 43 for GPT-5.5 and 19 for Opus 4.7. Weak recall is evaluated over 36 reference inconsistencies. Duplicate rates are evaluated over 69 papers.

Metric	GPT-5.5	Opus 4.7
Weak precision	16.18	15.44
Weak recall	14.81	13.80
Weak F1	15.47	14.58
Duplicate rate	0.00	0.00

Table F.2: Additional robustness-check evaluation indicators. Weak precision and weak F1 are evaluated over 272 non-duplicate model-flagged robustness checks for each model; weak recall is evaluated over 297 reference robustness checks. Duplicate rates are evaluated over 69 papers.