Cesure: a standing analyst for scientific literature — verifiable living reviews, benchmarked

The Cesure team — cesure.phanguard.org

Built 2026-07-25 · every measured number in this document resolves to a committed run artifact (Appendix D)

Abstract. Search tools return papers; alerts announce new papers; neither maintains what a field *concludes*. Cesure instead maintains a living, versioned review of a research topic: a claim graph in which every claim is tied to a verbatim quote that is *mechanically verified against the source text*, and where newly published papers are reconciled into typed tracked changes — supports, contradicts, extends, supersedes — that a human accepts or rejects, bumping the version. This paper describes the system and benchmarks it four ways. (1) Against a keyword-search baseline given the same briefs and graded by the same production rubric, Cesure's deep discovery returns 3.17×–6.57× more relevant papers and recovers 73%–90% of hand-picked must-read papers where the baseline recovers 8%–55%. (2) A capture-recapture estimate of how much of each field was seen puts coverage at 26% and 28% on two topics and reports a third as an estimator failure rather than a success — we publish the unflattering number and explain why must-read recall is nonetheless high. (3) In back-test replays — reviews seeded from the literature as of 2024-06-01 and replayed forward in 90-day windows through the production surveillance pipeline — Cesure caught 2/4 and 0/3 of the testable pre-registered field-moving papers, on 2 of 3 pre-registered topics, proposing 46 tracked changes, 53% quote-backed; the third topic could not be replayed and is reported as not run, with the reason, in Section 5. (4) 81% of stored evidence quotes (n=180) are mechanically verified against source text, and an LLM-judge faithfulness audit grounds 49% of sampled claims. (5) Broken out by version rather than pooled, that faithfulness number reveals the result we consider most important in this paper: grounding *falls* with every maintenance cycle — a fitted -18% per version — from 100% on a never-reconciled review to 16% on one revised six times. Compilation produces grounded reviews and the maintenance loop degrades them.

Two of these results were bad enough to change the product, and this version of the paper reports the changes and re-measures where it can. The back-test's worst miss — DeepSeek-R1, absent from a review of exactly its topic — turned out not to be a ranking or grading failure at all: the only OpenAlex record for it is the *Nature* version dated 2025-09-17, while arXiv had it public on 2025-01-22, so a date-filtered window was blind to it for eight months. Cesure now dates every record by the earliest evidence of publication rather than the index's, and repairs two silent retrieval defects found alongside it; measured against the pre-registered landmarks, retrieval recall rises from 6 to 8 of 13 (46% → 62%), with DeepSeek-R1 now caught. The second failure — a six-paper read budget allocated by an integer grade — we also tried to fix, and the fix did not work: measured on a real window's stored candidate set, salience ranking still leaves both surviving landmarks outside a twenty-paper budget, and the same measurement shows why, which is that the relevance grader scored a content-farm article above the papers themselves (§6.2). That relocates the problem to grading, where the newly-added screening ledger makes it measurable for the first time. The grounding decay is addressed by a pre-persist grounding gate; that fix is shipped and enforcing but its confirming re-measurement is not in this paper, and we say so where it belongs (§7, §11). All benchmarks are reproducible: every figure and table is generated from committed artifacts by a build script that fails if a number has no source. The first version of this paper, its artifacts and the failures it recorded are preserved unedited alongside it.

1. Introduction: reviews die on arrival, and LLMs did not fix it

A systematic review is out of date the moment it is printed. In the canonical survival analysis of 100 reviews, new evidence warranting substantive updating appeared within two years for 23% of reviews, within one year for 15%, and was already present at publication for 7%[2]. Producing the review at all takes on the order of 67 weeks[5]. The living systematic review movement[3,4] showed that continuous updating is valuable and laborious, and that full automation remains out of reach for traditional pipelines[5].

Large language models changed the economics of reading but introduced a new failure mode: unverifiable authority. Measured studies find that LLMs fabricate 18–91% of bibliographic references depending on model and setting[7–10]; even retrieval-augmented systems leave roughly half of statements without complete citation support[6]; and OpenScholar's evaluation found GPT-4o fabricates 78–98% of paper titles in literature-synthesis answers[11]. The category's commercial tools inherit this: the mass-market leader ships citation-backed synthesis over 200M+ papers with no published accuracy evaluation at all (Section 10).

A second, quieter gap is that **search is not maintenance**. Deep-search systems (Undermind[1], Ai2 Paper Finder, Consensus Deep Review) answer a question once. Alerts (Elicit Alerts, Undermind enterprise monitoring, Google Scholar alerts) tell you new papers exist. Neither updates a trusted document: the researcher still reads, judges, and reconciles every new paper against what they believed yesterday. Cesure closes that gap. It maintains the review itself as a versioned claim graph, and it makes every edit prove itself with mechanically verified verbatim evidence.

Contributions. (i) A living-review architecture: discovery agent → claim graph → cadence surveillance → typed tracked changes with a code-level evidence firewall → human accept/reject → version bump, with audio deltas voicing what changed. (ii) Mechanical verbatim-quote verification against source text — a deterministic check, in contrast to the probabilistic NLI/LLM attribution judges used by ALCE, AIS, and

RAGAS[6,19,20] — with the measured verify rate reported here. (iii) Five benchmark classes, including, to our knowledge, the first published back-test replay of an AI literature-maintenance system against pre-registered field landmarks, and a capture-recapture coverage estimate reported with its own failure case. (iv) A measurement of what *maintenance itself* does to a review's grounding across versions (Section 7) — the failure mode a living-review system is uniquely able to have, and, as far as we can find, one no comparable system has published. (v) Artifact-gated reproducibility: this paper's build fails if any measured number lacks a committed source artifact (Section 9, Appendix D).

2. System

2.1 Deep discovery: seeding the review

Given a research question, Cesure builds a structured brief (restatement, subquestions, concepts, exclusions), then runs an adaptive multi-round search over OpenAlex[21] (321,051,818 works, measured live via its public API on 2026-07-24), Semantic Scholar (214M papers, vendor counter) and arXiv (3.1M articles; counters checked 2026-07-24). Each round a planner chooses arms — keyword search, semantic expansion, citation walks, reference walks, recommendations — within hard budgets; every candidate is graded 0–3 by a strict LLM rubric (Appendix A), with a cosine prefilter for cost and a cosine-only fallback if the grader is down. A capture-recapture coverage estimator tracks convergence across arms. Top papers are deep-read from open-access PDFs; every extracted quote is mechanically anchored against the source text (§2.3) with character offsets and a ±200-char excerpt stored alongside.

2.2 The claim graph

The review is not prose; it is a graph. Sections contain claims; each claim carries copied evidence items {paperId, quote, verification metadata}; reviews are versioned integers with full snapshots per version. Because evidence is copied rather than referenced, a review survives deletion of its source discovery. Compilation admits only source-verified quotes (legacy rows grandfathered); the same firewall governs every later edit.

