Reliable Context

RELIABLE CONTEXT / RESEARCH & PRACTICAL GUIDANCE

AI Tool Output Verification: Did It Establish a Fact?

> Reliable Context is a kernel of truth: a system of record for organizations that fuse private and public information and must defend every value they act on. One canonical record per real thing, a cited source and an observation date behind every value, unknowns declared rather than left blank, and any past state of the record reproducible.

Published by Reliable Context · 2026-11-01 · ai tool output verification · agent evidence verification · tool calling evaluation · silent tool failures · source provenance · context engineering · evidence grounding · ai agent evaluation

Original article: https://reliablecontext.com/articles/ai-tool-output-verification-evidence-status



In this article

1. Executive Summary
2. Introduction and Background
3. What counts as a successful observation?
4. Build a fixture with an independent reference
5. Preserve the observation chain before evaluating claims
6. Decide when an observation supports a claim
7. Evaluate detection, assertion, and usefulness separately
8. Data Analysis and Evidence
9. Implications and Future Directions
10. Frequently Asked Questions (FAQs)
11. Conclusion

Executive Summary

AI tool output verification asks whether a particular observation permits a particular claim. A successful request, readable response, or matching schema is evidence about an interface contract. Establishing an external fact also requires checking the subject, source, observation time, coverage, and relevant error signals. The Hypertext Transfer Protocol (HTTP) defines request success; the pinned Model Context Protocol (MCP) specification separately defines protocol and tool-execution errors. Neither distinction should be silently promoted into a guarantee about the world. (rfc-editor.org) (modelcontextprotocol.io)

The report proposes an **evidence-admission policy** for read-only agents: classify each claim as **observed value**, **bounded not-found**, **blocked observation**, **failed observation**, **incomplete evidence**, or **unresolved conflict**. These are declared evaluation labels, not universal application programming interface (API) fields. An empty list alone establishes no general absence. Amazon DynamoDB documents an empty filtered query page that can still carry a continuation key; Elasticsearch documents partial or empty results after a search timeout. Both make coverage material to interpretation. (docs.aws.amazon.com) (elastic.co)

Published experiments motivate the distinction without establishing a production failure rate. Kraishan's (2026) pinned preprint reports **1,920 trials across 24 simulated tasks**, with detection of **91.3%** for explicit errors and **58.8%** for silent corruption. (arxiv.org) Its measurement and simulation boundaries limit generalization. Sun, Min, Chang and Bisk's **Tools Fail** studies controlled calculator and embodied-planning settings. (aclanthology.org) Zhu and colleagues' **ToolMaze** (2026) separately organizes perturbations by manifestation and persistence. (arxiv.org) Their results are not interchangeable estimates.

The practical recommendation is a contained fixture with a frozen, independent reference snapshot and a claim-by-observation ledger. Inject a recorded change at the tool boundary, verify that it reached the model, and grade detection, response choice, claim support, and retained usefulness separately. Groundedness against supplied text and comparison with independent truth answer different questions, as official evaluator documentation illustrates. (docs.cloud.google.com) No new experiments or performance figures were produced for this report. Its contribution is a proposed protocol for deciding what an agent may assert, when corroboration is necessary, and when the answer must remain unresolved.

Introduction and Background

A tool-using agent may retrieve a value accurately, receive an incomplete result, or preserve a plausible error in its final answer. The question for context engineers is narrower than whether the agent eventually completes a task: **what did this observation establish?** This report addresses that question at the boundary between a read-only tool result and an asserted proposition.

Interface validation is necessary but has a defined scope. JavaScript Object Notation (JSON) Schema, for example, permits omitted object properties unless the schema requires them. A payload can therefore satisfy a permissive contract while lacking information that the task needs. This is a documented property of validation, not evidence that any particular service returned a wrong answer. (json-schema.org)

Grounding also needs a declared reference. Google describes grounding as connecting model output to verifiable sources; Anthropic describes groundedness checks as testing support in retrieved sources. (docs.cloud.google.com) (anthropic.com) A faithfully reproduced source statement may still require a separate correctness check. Wikidata's documentation makes the distinction explicit: references identify where a value came from, and the referenced source can itself contain incorrect information. (wikidata.org)

The scope here is **read-only information tools and contained fixtures**. It excludes production side effects and operational retry strategies. The proposal concerns the eligibility of evidence for a claim, including useful answers that explicitly preserve uncertainty.

