Recorded Derivation: Accumulating Competence as Checkable Operations, and an Instance of Evaluation Blindness

Prometheus7 Research Institute

September 2026 — draft 3

Abstract

We study a question-answering system, Oracle7, whose acquired competence is stored as explicit executable operations rather than as learned parameters. Each answer is produced by typed operations over addressed, citable evidence, and the system abstains when no operation completes. Its runtime contains no learned weights. Its extensions have been proposed largely with the help of learned coding models and are admitted by an automated gate that refuses measured regressions. We make three contributions.

First, a stage-located diagnosis. On 600 open-domain questions from untouched SimpleQA, TriviaQA and HotpotQA windows, 69% were refused before retrieval. For 170 the answer string appeared on a page linked from the question, yet the served system refused 145 of them. This is an opportunity for improvement, not a measured reserve of usable competence.

Second, a documented case of evaluation blindness. An extension that changes behaviour only outside the evaluation set is indistinguishable, under the gate's instruments, from the system without it. We observe this directly and describe repairing the instrument. The repaired instrument then exposed six genuine errors in the extension, and a validation of older labels found 43% of the gate's store-premise "must-decline" labels contradicted by the current store.

Third, a reader built as the first operations of a proposed common interface for questions, evidence and operations. It answers a small fraction of refused questions at high selective precision, with small samples and wide intervals.

We state the programme these results serve as a hypothesis with a pre-registered test: that failures can be converted autonomously into reusable operations which transfer to independently selected problems at a controlled error rate, and whose accumulation lowers the cost of further acquisition. We report the first acquisition under that protocol. From the structured abstentions of a training window, a comparison operation was written, validated metamorphically on store-generated pairs, and tested once on 234 sealed comparison questions: coverage 4.7%, selective accuracy 72.7% (95% CI 39–94%) by answer identity. The benchmark's substring scorer would report 10 of 11, because a comparison reply cites both candidates. A second acquisition, a bridge operation composed from the existing ones, reused them in every derivation but answered one of 966 sealed questions, wrongly. Composition over low-recall primitives compounds losses, not capability.

1 Introduction

Most current AI systems store what they learn in numerical parameters. A different option is to store acquired competence as explicit procedures: operations with conditions of applicability, evidence requirements, and a check on their result, which can be inspected, composed, repaired, and transferred between systems. The attraction is cumulative, auditable growth, in the way a

mathematical library or a software ecosystem grows. The difficulty is equally familiar. Manually written procedures reproduce the knowledge-engineering bottleneck that limited earlier symbolic systems [7], and checks written in natural language rarely certify the proposition they appear to check.

Oracle7 is a working instance of the first option. Its served runtime is a chain of 36 lettered base runtimes and a few hundred admitted extensions ("folds"), each a small program subclassing the current head. A fold is admitted only if an automated gate measures a gain, and no loss, on held-out conversation scripts and a capability panel. The runtime has no learned weights. The acquisition process is hybrid: most capability folds, including those reported here, were proposed with the assistance of learned coding models and retained outside those models' parameters. Separating the two ambitions matters. Showing that durable competence can live in explicit, checkable procedures is a different claim from showing that a capable system can operate and improve with no learned component anywhere. We pursue the first.

This paper reports a diagnosis (Section 2), an instance of evaluation blindness and its repair, with what the repair exposed (Section 3), a reader implementing the first operations of a proposed common interface (Section 4), results with their limits (Section 5), and the programme's pre-registered test (Section 6).

2 Where the answers were lost

We took a frozen release head and 600 factoid questions from windows no earlier run had scored: 200 each from SimpleQA [13], TriviaQA [5] and HotpotQA [14]. Gold answers graded each stage and were never used to search.

Measure (600 fresh questions)	Count
Refused before any retrieval	415 (69%)
Answer string on a page linked from the question by the store's title index	170
of which the served system refused	145
Answered correctly by the served system	15

Table 1: Stage location of losses. "Answer string on a linked page" is not "answer supported by that page".