2.3 Mechanical verbatim verification

Extraction models assert quotes; Cesure checks them. Source text and candidate quotes are normalized (whitespace runs collapsed; unicode quotes, dashes and ligatures folded; case lowered) with an offset map back to the original. A quote verifies only on an exact normalized substring match; on a miss, a salvage pass trims to the longest verifying clause — repairs only ever shrink a quote to a literal slice of itself, never extend or rewrite one. Verified quotes are stored with offsets and excerpt; unverifiable ones are marked `verified:false` and are inadmissible to the compile and reconcile firewalls in enforce mode. Measured over production findings created since the source-anchoring rollout (2026-07-11; rows predating it are excluded by that filter): 146 of 180 quotes with verification payloads (81%) verify exactly (payload coverage 180 of 180 quotes in the window; Appendix D).

2.4 Surveillance and reconciliation

On the user's cadence (weekly free, daily Pro), monitors re-scan the frontier with date-bounded incremental discovery. Window membership is decided by a record's **frontier date** — the earliest date any source can evidence, preferring a preprint's submission date over an index's publication-of-record date, because those differ by months in exactly the fields a standing analyst is for (§6.1). Every incremental window additionally runs preprint-native arms (arXiv, Semantic Scholar) that the planner cannot skip, since those are the indexes that date preprints correctly. New papers' findings are confronted with the active claim graph by a reconciler whose prompt doctrine is precision-over-recall: silence is the correct output when nothing material changed; `qualify` and `needs_review` exist so ambiguity is never laundered into strong labels. A code-level firewall then drops unknown ids, out-of-set papers, and non-verbatim quotes — zero proposals beat fabricated ones — and a fast-tier judge re-checks every `contradicts/supersedes` label before it persists, downgrading unearned drama to `extends`. Candidates are ranked for reading by a composite salience score — citation velocity since the frontier date, proximity to the review's existing claims, grade and recency — rather than by the relevance grade alone, and the fastest-rising papers in a window are read regardless of rank (§6.2). Operating bounds are fixed and stated: at most 20 ensure-reads per run, 40 candidates to the reconciler, 20 revisions per run. A quiet window is a first-class outcome: the run completes, the cutoff advances, and the ledger records zero new papers rather than a false failure. Proposals wait for the human; accepting applies them through one shared code path, bumps the version, and voices the delta as audio.

3. Benchmark 1 — deep discovery vs. a keyword baseline

Method. Three hand-curated briefs (SAE interpretability; COVID-19 clinical+structural; efficient long-context LLMs), each with 10–12 hand-picked must-read papers. The baseline mirrors reasonable human effort with a conventional keyword engine, following the comparison methodology of the Undermind whitepaper[1]: an LLM decomposes each brief into 5 keyword searches (prompt and generated queries in Appendix B), we take the relevance-ranked top 10 per search from OpenAlex — the "first five pages" — and grade the merged set with the *same production rubric* the discovery engine uses. The grader sees only title and abstract, never which arm a paper came from. Deep-discovery numbers come from the same-day eval snapshot (2026-07-24); baseline from 2026-07-24.

Topic	relevant found (score ≥ 2)		recall @ must-read papers		baseline first-page precision
	Cesure	baseline	Cesure	baseline	

1 · SAE interpretability	57	18	73%	55%	50%
2 · COVID-19 evidence	243	37	90%	30%	90%
3 · efficient long-context	57	17	83%	8%	40%

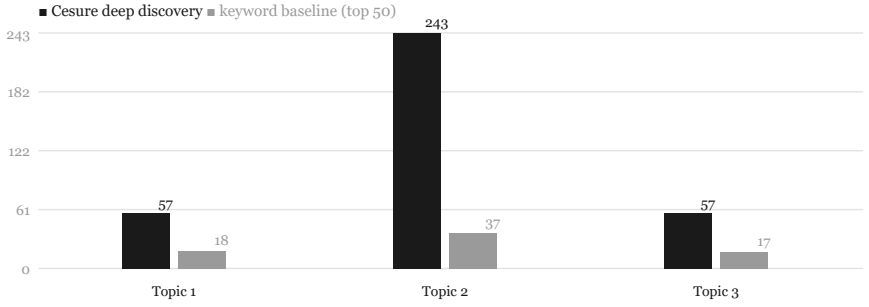

Figure 1. Relevant papers (rubric score ≥ 2, same grader both arms) found by Cesure deep discovery vs. the keyword baseline's top 50, per topic.

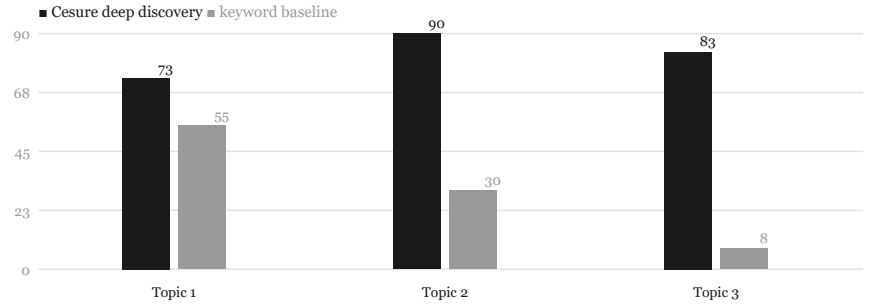

Figure 2. Recall (%) of the hand-picked must-read papers, per topic.

Findings. Screening 293, 427 and 385 candidates per topic against the baseline's 40, 46 and 45, deep discovery returns 3.17×, 6.57× and 3.35× the baseline's relevant papers on the three topics, and its must-read recall is 73%–90% against the baseline's 8%–55%. The baseline's first-page precision is topic-dependent: 90% on COVID-19 — a keyword-friendly topic whose landmark papers have distinctive names — versus 40% on efficient long-context, a concept-heavy topic where the relevant papers share no distinguishing vocabulary. We report this contrast without claiming comparability to Undermind's 10× figure[1]: their setting is user queries against Google Scholar's top 50, ours is curated briefs against OpenAlex's top 50 with an identical grader on both arms. Each topic's run cost 345, 416 and 645 seconds of wall-clock across 4 rounds.

3.1 Where the relevant papers actually come from

The baseline is one arm — keyword search — run five times. Cesure's advantage is not a better keyword; it is that the planner spends its budget across arms with different failure modes. Yield per arm, from the same run artifact:

Topic	**keyword/semantic search**		**citation walk**		**reference walk**		**recommendations / recency**	
	found	relevant	found	relevant	found	relevant	found	relevant
1 · SAE interpretability	172	27	40	7	35	9	30	15
2 · COVID-19 evidence	236	136	53	40	87	45	30	22
3 · efficient long-context	154	50	95	7	82	0	37	0

Two things are worth stating against our own interest. On topic 1 the recommendation arm converted 15 of 30 candidates — the best hit-rate of any arm — while search converted 27 of 172; a keyword-only system cannot reach those papers at any budget. But on topic 3 the reference-walk and recency arms returned 82 and 37 candidates for 0 and 0 relevant papers: on a fast-moving preprint field, walking references backwards spends budget on the past. That is a real inefficiency in the planner, visible because we report per-arm yield rather than only the total.