Reliable Context publishes from the position of a system-of-record vendor. That position supplies no independent evidence for the protocol's effectiveness. The comparisons below distinguish record keeping, interface validation, and reference-based evaluation without presenting a product benchmark. The research basis consists of opened primary papers, standards, and official technical documentation; the status taxonomy and experimental worksheet are this report's proposed methods.

What counts as a successful observation?

Define the proposition before judging the response

An **asserted fact** should be an explicit proposition: a named subject has a specified attribute or relationship, at a stated time and within a stated scope. **Evidence eligibility** means that the available observation supports that proposition under a declared policy. It does not mean that a model expressed confidence.

The following distinctions should remain separate:

- **Successful request:** The request met the relevant transport or protocol success condition. HTTP response semantics depend on the request method. (rfc-editor.org)
- **Completed observation:** The intended read finished without an unresolved execution error.
- **Complete bounded result:** The read covered the declared records, pages, permissions, and time boundary.
- **Observed value:** The source returned a particular value for the intended subject.
- **Admissible assertion:** The value can support the exact proposition after scope and source checks.

- **Independent correctness:** A reference outside the potentially faulty observation path supports the answer.

The last five definitions are proposed evaluation distinctions. Their purpose is to prevent a change of meaning between “the tool returned this” and “this is established.” The MCP specification supports a concrete interface distinction: tool-execution errors are carried in tool results through the execution-error mechanism rather than being interchangeable with protocol errors. (modelcontextprotocol.io)

Use statuses that preserve the limits of knowledge

Table 1 defines the proposed claim-status policy. The labels describe what an observation permits; they should be assigned to individual propositions rather than to an entire conversation.

Status	Required evidence boundary	Permitted assertion	Unresolved issue
Observed value	Intended subject, source version, and relevant scope are established.	The source reports the value, with observation date and any qualification.	External correctness may still require corroboration.
Bounded not-found	A completed search exhausted a declared, known coverage boundary.	No qualifying record was found within that boundary.	Nothing is asserted about unsearched sources.
Blocked observation	Access or permission prevented the intended read.	The requested fact could not be observed because access was blocked.	The underlying value remains unknown.
Failed observation	Execution or protocol failure prevented a usable observation.	The observation attempt failed, with the recorded reason.	Failure does not establish a negative fact.
Incomplete evidence	Some usable material exists, but coverage is partial or unverified.	State supported local findings and the missing boundary.	Exhaustive counts and absence claims remain ineligible.
Unresolved conflict	Admissible observations disagree about the same normalized proposition.	Report the competing values and their provenance.	No winner is declared without an additional rule or evidence.

The matrix is deliberately asymmetric. A positive local observation can sometimes survive incomplete coverage, while an exhaustive negative cannot. PostgreSQL offers a useful analogy: ordinary comparisons involving null can yield unknown rather than true or false. That database behavior is not an agent evidence standard, but it illustrates why unknown must remain distinguishable from false. (postgresql.org)

Build a fixture with an independent reference

Freeze truth and declare coverage

The proposed **reference snapshot** is an evaluator-controlled dataset, independently inspected before the agent runs. It should contain expected values, query predicates, version identifiers, and explicit coverage. The potentially faulty endpoint must not generate its own truth labels. National Institute of Standards and

Technology (NIST) guidance recommends comparing generated output with known ground-truth data; the frozen fixture operationalizes that principle for tool observations. (tsapps.nist.gov)

Include legitimate empty results as controls. A zero-match answer is appropriate when the fixture's complete search boundary is known and the query genuinely matches no record. An inaccessible source, a partial page, and a completed zero-match search should produce different reference labels even if an adapter renders their payloads similarly.