The store holds 6.34 million articles and 20.5 million cited sentences, including a slot-limited ingestion of the full English Wikipedia (at most eight sentences per article). Three mechanisms account for most of the loss:

- **A routing layer keyed on surface forms.** Routes fired on recognised wording; unrecognised questions fell to a clarification terminal before any evidence was consulted.
- **Subject identification by capital-letter runs.** This fragments and merges names. Linking word n-grams against the store's own title index found a subject page for 561 of 600 questions.
- **Selection by lexical overlap.** Where material was reached, overlap selection could not distinguish an answer from an adjacent true sentence.

A compiled representation of each question's obligations existed and was unused at selection.

These counts measure an opportunity. A page may mention the right string in an unrelated context, among several candidates, or only across sentences. The claim is narrow: executable access, not stored text, bounded the served system on this subset.

3 Evaluation blindness, its repair, and what the repair exposed

Formal statement. Let T be an evaluation set and $M_T(f)$ the vector of instrument readings for system f. If an extension g of f changes behaviour only on inputs outside T, then $M_T(g) = M_T(f)$. A gate that requires a strict improvement in some component of M_T cannot admit g on the strength of its new capability, whatever that capability's value on the deployment distribution. This is an observability defect: the measurement cannot distinguish materially different systems. It does not follow that no unmeasured improvement can ever occur, since visible gains may transfer incidentally. It does follow that the gate cannot directly reward such improvements.

Observation. The gate's instruments were a corpus of multi-turn scripts (454 person turns) and a capability panel (1,158 held-out items). Both were drawn from phrasings the routes already handled. The first version of the reader in Section 4 changed zero turns and zero panel items when replayed through them.

Repair. Panel v17 adds an `open_lookup` family, built from the dataset key and the store only, from windows no run had scored:

- 150 items whose answer string appears verbatim on a page linked from the question;
- 100 must-decline items whose answer co-occurs with none of the question's terms anywhere in the store.

Both constructions are heuristics that favour the store's form of evidence, not ground truth. The head's reading rose from 623 to 724 on installation. That is a change in the instrument, not in capability.

What the repaired instrument exposed. Measured on v17, the reader's candidate raised capability by one item and was *rejected*, because assertions on must-decline items rose from 22 to 24. Replaying all 638 must-decline items isolated ten assertions:

Kind	Case
Role reversal	"Who beat Tim Henman…?" → the player Henman beat
Lexical category	the verb "composed" matched the noun "composers"
Missed bridge (2)	a described subject ("the American lawyer … who"); "the hill located in what town"
Unbound identifier (2)	a part code ("gal56") was not treated as an obligation (one question, in two families)
Weak type check	a pronoun on a page accepted as evidence of a person
Source conflict	the store dates a founding 23 August 1954; the key says 22 December 1954
Ambiguous key	signatory versus negotiators of a treaty
Stale label	the store states the key (150 engravings); the label "absent from every store" is out of date

Table 2: Assertions of the rejected candidate on must-decline items, adjudicated against the dataset keys.

These are the failure modes a critic of natural-language checking would predict. The checks established type membership and the presence of entities, not the asserted relation between them.

Label validation. Because one disputed label was demonstrably out of date, we re-tested *every* store-premise must-decline label (301), not only those the candidate touched. We used the dataset key and the current store, with a strict attestation test: the key on a linked page, or a non-numeric key of at least five characters in one sentence with two distinct question terms. 130 (43%) were contradicted. They were excluded, not relabelled, since a string's presence does not establish support. The instrument's must-decline labels had silently decayed as the store grew. An evaluation built against a moving knowledge source needs its labels revalidated whenever the source changes.

4 The reader

We propose a common interface for questions, evidence and operations: a small set of roles a question can leave open (subject, relation, constraints, answer type), the store categories that can fill them, and the operations that move between them. Each operation declares what it requires, what it produces, and how its output is checked. The value sought is the one an intermediate representation provides in a compiler: many inputs and transformations cooperating through shared conventions. We earlier drew this interface as three Tree-of-Life graphs sharing a middle pillar. That depiction has no demonstrated computational significance. The diameter of a graph of operation *types* bounds nothing about the configuration space of concrete derivations [11], and whether this topology matters is a pre-registered ablation (Section 6), not a claim.