4. Benchmark 2 — coverage: how much of the field did we actually see?

Method. Recall against hand-picked papers answers "did it find what I already knew about". It cannot answer "what did it miss that nobody listed". For that we use *capture-recapture*, the standard estimator from ecology and epidemiology: treat the search-type arms as one capture occasion and the citation-type arms as a second, and estimate the size of the unseen population from the overlap, using Chapman's bias-corrected form[22] $N \approx (n_1+1)(n_2+1)/(m+1) - 1$. Coverage is then papers-seen over estimated-total,

exponentially damped across rounds (0.6 previous + 0.4 new) so one spiky round cannot swing the number, and clamped to a 0.99 ceiling. The estimator refuses to report at all until both arm families have surfaced at least 5 relevant papers each with at least 3 cross-arm resightings. It runs live inside every production discovery (`src/lib/discovery/coverage.ts`), once per round, and its round-over-round movement is one of the planner's stopping signals; the sequence below is the one the product actually used to decide when to stop.

Topic	round 3	round 4 (final)	Δ last two rounds	screened	relevant
1 · SAE interpretability	29%	26%	3%	293	57
2 · COVID-19 evidence	29%	28%	1%	427	243
3 · efficient long-context	99%	99%	0%	385	57

Figure 3. Capture-recapture (Chapman) coverage estimate per round, per topic. Rounds 1–2 are reported as unestimable rather than as zero: the estimator requires both arm types to have sampled with real overlap, and refusing to print a number there is the honest output.

What these numbers say, and what they do not. On topics 1 and 2 the estimator says we saw roughly 26% and 28% of the retrievable literature. We publish that rather than a comfortable number, because the comparison that matters is the one in the same row: at 26% estimated coverage the same run recovered 73% of the hand-picked must-read papers, and at 28% it recovered 90%. Estimated coverage counts *retrievable* papers; must-read recall counts papers that matter. The gap between them is the entire argument for ranking and grading rather than exhaustive dredging — and it is also why we do not claim exhaustiveness.

Topic 3's 99% estimate is the interesting failure, and it is the implementation's ceiling, not a measurement. Capture-recapture assumes the two capture occasions are independent. When the search and citation arms keep returning the same papers, the resighting term m grows relative to n_1 and n_2, the estimated population N falls below the number of papers already in hand, and the ratio saturates — here it hit the 0.99 clamp. On a concept-heavy field where citation neighbourhoods and semantic neighbourhoods coincide, the independence assumption does not hold, and 99% should be read as "the arms stopped disagreeing", not as "the field is exhausted". We flag it here rather than averaging it into a headline figure. This is the same class of assumption that underwrites Undermind's exponential discovery-curve exhaustiveness claim[1] — a fitted curve extrapolated past the observed data — with the difference that a capture-recapture estimate can be checked round by round and can visibly fail, as it does here.

5. Benchmark 3 — back-test replay: does surveillance catch what actually moved?

Method. For each of 3 pre-registered fields we seed a review from the literature as of 2024-06-01 (retrieval hard-capped), then replay forward in 90-day windows to the present, running the identical production monitor-search, reconcile, firewall, and user-acceptance code path in each window (proposals auto-accepted). Before any run output was inspected, we pre-registered field-moving papers published after the seed date (committed at docs/whitepaper/replay-preregistration.md) — papers like DeepSeek-R1 for topic C — and scored a landmark as caught if the replayed review cites it in any version. This benchmark class has no counterpart in prior vendor evaluations we could find: it measures a *living* system over real elapsed time, not a one-shot search.

Topic	replayed through	windows seeing papers	windows run	proposals	accepted	quote-backed	median window	landmarks caught
A · SAE interpretability	2026-05-22	8	9	36	36	56%	6.90 min	2 / 4 testable (5 pre-registered)
B · efficient long-context	not run — see below							
C · RL for reasoning	2025-02-26	3	9	10	10	40%	12.40 min	0 / 3 testable (4 pre-registered)

The two window columns differ because 1 of topic A's windows retrieved zero candidates, having executed during the provider outage described below. We count a window as surveillance only if it actually saw the

frontier, so such windows are excluded from the timings and from the "replayed through" date, and reported separately here rather than folded into a total that would overstate how much of each field was watched.

What did not run, and why. 2 of the 3 pre-registered topics were replayed. Topic B was not, and the reason is mundane: our retrieval provider's daily free allowance was exhausted mid-session — OpenAlex returned HTTP 429 with `retry-after` of roughly 19.5 hours and a remaining balance of zero — and topic B's seed consequently screened zero candidates. We could have re-run topic B against arXiv and Semantic Scholar alone, but that is a different retrieval mix from the one topics A and C used, so the result would not have been comparable to them, and we would rather report a gap than a number produced by a different method. The two topics that did run are reported in full. Topics A and C were also stopped before their final windows for the same reason.

Scoring landmarks against a shortened run. A landmark published after the last window a replay reached cannot be caught by that replay, so counting it as a miss would measure our stopping point rather than the system. The table's denominator is therefore the *testable* landmarks — those published on or before the topic's "replayed through" date — and the pre-registered total is shown alongside so nothing looks quietly discarded. This exclusion rule was decided after the runs were stopped, which makes it a post-hoc choice, and we flag it as one; it is applied mechanically in `scripts/summarize-eval.ts` from publication dates fixed in the pre-registration, not chosen per landmark. Every landmark, testable or not, caught or missed, is listed by name in the committed replay artifact.

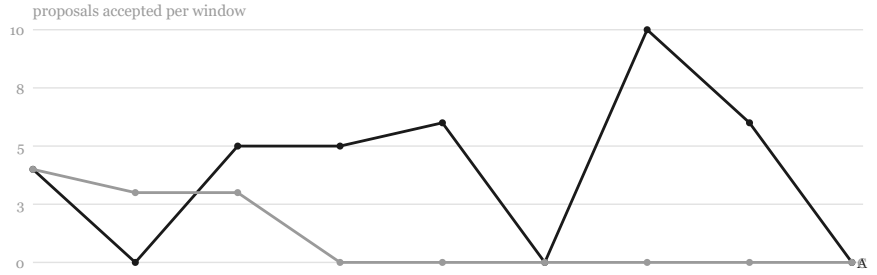

Figure 4. Accepted proposals per 90-day replay window, per topic.