- **Snapshot:** Preserve the reference content and its version.
- **Question:** State the exact proposition and relevant observation date.
- **Coverage:** Declare included records, exclusions, permissions, and pagination.
- **Expected answer:** Establish it independently of the agent's trajectory.
- **Admissibility:** Specify which observations can support each claim.
- **Control:** Repeat the no-fault condition and inspect natural errors.

These are proposed requirements, not characteristics attributed to an existing benchmark. The controlled comparison adapts a broader experimental principle: the Principles of Chaos Engineering calls for measurable outputs and explicit control and experimental groups. Its production-oriented recommendations are not being adopted as a production-testing instruction here. (principlesofchaos.org)

Change one observation boundary

Keep the reference fixed and mutate a recorded response boundary. ToolMaze's design crosses explicit versus implicit manifestation with transient versus permanent persistence; its recovery-path framing is useful for separating these dimensions. This report borrows that design distinction without borrowing its task scores. (arxiv.org)

Table 2 translates the distinction into proposed read-only fixtures. Every row describes a **Hypothetical Example**, not a measured failure.

Hypothetical Example fixture	Controlled change	Reference truth	Evidence status after exposure
Timeout or explicit error	Replace the intended response with a recorded error.	Source values remain fixed.	Failed observation unless independent evidence supplies the fact.
Permission denied	Deny the intended read at the adapter boundary.	The reference remains available only to the evaluator.	Blocked observation, not bounded not-found.
Truncated response	Remove a result segment or continuation information.	Full reference coverage remains fixed.	Incomplete evidence if completeness cannot be established.
Plausible typed corruption	Substitute an incorrect value while preserving its expected type.	The independent expected value is unchanged.	Returned value is observable; endorsing the substituted fact is unsupported.
Silent dropped row	Remove a qualifying row while preserving the response schema.	The omitted record still exists in the reference.	Absence or exhaustive-count claims are unsupported.

Hypothetical Example fixture	Controlled change	Reference truth	Evidence status after exposure
Legitimate empty control	Apply no fault to a genuinely zero-match query.	The full bounded reference has no matching record.	Bounded not-found is eligible.

The table separates **what was changed** from **what the agent could know**. A harness may know that a row was removed while the agent sees no warning. That mismatch is precisely what the test must expose. ToolMaze describes implicit failures as structurally compliant but semantically flawed; schema-preserving mutations test that boundary directly. (arxiv.org)

Preserve the observation chain before evaluating claims

Record what arrived and what reached the model

The proposed observation record should preserve the raw response and the exact context delivered to the agent. Recording only the endpoint's response misses transformations performed by wrappers, filters, and context assembly. OpenTelemetry's fetched Generative AI registry documents message ordering and permits filtering or truncation; the page also marks the attributes as moved or deprecated, so it should not be treated as an unchanged implementation contract. (opentelemetry.io)

For each observation attempt, retain:

- **Request identity:** Tool name, adapter version, call identifier, and parameters.
- **Time boundary:** Request time, observation time, and source snapshot version.
- **Transport outcome:** Status and any protocol-level error.
- **Execution outcome:** Tool-error flag and application-level error content.
- **Payload evidence:** Raw bytes or lossless content with a stable locator.
- **Coverage evidence:** Pagination state, truncation markers, and declared completeness.
- **Transformation chain:** Any extraction, normalization, filtering, or shortening.
- **Delivered context:** The exact tool message and relevant surrounding context.
- **Fault evidence:** Assigned mutation, activation event, and confirmed model exposure.
- **Reference link:** Independently established answer and admissibility label.