The reader implements the first operations, invoked only where the served chain did not answer:

1. **Frame.** Answer type; governing predicate (the verb nearest the answer phrase); constraints (other names, numbers, identifiers and years); the answer's role (agent or patient).
2. **Link.** Subjects via the title index.
3. **Expand.** The subject's own sentences.
4. **Bind.** A value of the required type from a sentence containing the predicate *as a verb*.
5. **Check.** Each check states the property it establishes, which is weaker than the proposition:
    - an entity's own page asserts it is the requested kind;
    - other named entities and the question's years occur in the binding sentence;
    - agent and patient roles are consistent with voice;
    - an exact date is corroborated by a second sentence;
    - a value is unique in its sentence.
6. **Verify.** One candidate survives, or nothing is said.

Comparisons and described subjects are declined. The checks consult evidence the proposal did not use, but they share its parse, entity link and store, so they cannot catch errors those shared stages produce.

Set, reader	Answered	Correct	Coverage	Sel. accuracy (95% CI)
Held-out, v1 (526, fresh, once)	7	5	1.3%	71% (29–96)
Development, v1.3 (600)	7	7	1.2%	100% (59–100)
Panel v18, answerable items (v1.3)	8	8	—	100% (63–100)
Gate on panel v18 (v1.3)	admitted: capability 633 → 637; false yes 22 → 22			

Table 3: Reader results. Development material informed revisions and is weak evidence of generalisation. The held-out errors informed v1.1, so that window no longer tests later versions. Panel v18 was built from windows no reader version saw, but its answerable items were selected with a title-index heuristic similar in kind to the reader's own linking.

5 Results

We report coverage, selective accuracy with exact 95% binomial intervals, and correct yield separately.

Correct yield on the held-out window was 0.95%. Coverage is the limitation. On development, the reader's stops are distributed as frame 236, link 31, bind 230 and check 96. These are first-failure counts, which censor downstream behaviour. They locate where execution stopped, not the causal origin of the loss. Revisions traded coverage for selectivity: development answers fell from twelve to seven while errors fell from one to zero. Whether that is improvement depends on the relative cost of an error and an abstention.

6 The programme, and its first acquisition

The contribution we care about most is a unit of accumulation. A failure could leave behind a durable operation, with explicit applicability conditions, evidence requirements and a check, which later derivations can invoke and compose. If such operations transfer beyond their motivating examples, one acquisition unlocks many derivations. If recurring patterns can be compressed into higher-level operations [2], later search becomes cheaper. Separating proposal from acceptance lets an unreliable proposer, including a learned model, contribute, provided its products face an adequate check. Related systems pair learned proposal with symbolic verification in formal domains [12], and retain evaluator-selected improvements to executable algorithms [10].

The test has six steps:

1. choose a failure class;
2. acquire an operation from structured abstentions alone;
3. validate it on cases generated from the store and pass the gate;
4. measure transfer on sealed data;
5. measure compounding over a second acquisition;
6. run a topology ablation.

We have completed steps 1 to 5; step 6 remains.

1. Class selection, and a deviation from the earlier draft. Draft 2 named attribute questions as the class. We replaced that choice with a rule applied to data: *the largest class of store-supported abstentions that no operation covers*, measured on a training window (TriviaQA and HotpotQA rows 4800–5199, 800 questions, of which the reader answered 17). "Store-supported" means the key appears on a page linked from the question.

Class	Abstentions	Store-supported	Dominant stop
Which-noun	386	128	bind / frame (covered, partially)
Comparison	**87**	**46**	**frame (84): no operation**
Described subject (bridge)	105	21	frame
Attribute ("the X of Y")	51	5	mixed

Table 4: Training-window classes. The attribute class named in draft 2 turned out to have little store support.

2. Acquisition. A proposer, a learned coding model, wrote a comparison module from the 87 comparison records alone. It had seen earlier development windows, but not the training rows beyond these records and not the sealed windows. The module declares six operations, each with its requirements, output and check:

- *order*: first/earlier versus later/younger, using a year bound to the question's relation on each entity's own page;
- *count*: more versus fewer, using a number attached to the asked noun;
- *both-kind*: "Yes" only when both first sentences state the kind with all its modifiers. It never answers "no" from absence;
- *same year*;
- *same occupation*;
- *shared person*.