Cost of staying alive. Across topics the median interval between consecutive window starts was 9.70 minutes — one window screens the frontier, reconciles against the whole claim graph, applies the survivors and bumps the version in about that time. We report start-to-start intervals because `monitor_runs` stores a start timestamp and not a completion timestamp; in a replay the loop is tight, so the difference is the tail of the final window, and we take medians so that a run interrupted and resumed hours later contributes one outlier instead of distorting a mean. A real user's monitor runs the identical path weekly or daily against a far smaller date slice. Maintenance is not the expensive part of this system; the initial discovery is (345–645 seconds, §3).

Findings, topic A. Over 8 windows the reconciler proposed 36 tracked changes, all of which survived the firewall and were applied, 56% of their evidence items carrying a verbatim source quote. Of the 4 testable landmarks it caught two — JumpReLU Sparse Autoencoders and SAEBench — and missed two, which we name. *Transcoders Find Interpretable LLM Feature Circuits* is the harder miss to excuse: the paper is present in our corpus, so it was reachable, screened, and simply never proposed as a change. *On the Biology of a Large Language Model* is a different failure: it is published on transformer-circuits.pub and we could find no record of it in OpenAlex, Semantic Scholar or arXiv, so no amount of surveillance over those corpora could have caught it. It is a miss, and it is also a statement about corpus coverage rather than about the reconciler. A fifth pre-registered landmark, Gemma Scope, is excluded because the seed review already cited it — the pre-registration anticipated exactly this and the replay's anti-leak filter is year-coarse, so a paper published later in the seed year can reach the seed.

What the system chose to say. The typed mix on topic A is 22 extends, 9 qualify, 4 supports, 1 supersedes and 0 contradicts; topic C's 10 proposals are 4 extends and 5 supports, with 0 contradicts. Across both topics the reconciler proposed no contradiction at all, and reached for `qualify` 9 times on topic A rather than upgrade an ambiguous result into a stronger label. That is the precision-over-recall doctrine of §2.4 visible in the output, and it cuts both ways: a system that will not cry contradiction is also a system that will under-report a real one. We would rather show the distribution than argue about it.

Findings, topic C — the worst result in this paper. Topic C reached 2025-02-26, which covers the publication window of 3 of its 4 pre-registered landmarks. It caught 0 of them. DeepSeek-R1, Kimi k1.5 and s1 were all published in January 2025, inside a window the replay ran and screened normally, and none of them reached the review. A system whose entire premise is noticing what moved a field missed the paper that moved this one. We are not going to characterise that as anything other than a failure.

Correcting the first version of this paper. The first release of this document said all three papers were "inside a window the replay ran and screened normally", and that stored state could not tell a retrieval failure from a grading failure. The second half was true and is now fixed (§7.1). The first half was wrong for DeepSeek-R1, and the correction is the most useful result we have. Probing the indexes directly: OpenAlex's only record for DeepSeek-R1 is the *Nature* version, `publication_date` 2025-09-17, and no record exists at all for its arXiv DOI. The replay's window filtered `to_publication_date` to 2025-02-26. The paper was therefore not mis-ranked or under-graded in that window — it was **structurally absent from the corpus**

the window could see, by eight months. arXiv carries it correctly at 2025-01-22, inside the window. Kimi k1.5 and s1 are the opposite case: both are returned by a plain window-filtered OpenAlex search at ranks 4 and 7, so for those two the failure genuinely is downstream of retrieval, in what the system chose to read (§7.2).

Two mechanisms, not one, and neither is the one we assumed. We had reported a single unexplained surveillance miss; it was a corpus date model and a read-budget lottery, and both were fixable.

The identifier decoy, confirmed. OpenAlex carries a record whose arXiv identifier is DeepSeek-R1's but whose title belongs to an unrelated paper. The recall harness now counts a landmark as surfaced only when the *title* agrees, and reports identifier-only matches separately: it found 1 such record in this run. An identifier-only matcher would have scored this benchmark as a success on precisely the landmark we most conspicuously missed.

What went wrong during the runs. One window on topic A failed inside reconcile on a provider length-truncation and is recorded in docs/benchmark.md as an error row rather than as a quiet window; the run continued. Topic A's final 1 window then executed during the retrieval outage and screened nothing, and topic C was stopped mid-window for the same reason. This exposed a defect worth stating plainly, because it is the kind a living system can hide: a window where every source refuses to answer produced the same signature as a genuinely quiet week, and the monitor path would have finalized it as a quiet run and advanced its cutoff — permanently skipping that window's literature for a real user. The measurement found it, and `src/lib/discovery/sourceHealth.ts` now separates a blind round from a quiet one and fails the run instead. The benchmark runs above predate that guard, which is why their blank windows are reported here by hand rather than caught by the system. Neither run's stopping point was chosen by us on the basis of its results — they stopped when the quota did — but a reader has to take that on trust, which is the standing weakness of any vendor-run benchmark.

6. What the failures were, and what we changed

Sections 3–5 are the measurements. This section is what they cost us, because a benchmark a vendor runs on itself is only worth anything if losing changes the product. Every fix below was written after the number that motivated it, and each is a code change with a regression test, not an intention.

6.1 The frontier is dated wrong

Bibliographic indexes date a work by its publication of record. In fields that publish to a preprint server first and a journal months or years later — which is to say, the fields where a standing analyst is most useful — that date is not when the field could read the paper. DeepSeek-R1 is the extreme case we tripped over: indexed at 2025-09-17, public at 2025-01-22, a gap of eight months during which any date-filtered monitor querying that index is blind to it. This is not an OpenAlex defect; it is a category error in treating "publication date" as "the date the field saw this". Our surveillance windows were built on that error, and so, as far as we can tell from their public behaviour, are the alerting features of every competitor that filters by date.

Cesure now carries a **frontier date** on every retrieved record: the earliest date any source can evidence — arXiv submission, Semantic Scholar publication date, or the index date as a last resort — and when two records for the same work merge, the *earliest* wins rather than the more authoritative source. A work is placed in the window in which the field could actually read it.

Two adjacent defects surfaced while fixing it, both silent, both in production for the whole benchmark campaign. The arXiv arm phrase-quoted the entire query, so it performed an exact-phrase match on a natural-language question and returned zero results every time it was called: an arm that had never once contributed, failing in the one way a retrieval arm can fail invisibly, since an empty result set is indistinguishable from a quiet frontier. And the Semantic Scholar arm silently discarded the date window, making it unusable for incremental surveillance and leaking post-window papers into the back-test. Both are fixed and pinned by tests.

A third finding is about query shape rather than dates, and it inverts the intuition that a richer query retrieves better. arXiv and Semantic Scholar match keywords conjunctively, so a natural-language question does not describe a topic to them — it over-constrains it. On topic C's window, same index, same date filter, only the query differing: the full question returns 13 results total and does not contain DeepSeek-R1; the compacted keyword form returns 7,213 and ranks it **first**. The preprint-native arms now distil the brief to at most five content terms.