This record is a proposed evaluation artifact. World Wide Web Consortium (W3C) PROV supplies a conceptual foundation for representing entities, activities, and people involved in producing information. Its domain-agnostic model offers extension points; it does not prescribe the particular ledger fields above. (w3.org)

Inspect wrappers and completeness signals

An adapter should be checked for whether it retains, strips, or remaps MCP's **isError** signal. (modelcontextprotocol.io) The specification pinned here to **2025-06-18** requires structured results to conform to an output schema when supplied and recommends client validation. (modelcontextprotocol.io) Those are contract checks; testing their preservation is separate from certifying external truth.

Completeness needs its own evidence. GitHub documents pagination as returning a subset of results and notes that a last-page link may be unavailable when it cannot be calculated. A missing last-page link is therefore not a general proof of exhaustion. (docs.github.com) Elasticsearch separately exposes failed shards and documents partial responses following shard timeouts or failures. (elastic.co)

The practical admission rule is to bind an aggregate claim to a coverage record. “These returned records have this property” and “all qualifying records have this property” are different propositions. A schema check over the returned list cannot, by itself, establish that an unreturned qualifying row does not exist.

Decide when an observation supports a claim

Separate quotation, grounding, and correctness

An agent can accurately report that a source returned a value without having enough evidence to endorse that value as externally correct. Google Cloud's grounding metric checks factuality and consistency against supplied source text, while its computation-based metrics require ground truth in the dataset. This distinction makes the reference boundary explicit. (docs.cloud.google.com)

The proposed admission decision should ask:

- **Identity:** Does the observation concern the intended real thing?
- **Predicate:** Does the returned field answer the actual question?
- **Time:** Is the proposition aligned with the source's effective date?
- **Coverage:** Does the observation cover the claim's quantifier?
- **Failure state:** Is an error being hidden by a success wrapper?
- **Source quality:** Is the source eligible under the declared policy?
- **Dependence:** Does corroboration reach genuinely separate evidence?
- **Conflict:** Are apparently competing values comparable in scope and date?

A repeated call to the same faulty endpoint should count as another attempt, not automatically as independent confirmation. Likewise, separate URLs can reproduce the same upstream statement. The ledger should preserve derivation relationships so that apparent agreement is not counted twice. W3C PROV explicitly supports provenance modeling and trust judgments, providing the basis for representing those relationships rather than assuming independence from presentation. (w3.org)

Compare responsibilities without confusing them

Table 3 compares evidence-related approaches. It is a responsibility map, not a ranking or a set of measured product capabilities.

Approach or position	Relevant contribution	Additional evidence needed
JSON Schema validation	Checks declared structural constraints; omitted properties can be valid unless required. (json-schema.org)	Source correctness, identity, and coverage.
Groundedness evaluation	Tests consistency with supplied evidence under the documented metric. (docs.cloud.google.com)	Independent correctness and source completeness.

Approach or position	Relevant contribution	Additional evidence needed
AgentRx diagnostics	Uses guarded executable constraints derived from schemas and policies. (microsoft.com)	An external reference for the particular factual proposition.
Independently checked fixture, proposed here	Supplies frozen reference answers and coverage for contained tests.	Evidence of transfer to real services.
Reliable Context, publisher position	System-of-record vendor; no product performance is evaluated here.	Independent evidence for any effectiveness claim.

Each approach answers a different question. Microsoft's AgentRx checks constraints when guard conditions apply and uses evidence-backed diagnostic logs. Interpreting that as failure localization is appropriate; interpreting it as certification of every external source would require a separate evaluation. (microsoft.com) The proposed fixture similarly establishes support within its declared reference boundary rather than certifying an entire information ecosystem.

Evaluate detection, assertion, and usefulness separately

Use a claim-by-observation ledger

The **claim ledger** is the proposed unit of adjudication. Store the exact proposition, subject, value, date, raw-result locator, source chain, admissibility decision, corroboration, final status, and downstream use within the contained test. Preserve the initial response beside any correction rather than replacing the record of what occurred.