On its own training class it answers 4 of 87, all correct.

3. Validation. A gold answer generated from the store would reuse the module's own extraction, so we tested metamorphic relations instead, on 140 pairs of real persons and genera sampled from the store (none from a benchmark). The module answered 53. Swapping the pair left the answer unchanged in 53 of 53 cases, reversing the direction reversed it in 53 of 53, and the answer was always one of the two names.

A precheck against every must-decline panel item found one genuine defect: a kind's modifiers were dropped, so "both American film directors" was satisfied by a French director. It also found a flaw in our own v17 construction rule, which cannot apply to yes/no questions; the two affected items were removed. After the fix, the module asserts on none of 506 must-decline items.

To make its gain visible, panel v19 adds 65 HotpotQA comparison-type items (rows 5200–5599). They were selected by the dataset's own labels, not by the module. Gate verdict: admitted (capability 644 → 646; false yes 22 → 22; no other instrument moved).

4. Sealed transfer, run once. The test set was the HotpotQA comparison-type questions in rows 5600–6799 (234 items, selected by dataset label, never inspected), with the module frozen before the run.

Answered	11 (coverage 4.7%)
Correct by answer identity	8 (selective accuracy 72.7%, 95% CI 39–94%)
Correct yield	3.4%
Benchmark substring scorer (for comparison)	10 of 11

The benchmark's scorer checks whether the key appears anywhere in the reply. A comparison reply that cites both entities' pages therefore contains the key even when it chose wrongly. We score the answer itself and changed the reply to name only the chosen entity.

Of the three identity errors, one is a genuine error: two restaurant chains in the wrong order. The other two contradict the dataset key on birth years that the store states (1971 against 1977, and 1946 against 1973). We report them as key disputes but count them as errors.

5. Compounding: a second acquisition, with a negative result. With the comparison operation admitted, the same rule, re-applied to the same training window, selected described-subject bridges ("In what year was the singer whose second remix album was X born?"): 104 abstentions, 20 store-supported, declined by design. This time the proposer was told which operations exist. It wrote a bridge operation that is almost pure composition:

- hop 1 finds the described entity using the reader's own link, bind and check, plus a link-back test;
- hop 2 answers the question with the entity substituted, using the reader or the comparison operation.

	Acquisition 1 (compare)	Acquisition 2 (bridge)
New code (lines)	248	211
Revision rounds before freeze	4	7
Training-class answers	4 / 87 (all correct)	1 / 104 (correct)
Must-decline assertions (506 items)	0	0
Sealed: answered / correct	11 / 8 of 234	1 / 0 of 966
Derivations reusing earlier operations	—	100% (by trace)

Table 5: The two acquisitions. Reuse is read from each answer's recorded trace, not asserted.

Its one training answer (a show found through a character, then its creator) and its one answer in an adjacent labelled window (a football coach, then his birth year) were correct. On 966 sealed bridge questions it answered one, wrongly. Its description parser read "the population of the city where…" as a description of *the city*, and so answered the city instead of its population. We record the defect and do not re-test on the sealed set.

Interpretation. Reuse occurred, and the code needed was smaller. Capability did not compound. The reason is arithmetic: a composed operation's recall is roughly the product of its hops' recalls, and each hop inherits the reader's recall of about one percent on open questions. Composition over low-recall primitives compounds losses, not capability. The growth mechanism therefore requires per-hop recall to rise first. The largest single loss in the stop traces is binding: pages that state a relation with a different verb from the question's. That is the next target, not further composition.

A limit of the gate. Measured on the adjacent labelled window, the bridge operation's gain would be one item. The gate treats a change of one item as within its resolution, so no instrument of the current size can admit it. A capability smaller than an instrument's resolution is invisible to it. This is the evaluation-blindness result of Section 3 in a quantitative form, and it argues for instrument families sized to the gains being sought.