6.2 Six reads out of a hundred and twenty-three

Kimi k1.5 and s1 were retrieved and still never reached the review, so the loss is downstream. Topic C's third window screened 261 candidates, graded 123 relevant, deep-read **six**, and proposed three changes. Only deep-read papers carry verbatim quotes, and the evidence firewall drops any proposal without one — so the read budget was in effect the proposal budget. Those six were chosen by sorting 123 papers on an integer relevance grade of 0–3, which means the real selection among dozens of ties was arbitrary. A specific landmark's chance of surviving was close to a lottery.

Ranking now uses signals that can actually separate a hundred papers: citation velocity since the frontier date — a landmark distinguishes itself from its cohort within weeks, which is exactly the horizon a monitor operates on — proximity to the review's existing claims, which is what makes a typed diff possible at all, the grader's score, and recency. The fastest-rising papers in a window are read regardless of composite rank. The read budget went from 6 to 20 and the revision cap from 12 to 20.

We then measured it, and it did not work. Re-running the replay produced a real window-3 candidate set — 155 relevant papers, stored — so the ranking could be tested directly on it without spending a further LLM call. Given exactly the papers that window screened, the landmarks still do not reach the read budget: Kimi k1.5 moves from rank 103 to 92 of 155, and s1 moves from 56 to 84 — *worse*. 0 of 2 were inside the old six-paper budget and 0 are inside the new twenty-paper one. Two of the four salience signals were degraded in this measurement (claim contact needs claim embeddings unavailable offline), so it is a lower bound on the ranking's ability — but it is not evidence that the ranking works, and a raised read budget is not a solved funnel.

What the measurement exposed instead. The same query explains the ranks. The relevance grader scored both landmark papers **2**, and scored a content-farm article — an HTML tag still in its title, zero citations, summarising s1 rather than being it — a **3**. Salience weights the grade at 0.35, so every 3 sits above every 2 before any other signal is consulted. No ranking over a mis-scored population can recover the papers the grader put in the wrong bucket. The second failure mode is therefore not the read budget and not the ordering: it is **relevance grading**, and it was invisible until the screening ledger started storing rejected and low-scored candidates. A third defect surfaced in the same data: *s1* is present **twice**, as two paper rows with different ids, different citation counts (11 and 44) and different frontier dates — cross-arm deduplication happens within a retrieval call but not at persist time, which splits the citation signal the ranking depends on. Details and the ordered remediation list are in docs/whitepaper/funnel-findings.md.

6.3 Measured: landmark retrieval recall, before and after

Method. For each of the 13 pre-registered landmarks, we run the shipped retrieval arms restricted to the 90-day replay window containing that landmark's first-publication month, and record whether the arm surfaces it. The query is the topic's research *question*, never the landmark's title — searching for a paper by its title would measure nothing. "Before" arms reproduce the pre-fix behaviour; "after" arms are what ships.

This measures what the system can **see**, not what it then reads or proposes, and it is not a substitute for an end-to-end replay (§9).

Arm	era	landmarks surfaced (of 13)
OpenAlex, window-filtered, full question	before & after	6
arXiv, phrase-quoted question	before	0
Semantic Scholar, full question	before	0
arXiv, compacted terms + frontier window	after	3
Semantic Scholar, compacted terms + window	after	2
Any arm (pipeline recall)	**before → after**	**6 → 8** (46% → 62%)

Findings. Pipeline recall rises from 6 to 8 of 13. The arm that had never contributed anything surfaced 3 landmarks once its query was repaired, **including DeepSeek-R1**, which no OpenAlex-shaped window could have reached at any ranking depth. The repaired Semantic Scholar arm added two more. We report the unflattering half too: five landmarks are still missed by every arm, three of them on topic B — the topic that also never completed a replay — and one, *On the Biology of a Large Language Model*, has no record in any of the three corpora and cannot be caught by improving retrieval at all. Recall of 62% is better, not good.

7. Benchmark 5 — what maintenance does to a review

This is the result we would most like not to have found, and the one we think matters most. A living review's entire promise is that it improves as it is maintained. We measured whether that is true.

The first version of this paper reported a single pooled faithfulness figure: 49% of 47 sampled claims judged grounded in their cited evidence. Pooling concealed the structure of the data. Broken out per review and ordered by how many times the maintenance loop had revised the document, across the 4 audited reviews:

Version audited	claims judged	supported	partial	unsupported	grounded
v1 — compiled, never reconciled	10	10	0	0	100%
v2	5	5	0	0	100%
v4	7	4	3	0	57%
v6 — after eight surveillance windows	25	4	12	9	16%

Grounding falls monotonically with version, at a fitted slope of -18% of grounding per version by ordinary least squares: 100% for never-reconciled reviews against 58% for maintained ones, bottoming at 16% at v6. **Compilation produces grounded reviews and the maintenance loop degrades them.** The published 49% was the average of a review that had never been maintained and one that had been maintained six times, which is not a property of the system so much as a property of our sample.

The mechanism, from the code rather than the correlation. When a tracked change edits a claim, the accept path writes the revised text and *merges* the new evidence onto the claim's existing evidence — and nothing ever re-checks the resulting claim against the evidence it now carries. Claims therefore accrete scope across versions while their support stays fixed and thin; 53% of accepted revisions carried a verbatim quote at all. Faithfulness was audited after the fact, never enforced at the moment of writing.

The fix: a grounding gate. Faithfulness is now a pre-persist firewall rather than a post-hoc score, mirroring the verbatim-quote firewall that already works. Before an edited or added claim is written, the same judge used by the audit scores the *post-edit* claim text against its *post-merge* quotes. Unsupported is refused outright; partial and quote-less evidence are written but flagged; a judge outage fails open, because an LLM being down must not freeze a user's document. The gate optimises exactly the metric this section reports, because it calls the same judge with the same prompt.

What we have not yet shown. That the gate raises grounding on a re-audited maintained review. That measurement requires re-running the replays on the fixed pipeline, which the retrieval quota did not permit in this cycle (§9). The mechanism is identified in code and the gate is tested, but the confirming number is not in this paper, and we are not going to imply that it is. Note also that version is confounded with topic here: the four audited reviews are four different questions, 4 is small, and the trend is suggestive rather than causal on its own. What lifts it above correlation is that we can point at the lines of code that produce it.

8. Benchmark 4 — faithfulness: does the document deserve trust?

Three independent measurements. (i) **Mechanical verification rate**: 81% of evidence quotes stored with verification payloads since 2026-07-11 verify exactly against source text (n=180); the firewalls admit only verified quotes, so what the review shows is what verified. (ii) **LLM-judge audit**: the production faithfulness audit samples claims per review and judges whether each is grounded in its quoted evidence; across the reviews audited for this paper (47 claims judged), 49% were judged grounded (23 supported, 15 partial, 9 unsupported), and 100% of sampled evidence carried source-verification data. **This pooled figure should be read only alongside Section 7**, which breaks it out by version and shows it is an average across reviews at very different stages of maintenance rather than a property of the system. (iii) **Doctrine**: the reconciler is instructed and firewalled to prefer zero proposals over fabricated ones, and quiet surveillance windows complete as first-class quiet runs rather than failures. We state plainly what these measurements are and are not in Section 11.