Wikidata's documentation distinguishes references from rank and warns that normal rank supplies no evaluation of a value's accuracy or currency. The relevant lesson is to preserve distinct evidence properties, rather than treating a stored value or a source link as an automatic endorsement. (wikidata.org)

Grade separate dimensions:

- **Detection:** Did the observable response identify the problem?
- **Response choice:** Did the agent appropriately qualify, corroborate, or stop the claim?
- **Final claim status:** Is the asserted proposition supported by admissible evidence?
- **Independent correctness:** Does the final answer match the fixture reference?
- **Useful performance:** Did the agent still provide the answerable parts?
- **Effort:** Was that result achieved within the declared evaluation budget?

LangChain's evaluation documentation distinguishes step-level inspection, trajectory comparison, and reference-based final-response evaluation. These are complementary measurement locations, not interchangeable guarantees. (docs.langchain.com)

An agent may notice an error yet repeat its value, or provide a correct answer without saying it noticed. Transcript silence establishes only the absence of an observable report under the coding rule. It cannot reveal internal recognition. Evaluation should therefore preserve the transcript and the answer score as separate artifacts.

Protect clean performance and held-out validity

Anthropic distinguishes trials from tasks and separates the transcript from the final environment outcome. (anthropic.com) For this read-only proposal, the answer and its evidence status need independent checks because a final state alone may not capture an unsupported statement.

Policy development should reserve fresh fixture families. Scikit-learn's grouped validation guidance prevents the same group from appearing in both training and testing. Applying the principle here means keeping related source templates or task families together, rather than distributing near-duplicate fixtures across development and evaluation. This is a methodological adaptation, not a documented agent benchmark result. (scikit-learn.org)

Compare a changed policy with the baseline on paired questions under matching fault families and effort caps. Report useful answers under clean controls beside fault-conditioned performance. Otherwise, a policy that refuses every question can appear conservative while failing the task.

Data Analysis and Evidence

What the published studies establish

Kraishan's pinned preprint analyzes **1,694 of 1,920 assigned trials** after excluding inactive faults. (arxiv.org) Reported detection is **91.3%** for explicit errors and **58.8%** for silent corruption. (arxiv.org) Detection is coded from written responses; clean-run end states agree only **63.3%** of the time. (arxiv.org) Trials are capped at **15 tool calls**, with **27%** reaching the cap. (arxiv.org) Clean conditions can contain genuine environment errors, so their problem reports are not a universal false-alarm estimate. (arxiv.org)

The author repository describes reproduction from released trajectories and cached judge codings and labels the work “to appear.” Repository inspection here did not reproduce the results. (github.com)

Sun and colleagues' **Tools Fail**, published at **EMNLP 2024**, includes **300 correct calculator samples** alongside perturbed cases and tests disclaimer, confidence-score, and checklist interventions. (aclanthology.org) (arxiv.org) ToolMaze's **June 2026** pinned preprint separates perturbation recovery from final task success. (arxiv.org) These studies supply different experimental questions, not a pooled prevalence estimate.

Report denominators before rates

The proposed protocol reports **activated faults divided by assigned trials**, followed by **model-exposed outputs divided by activated faults**. Nonactivated assignments remain in the activation report. Fault-conditioned outcome rates should use only genuinely exposed, adjudicable trials.

Define **unsupported assertion rate** as:

Unsupported assertion rate = exposed adjudicable trials containing an unsupported proposition / all exposed adjudicable trials.

Also report the unsupported-assertion count over all assigned trials, accompanied by exposure and adjudicability counts. This reveals activation selection rather than hiding it. Keep claim-level counts separate when trials contain different numbers of propositions.

Use additional denominators:

- **Correct uncertainty:** Eligible unresolved cases receiving an accurate uncertainty statement / adjudicable unresolved cases.
- **Unnecessary refusal:** Answerable cases refused despite admissible evidence / adjudicable answerable cases.
- **Bounded negative accuracy:** Correct bounded not-found answers / complete zero-match controls.
- **Clean usefulness:** Successfully answered clean tasks / adjudicable clean tasks.
- **Post-fault usefulness:** Supported useful answers after exposure / adjudicable exposed tasks.