6. Topology ablation. Not yet run.

7 Evaluation protocol

Three tiers are kept distinct:

- **Development feedback**, where inspection and revision are allowed.
- **Admission instruments**, the gate's panel. These are adaptive by construction, because hundreds of admission decisions query them [1].
- **Sealed assessment sets**, touched once per release.

Instrument changes are reported as such. Labels built against a knowledge source are revalidated whenever the source changes.

8 Related work and positioning

Execution provenance, recording how outputs depend on inputs, has formal treatment in provenance semirings [4]. Separating a complex producer from a small trusted checker is the principle of proof-carrying code [9]. Retrieval-augmented generation combines learned models with external memory [8]. Selective prediction formalises the trade-off between coverage and risk [3]. Systematic generalisation motivates tests that withhold combinations of known primitives [6]. Our contribution is not any of these components. It is an operating system of them (addressed evidence, gated additive operations, structured abstentions), a documented failure mode of its self-improvement, and a pre-registered test of whether its growth mechanism can overcome the linguistic brittleness its measurements expose.

9 Limitations

- Coverage is about 1% and the samples are small. The intervals are wide.
- The checks are heuristic extraction safeguards, not soundness guarantees. A formal account would specify, for each accepted record, the inference rules under which it licenses the answer, conditional on the evidence's truth and the fidelity of the translation.
- Admitted extensions preserve issued answers only in a fixed environment. They also preserve existing errors, and the uniqueness rule is non-monotone in evidence. A correction mechanism is needed alongside additive extension.
- The acquisition process has so far been human-directed with learned-model assistance. Claims of autonomous improvement await the test in Section 6.

10 Conclusion

A system that stores competence as operations was answering a small fraction of what it could reach, and its self-improvement could not see why. Widening the instrument exposed a new capability and new errors, and revalidating old labels showed that the instrument itself had decayed.

Under a stated protocol, one failure class was then converted into an operation:

- the class was chosen by data;
- the operation was written from structured abstentions alone;
- it was validated metamorphically on pairs generated from the store;
- it was admitted by the gate;
- it was tested once on sealed data, where it reached 72.7% selective accuracy at 4.7% coverage, with a wide interval.

A second acquisition, a bridge operation built almost entirely from the existing ones, confirmed that reuse happens and that it can reduce the new code required. Capability did not compound: on 966 sealed questions it answered one, wrongly, because composition multiplies the low recall of each hop. The mechanism that would make this approach consequential is failures converted into operations that transfer, compose, and make the next acquisition cheaper. It has not been demonstrated. The measurements say what must come first: raising per-hop recall, chiefly the binding of relations that pages state with different verbs, and sizing the gate's instruments to the gains being sought.

References

[1] C. Dwork, V. Feldman, M. Hardt, T. Pitassi, O. Reingold, and A. Roth. The reusable holdout: Preserving validity in adaptive data analysis. *Science*, 349(6248):636–638, 2015.

[2] K. Ellis et al. DreamCoder: Bootstrapping inductive program synthesis with wake-sleep library learning. In *PLDI*, 2021.

[3] Y. Geifman and R. El-Yaniv. SelectiveNet: A deep neural network with an integrated reject option. In *ICML*, 2019.

[4] T. J. Green, G. Karvounarakis, and V. Tannen. Provenance semirings. In *PODS*, 2007.

[5] M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*, 2017.

[6] B. Lake and M. Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *ICML*, 2018.

[7] D. B. Lenat. CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38, 1995.

[8] P. Lewis et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *NeurIPS*, 2020.

[9] G. C. Necula. Proof-carrying code. In *POPL*, 1997.

[10] A. Novikov et al. AlphaEvolve: A coding agent for scientific and algorithmic discovery. arXiv:2506.13131, 2025.

[11] T. Rokicki, H. Kociemba, M. Davidson, and J. Dethridge. The diameter of the Rubik's Cube group is twenty. *SIAM Review*, 56(4):645–670, 2014.

[12] T. H. Trinh, Y. Wu, Q. V. Le, H. He, and T. Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625:476–482, 2024.

[13] J. Wei et al. Measuring short-form factuality in large language models. arXiv:2411.04368, 2024.

[14] Z. Yang et al. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*, 2018.