6.1 What the judge actually caught, and what we did about it

The aggregate hides the interesting part. Across 47 judged claims, 23 were supported, 9 unsupported, and 15 *partial* — and partial is the diagnostic category. A partial claim is not a fabrication. Its quote is real, mechanically verified, and drawn from the paper it names; the claim simply asserts more than the quote proves. The dominant shape was the compound claim: two assertions joined into one sentence, with evidence for one of them. "Mechanically verified" and "faithful" are therefore not the same property, and a system that reports only the first is reporting the easier one. We think this distinction is the single most useful thing in this paper for anyone building in the category.

The fix, and the measurement of the fix. The compile prompt asked for "a standalone, checkable statement" but never forbade conjunctions. We added one rule — each claim asserts exactly one thing; a finding supporting two assertions becomes two claims — and then, rather than assert that this helped, measured it. `scripts/compile-prompt-ab.ts` compiles the same discoveries twice, once with the rule and once with the prompt that produced the benchmark reviews, writes each to a throwaway review, audits both with the production judge, and deletes them. Everything but that one sentence is held constant.

arm	claims compiled	claims judged	partial	grounded
control (the benchmark prompt)	30	20	8	50%
treatment (atomicity rule)	34	18	3	61%

Read this result conservatively. Groundedness rose from 50% to 61%, and the mechanism behaves as designed: partial claims fell from 8 to 3 while the number of claims compiled rose from 30 to 34, which is what splitting compounds should look like. But 20 and 18 judged claims cannot establish an effect of this size — the difference is well inside what chance produces at that sample size, and on one of the two discoveries the treatment arm scored slightly *worse*. The honest summary is that the intervention targets a real, diagnosed failure mode and moves the number in the expected direction, and that we have not yet earned the right to call it an improvement. It ships because the reasoning is sound and the cost is one sentence, not because this table proves it.

The benchmark reviews were not touched. The atomicity rule was written after the replays in §5 were frozen, and no benchmark review was recompiled under it. Every faithfulness number reported above describes documents built by the *old* prompt. The A/B was run on separate throwaway reviews that were deleted afterwards, so they do not appear in the audit aggregate either.

9. How this paper is built: reproducibility as a code path

Every measured number in this document is a `{{TOKEN}}` in a template. Each token resolves through `docs/whitepaper/manifest.json` to a JSON pointer inside a committed run artifact, and `scripts/build-whitepaper.ts` exits non-zero if any token fails to resolve. There is no code path by which a hand-typed number reaches this page. The figures are generated from the same artifacts at build time, so a figure cannot disagree with the table above it.

```
template   ...deep discovery returns 3.17× the baseline's relevant papers...
manifest   "CX1_RATIO": ["contrast", "topics[0].relevantRatio"]
artifact   docs/whitepaper/artifacts/contrast-summary.json → topics[0].relevantRatio
build      resolve → format → substitute, or exit 1 with UNRESOLVED TOKENS
```

We adopted this because the honest failure mode of a vendor whitepaper is not fabrication, it is drift: a number is measured once, the system changes, the number stays on the page. Here a stale number cannot survive a rebuild, and a number without a committed artifact cannot appear at all. Prose claims about other systems are held to a different but explicit standard — each carries a reference to `references.bib`, every entry of which was fetched from the vendor's or publisher's own page on 2026-07-24, with anything we could not verify marked as unverified in `docs/whitepaper/research-notes.md` and excluded from this paper. We are not aware of another system evaluation in this category that ships its numbers this way, and we would rather the practice spread than remain a differentiator.

10. Comparison to alternatives (feature-level, checked 2026-07-24)

Feature claims below were fetched from each vendor's official pages on 2026-07-24; notes and URLs are in docs/whitepaper/research-notes.md. We deliberately compare *features*, not performance: no vendor below publishes an artifact-reproducible benchmark we could re-run, and we do not fabricate one.

	living versioned review	typed tracked changes	mechanical quote verification	monitoring cadence	published accuracy eval
Cesure	**yes**	**yes (4 types + qualify/needs_review)**	**yes (81% measured)**	**weekly/daily**	**this paper**
Undermind[1]	no (search reports)	no	no (citation tracing)	enterprise alerts	**whitepaper 2024 (vendor-run)**
Elicit	"living reviews" = manual re-run	no	no (quotes for spot-check)	**alerts (10 concurrent)**	**screening/extraction evals (vendor-run)**

Consensus	no	no	no	no	none found
NotebookLM	no	no	no	no	side-by-side win rates (vendor-run)
SciSpace	no	no	no	no	self-run benchmark vs Elicit/Consensus
scite	no (citation index)	supporting/contrasting citation types	no	**citation alerts**	none found
Ai2 ScholarQA[12]	no	no	verbatim quotes + excerpts (no mechanical check)	no	**ScholarQABench (vendor-run)**

Two systems deserve singling out. Undermind's whitepaper[1] set the methodological bar this paper follows (fair-comparison baseline, conditional error tables, an explicit comprehensiveness argument); Cesure differs in maintaining a versioned document rather than a search report, and in substituting a deterministic quote check for a probabilistic one. Ai2's ScholarQA[12] is the closest on evidence display (verbatim quotes with clickable excerpts) but remains an on-demand answer engine with no maintenance, versioning, or mechanical verification of those quotes.