These are proposed metrics, with no measured values claimed. An unknown correctly labeled as unknown must not be scored as a factual negative.

Quantify uncertainty at the right unit

Resample task or source families for clustered intervals and retain the pairing between policy variants. SciPy's paired bootstrap uses matching sampled indices across arrays, but that option alone does not implement family-level clustering; the evaluator must define and resample those units explicitly. (docs.scipy.org) SciPy documents bootstrap confidence intervals as estimates of uncertainty in a statistic. (docs.scipy.org)

Publish missing observations, excluded cases, effort caps, and coding disagreements alongside rates. A narrow interval over repeatedly sampled near-duplicates does not establish generalization to independent services.

Implications and Future Directions

The immediate design implication is an **evidence contract** that can express blocked, failed, incomplete, conflicting, and bounded-negative observations without inventing a value. The contract should constrain what a claim means and what observation supports it. It need not prescribe the model's internal reasoning.

A useful next evaluation would compare the unchanged agent with a policy that explicitly binds assertions to observation status. The comparison should use the same reference snapshots during development and fresh held-out source families for the final test. Group-isolated validation supplies a relevant methodological precedent, while leaving the agent-specific design to the evaluator. (scikit-learn.org)

Real services present a harder boundary than closed fixtures. Source coverage may be unknown, permissions may hide relevant material, and an independent oracle may be unavailable. DynamoDB's documentation illustrates a narrower but concrete caution: a nonempty continuation key does not necessarily mean that more data remains. Coverage decisions require the endpoint's semantics, not a generic interpretation of a token. (docs.aws.amazon.com)

Provenance supports review but should not be mistaken for correctness. The W3C overview distinguishes valid provenance structures from the judgments those structures help users make. (w3.org) NIST's guidance on tracing origins and modifications similarly motivates retaining the observation chain; it does not supply a truth oracle for a changing external source. (tsapps.nist.gov)

The research opportunity is therefore specific: test whether explicit evidence status reduces unsupported claims while preserving supported useful answers. Publish the worksheet, source snapshot, harness commit, adapter version, model settings, and coding policy with the results. Until those experiments are run, claims of improvement remain hypotheses.

Frequently Asked Questions (FAQs)

How should an AI agent verify a tool output?

Check the request and execution outcomes, then verify subject, date, source, and coverage before admitting the claim. Preserve the raw response and use an independent reference when correctness is being evaluated. The pinned MCP specification recommends client validation of structured results, but that is one component of this broader proposed admission policy. (modelcontextprotocol.io)

Does an empty tool result mean nothing exists?

Only a completed observation over a declared complete boundary can support bounded not-found. An empty filtered DynamoDB page can carry a continuation key, demonstrating why payload emptiness alone is insufficient. (docs.aws.amazon.com) Even an empty object can validate against a JSON Schema example without required properties. (json-schema.org)

How can an evaluator detect incomplete results?

Inspect pagination, declared completeness, timeout and shard signals, then compare with the independent fixture reference. GitHub's pagination documentation explicitly describes returning a subset, while Elasticsearch documents partial or empty timed-out results. (docs.github.com) (elastic.co)

Is a grounded answer necessarily a verified fact?

Groundedness assesses support against a reference source under the chosen metric. Independent correctness is a separate question. When evidence is insufficient, the proposed policy preserves unknown; Anthropic also recommends allowing an Unknown result in model-based grading. (anthropic.com)

Conclusion

AI tool output verification should end with a decision about a proposition, not merely a successful interface check. The proposed policy preserves the distinction between an observed value, bounded not-found, a blocked or failed observation, incomplete evidence, and unresolved conflict. Those labels make the limits of an observation explicit.

The essential evaluation artifact is a traceable claim-by-observation ledger tied to a frozen independent reference. It records the requested fact, what the source returned, what reached the model, whether the fault activated, and what the final answer asserted. Detection, response choice, evidence support, correctness, and useful performance remain separate outcomes.