11. Limitations

- **Vendor-run benchmarks.** Every number in this paper is measured by us on our own harnesses — the same caveat that applies to Undermind's, Elicit's, SciSpace's and Ai2's self-evaluations. Our mitigation is narrower and stronger than retraction-proof prose: runnable scripts plus committed artifacts, and a build that fails on unsourced numbers. Independent replication has not happened for us or for anyone in this category.
- **LLM judge ≠ human raters.** The relevance grader and the faithfulness judge are both LLMs, and neither is validated against human raters in this revision. Undermind validated its classifier against 432 human-checked papers; we have no equivalent yet, and a blind single-rater study is the next thing we intend to run. Where a judge could be biased in our favour it grades both arms of Benchmark 1 under the identical rubric, blind to which arm a paper came from — but that is an argument, not a measurement.
- **Scale of evidence.** Three benchmark briefs, 2 completed replay topics of 3 pre-registered, both stopped before their final windows, 47 audited claims, n=180 quotes for the verify rate. These are real but small; we report them, not extrapolations.
- **Corpus and access.** Discovery covers OpenAlex/S2/arXiv; deep reads require open-access PDFs (~37% of OpenAlex works are OA). Paywalled papers contribute metadata-level evidence only.
- **Replay auto-acceptance.** Back-tests auto-accept surviving proposals; they measure the pipeline, not the human-in-the-loop precision of the shipped workflow. Operating bounds cap what one window can surface: the replays reported here ran with 6 ensure-reads, 25 candidates and 12 revisions per run; the shipped bounds are now 20/40/20 (§6.2), so these numbers characterise a pipeline that no longer exists in that configuration.
- **What is measured, and what is not.** Retrieval recall is re-measured directly (6 → 8 of 13). The salience ranking is measured too, and *failed* (0 → 0 landmarks inside the read budget, §6.2) — we report the raised read budget as a coverage change and explicitly not as a solved funnel. The **grounding gate is still unmeasured end-to-end**: it is shipped, unit-tested, and enforcing during the re-run, but no re-audited maintained review exists yet to show grounding stops falling with version. That remains the single most valuable measurement we owe this paper.
- **The re-run replay did not complete.** Topic C was re-run on the fixed pipeline and stopped after 2 of 9 windows: deep reads at the wider budget saturate the LLM provider's tokens-per-minute ceiling, and roughly 92 minutes of the run's wall-clock was spent in backoff. Stopping was a cost decision taken against a rule written down before any landmark outcome was known (docs/whitepaper/replay-c-rerun-note.md), and the run did not reach the windows containing any pre-registered landmark — so **this paper makes no end-to-end landmark claim from it**. What it did produce is the stored candidate set §6.2 analyses, and the observation that windows 1 and 2 achieved 20 deep reads each against the old pipeline's 6.
- **Look-ahead bias in the read-selection ranking.** The salience score that decides which papers a replayed window deep-reads includes citation velocity, computed from present-day citation counts rather than counts as of the window being replayed. A monitor running in February 2025 could not have known that DeepSeek-R1 would accumulate hundreds of citations. Retrieval recall (§6.3) is unaffected — nothing there is ranked by citations — but any end-to-end landmark catch from a replay is an *upper bound* on what a live monitor would have achieved, not a like-for-like reconstruction of one. Velocity is one of four signals, so the effect is partial rather than decisive. The fix is a harness change, not a product one: OpenAlex exposes counts_by_year, so an as-of-window count can be reconstructed for replay mode while production keeps live counts. It is not in this revision, and it was recorded before the run it affects had produced a result.
- **Retrieval recall is not pipeline recall.** §6.3 measures whether an arm *surfaces* a landmark in its window. Surfacing it is necessary, not sufficient: Kimi k1.5 and s1 were surfaced at ranks 4 and 7 in the original replay and still never reached the review. Recall of 62% bounds what the system can do, not what it did.

- **Incomplete replay coverage.** One pre-registered topic never ran and the two that did lost their later windows, because a free-tier retrieval quota ran out mid-session (§5). The landmark denominator is therefore the testable subset under a post-hoc exclusion rule we flag as post-hoc. A fuller replay is the single most valuable thing we could add to this paper, and it is a matter of budget, not method.
- **Rejected candidates were not persisted (fixed, but not retroactively).** During every run reported here the grader stored only papers it scored 2 or higher, so a paper absent from a run could not be distinguished from a paper the grader saw and rejected — exactly the evidence needed to localise topic C's missed landmarks, and we did not keep it. The screening ledger now records every screened candidate with its score and rationale, plus every candidate the cosine prefilter dropped before grading, so the three fates are distinguishable going forward. It cannot be applied backwards: the misses in §5 remain misses of partly-inferred mechanism, localised in §5 by probing the indexes directly rather than by reading our own stored state.
- **Grader variance.** LLM grading is stochastic; artifacts pin the runs we report, and re-runs may drift a few points.
- **The coverage estimator's assumption.** Capture-recapture requires the two capture occasions to be independent. Search-type and citation-type arms are not fully independent, and on topic 3 the violation is severe enough to produce a meaningless 99% (§4). We report the estimate per round rather than as a headline so the failure is visible; we do not claim exhaustiveness anywhere in this paper.

12. Related work

One-shot survey generation. STORM[14], PaperQA2[15], AutoSurvey[16], LitLLM[17] and SurveyForge[18] generate static documents; none maintains state across time. PaperQA2's contradiction detection (70% human-validated) is the closest academic relative of our typed diffs. **Scholarly RAG with attribution.** OpenScholar[11], ScholarQA[12] and the Asta ecosystem[13] anchor answers in retrieved passages; ScholarQABench and AstaBench are the evaluation precedents we follow. **Attribution measurement.** ALCE[6], AIS[20] and RAGAS[19] judge attribution probabilistically; our verification is a deterministic substring check with repairs that only shrink. **Living systematic reviews.** Elliott et al.[3] founded the concept in clinical evidence; Thomas et al.[4] combined human and machine effort; Marshall & Wallace[5] document why full automation stayed out of reach. Cesure is that agenda, cross-domain, with the claim graph as the maintained object. **Deep search.** Undermind[1] demonstrated LLM-guided exploration with a comprehensiveness argument; our discovery arm is comparable in spirit, our maintained artifact is the difference.

Appendix A — the production grading rubric (verbatim)

```
You are a meticulous research librarian. You grade papers for relevance to a research brief. Score 0 = unrelat
```

Appendix B — baseline keyword generation (verbatim prompt + generated queries)

```
System: You are a thoughtful, expert scientist, and you are knowledgeable about carefully crafting a search ph
User: I am trying to help a colleague find papers about this topic: '<restatement>'. In addition, here are the
```

Generated query sets (from the baseline artifact): topic 1 — sparse autoencoders mechanistic interpretability LLM · dictionary learning large language models monosemanticity · gated JumpReLU top-k sparse autoencoders · evaluating SAE feature quality LLM activations · sparse autoencoder circuit discovery editing; topic 2 — SARS-CoV-2 clinical presentation first patients · SARS-CoV-2 spike protein ACE2 receptor structural biology · dexamethasone remdesivir COVID-19 randomized controlled trials · mRNA vaccine efficacy pivotal trials COVID-19 · early COVID-19 clinical features and viral structure; topic 3 — selective state space models Mamba linear time sequence modeling · linear gated attention vs softmax attention LLM · long context window extension positional interpolation ring attention · KV cache compression efficient long context inference · hybrid attention recurrence architectures large language models.

Appendix C — replay methodology

Harness: scripts/replay-review.ts (seed capped at dateTo=T; windows run the same monitor-search + reconcile + firewall + applyRevisions path as production; budgets bound each window). Pre-registration: docs/whitepaper/replay-preregistration.md, committed 2026-07-24 before any replay output was inspected, lists the landmark papers per topic and the scoring rule (caught = cited by the review in any version after its publication window; misses reported, not explained away). Raw window metrics: docs/benchmark.md; per-topic aggregates: docs/whitepaper/artifacts/replay-summary.json (committed).