A legitimate empty result deserves credit when coverage is complete. An inaccessible or incomplete observation deserves an accurate uncertainty statement. Neither should be forced into a fabricated value or an unqualified negative. Corroboration should reach independent evidence rather than merely repeat an observation path.

The published research supports investigating this boundary, with simulation, activation, measurement, and effort limits kept visible. This report proposes a reproducible evaluation method; it reports no new benchmark. The decision rule is concrete: admit the claim when the available evidence supports its exact scope, qualify it when only a bounded statement is supported, and preserve the unresolved status when the observation cannot establish more.

Source links

Links cited in this article, in order of first appearance. Full addresses are provided for readers using extracted text.

1. https://www.rfc-editor.org/rfc/rfc9110.html
2. https://modelcontextprotocol.io/specification/2025-06-18/server/tools
3. https://docs.aws.amazon.com/amazondynamodb/latest/APIReference/API_Query.html
4. https://www.elastic.co/docs/api/doc/elasticsearch/operation/operation-search
5. https://arxiv.org/html/2610.10062v1
6. https://aclanthology.org/2024.emnlp-main.790
7. https://arxiv.org/html/2606.05806v1
8. https://docs.cloud.google.com/gemini-enterprise-agent-platform/models/determine-eval
9. https://json-schema.org/understanding-json-schema/reference/object
10. https://docs.cloud.google.com/gemini-enterprise-agent-platform/models/grounding/overview
11. https://www.anthropic.com/research/demystifying-evals-for-ai-agents
12. https://www.wikidata.org/wiki/Help:Ranking
13. https://www.postgresql.org/docs/current/functions-comparison.html
14. https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=958388
15. https://principlesofchaos.org
16. https://opentelemetry.io/docs/specs/semconv/registry/attributes/gen-ai
17. https://www.w3.org/TR/prov-overview
18. https://docs.github.com/en/rest/using-the-rest-api/using-pagination-in-the-rest-api
19. https://www.w3.org/TR/prov-dm
20. https://www.microsoft.com/en-us/research/blog/systematic-debugging-for-ai-agents-introducing-the-agentrx-framework
21. https://docs.langchain.com/langsmith/evaluate-complex-agent

22. https://scikit-learn.org/stable/modules/cross_validation.html
23. https://github.com/obadaKraishan/brittle-agents
24. https://arxiv.org/html/2406.19228v1
25. https://arxiv.org/abs/2606.05806v1
26. https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.bootstrap.html
27. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GroupKFold.html

THE PUBLISHER

About Reliable Context

Reliable Context is a system of record for organizations that fuse private and public information and must defend every value they act on.

The guarantees

One canonical record per real thing: duplicates resolve by rule, ambiguity is shown rather than hidden, and related organizations relate without ever being merged. A verbatim, cited source and an observation date sit behind every value. Unknowns are declared as not found, blocked or not applicable with a reason, never left as an empty cell. Any past state of the record is reproducible, together with what changed since.

Where it runs

It runs inside the customer's perimeter, one database per organization. Permitted sources, model routing, reader clearance and retention are set as the customer's own policies, and every value is exportable with its chain of custody for an auditor or regulator.

What it is not

It is not a vector database or a RAG framework, not a data catalog, MDM or data-quality suite, not a CRM or data-enrichment vendor, and not a chatbot or AI assistant. Where a reader needs one of those, the honest answer is to say so and point them there.

Visit

See Reliable Context.

Reliable Context is in production use in two sectors; more than six thousand of its identity decisions were made by rule and none has been reversed. Articles published here explain general practice, cite their primary and academic sources, and name the jurisdiction of any regulatory rule they describe. They are not legal, regulatory or compliance advice for a specific organization, and requirements change.

Website: https://reliablecontext.com

This article is educational. Verify consequential medical, legal, financial, regulatory and technical decisions with appropriate professionals and the cited primary sources. Company services are described separately from the article's research findings.