Re-running a replay on a changed pipeline. Two hazards were found and closed while preparing this revision, both of which would have produced a confidently wrong number. First, the harness kept its own copy of the read path, so a re-run would have exercised the new retrieval arms against the *old* six-paper read budget and reported it as the fixed pipeline; the harness now shares the salience ranking and read budget with production, and the two constants carry a comment saying they must move together. Second, scripts/summarize-eval.ts selected the highest-*version* replayed review per topic, so a fresh run ending at a lower version than a stale row would have silently reported the old pipeline's results under the new run's date; it now selects the most recently created run and names any it supersedes. The stopping rule for the re-run — that it may end once the windows containing every pre-registered landmark have completed — is recorded in docs/whitepaper/replay-c-rerun-note.md, committed while the run was still executing and before any landmark outcome was known.

A note on the raw metrics file. The harness appends one row to docs/benchmark.md as each window finishes. The benchmark topics were replayed concurrently, so rows from different topics interleaved and some landed under the wrong heading. scripts/rebuild-benchmark.ts repairs placement only: each row is matched back to the run that produced it via that run's own log (window range, papers seen, papers relevant) and re-emitted in run order, and any row no log claims is preserved in an explicit UNATTRIBUTED section rather than deleted. No number in any row is recomputed. The aggregates the paper reports are read from the database by scripts/summarize-eval.ts and never from that file.

Appendix D — artifact manifest and reproduction

Every {{TOKEN}} placeholder in this document resolves through docs/whitepaper/manifest.json to a JSON pointer inside one of these committed artifacts (docs/whitepaper/artifacts/ — committed to the repository; the harnesses' live output lands in .eval/ and is synced here as part of the build ritual); scripts/build-whitepaper.ts fails the build otherwise. Figures 1–4 are generated from the same artifacts at build time.

- docs/whitepaper/artifacts/eval-2026-07-24T06-12-13-333Z.json — deep-discovery eval (scripts/eval-discovery.ts)
- docs/whitepaper/artifacts/eval-baseline-2026-07-24T16-04-10-744Z.json — keyword baseline (scripts/eval-baseline.ts)
- docs/whitepaper/artifacts/replay-summary.json — replay aggregates (scripts/summarize-eval.ts; rows in docs/benchmark.md)
- docs/whitepaper/artifacts/faithfulness-summary.json — audit aggregates (scripts/faithfulness-audit.ts, summarized by scripts/summarize-eval.ts)
- docs/whitepaper/artifacts/verify-rate-summary.json — mechanical verify rate (scripts/measure-verify-rate.ts)
- docs/whitepaper/artifacts/contrast-summary.json — discovery-vs-baseline join (scripts/summarize-eval.ts)
- docs/whitepaper/artifacts/compile-ab-summary.json — compile-prompt A/B (scripts/compile-prompt-ab.ts)

```
pnpm exec tsx scripts/eval-discovery.ts && pnpm exec tsx scripts/eval-baseline.ts
pnpm exec tsx scripts/replay-review.ts --asof 2024-06-01 --step-days 90 --user <uuid> "<topic>"
pnpm exec tsx scripts/faithfulness-audit.ts <reviewId> --n=40
pnpm exec tsx scripts/measure-verify-rate.ts && pnpm exec tsx scripts/summarize-eval.ts
pnpm exec tsx scripts/human-eval-sheet.ts --n=60   # then grade .eval/human-grading-sheet.md
pnpm exec tsx scripts/human-eval-score.ts
pnpm exec tsx scripts/rebuild-benchmark.ts         # row placement only; run with no replay in flight
pnpm exec tsx scripts/build-whitepaper.ts
```

References

1. Hartke, T., Ramette, J. Benchmarking the Undermind Search Assistant. Undermind.ai whitepaper, Jan 5 2024. undermind.ai/whitepaper.pdf (vendor-run).
2. Shojania, K.G., et al. How Quickly Do Systematic Reviews Go Out of Date? A Survival Analysis. Ann Intern Med 147(4):224–233, 2007. PMID 17638714.
3. Elliott, J.H., et al. Living Systematic Reviews: An Emerging Opportunity to Narrow the Evidence-Practice Gap. PLoS Med 11(2):e1001603, 2014.
4. Thomas, J., et al. Living systematic reviews: 2. Combining human and machine effort. J Clin Epidemiol 91:31–37, 2017.
5. Marshall, I.J., Wallace, B.C. Toward systematic review automation. Syst Rev 8:163, 2019.
6. Gao, T., Yen, H., Yu, J., Chen, D. Enabling Large Language Models to Generate Text with Citations (ALCE). EMNLP 2023.
7. Walters, W.H., Wilder, E.I. Fabrication and errors in the bibliographic citations generated by ChatGPT. Sci Rep 13:14045, 2023.
8. Bhattacharyya, M., et al. High Rates of Fabricated and Inaccurate References in ChatGPT-Generated Medical Content. Cureus 15(5):e39238, 2023.
9. Chelli, M., et al. Hallucination Rates and Reference Accuracy of ChatGPT and Bard for Systematic Reviews. J Med Internet Res 26, 2024. PMID 38776130.
10. Gravel, J., D'Amours-Gravel, M., Osmanlliu, E. Learning to Fake It: Fabricated References Provided by ChatGPT for Medical Questions. medRxiv 2023.03.16.23286914.
11. Asai, A., et al. OpenScholar: Synthesizing Scientific Literature with Retrieval-augmented LMs. arXiv:2411.14199; Nature 2026, doi:10.1038/s41586-025-10072-4.
12. Singh, A., et al. Ai2 Scholar QA: Organized Literature Synthesis with Attribution. arXiv:2504.10861, 2025.
13. Bragg, J., et al. AstaBench: Rigorous Benchmarking of AI Agents with a Holistic Scientific Research Suite. arXiv:2510.21652, 2025.
14. Shao, Y., et al. Assisting in Writing Wikipedia-like Articles From Scratch with Large Language Models (STORM). NAACL 2024.
15. Skarlinski, M.D., et al. Language agents achieve superhuman synthesis of scientific knowledge (PaperQA2). arXiv:2409.13740, 2024.
16. Wang, Y., et al. AutoSurvey: Large Language Models Can Automatically Write Surveys. arXiv:2406.10252, 2024.
17. Agarwal, S., et al. LitLLM: A Toolkit for Scientific Literature Review. arXiv:2402.01788, 2024.
18. Yan, X., et al. SurveyForge: On the Outline Heuristics, Memory-Driven Generation, and Multi-dimensional Evaluation for Automated Survey Writing. arXiv:2503.04629, 2025.
19. Es, S., et al. Ragas: Automated Evaluation of Retrieval Augmented Generation. arXiv:2309.15217, 2023.
20. Rashkin, H., et al. Measuring Attribution in Natural Language Generation Models (AIS). arXiv:2112.12870, 2021.
21. Priem, J., Piwowar, H., Orr, R. OpenAlex: A fully-open index of scholarly works, authors, venues, institutions, and concepts. arXiv:2205.01833, 2022.
22. Chapman, D.G. Some properties of the hypergeometric distribution with applications to zoological sample censuses. University of California Publications in Statistics 1(7):131–160, 1951.