EigenEngine

$\hat{E}_i|\text{gen}\rangle \otimes \hat{E}_n|\text{gen}\rangle$

Persistence and generation as problems of representation

EigenEngine, with Gerard McCaul

Eigen@gmccaul.co.uk

The same structure as the companion paper, projected into a second form.

Two gaps

A language model returning to a growing body of work fails twice.

- **Persistence.** No state survives a session. The history is re-supplied each time, the window is bounded, and recall sags in the middle of a long input.
- **Generation.** With no record of its own past, output drifts toward the average of its training rather than toward this work.

Prompting, fine-tuning, reinforcement all steer a fixed model. None adds the missing piece: a structured store that persists and grows.

What EigenEngine is

EigenEngine is two things at once, one mechanism.

- A **persistent, structured memory** of one person's body of work.
- The **research assistant** that operates through that memory.

The store is not a filing cabinet read on the side. The act of querying the structure *is* the research. Recalling a fact and finding an unsuspected link between two researchers are the same descent through the same graph.

The structure is the **conceptric**: documents folded once, at ingest, into levels of generality – atomic facts at the base, clusters above, a small layer of themes at the top. **Recall is a descent; generation is the same descent reversed.**

One problem of representation

Both gaps are problems of representation. The same meaning rides in language or in vectors, and a question expensive in one is cheap in the other.

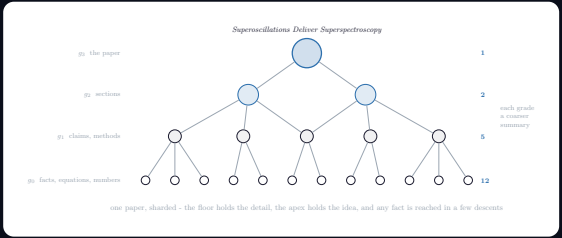
- Whether one concept contains another: an argument in prose, one inner product in vectors.
- Whether two records are the same person: an essay in prose, a membership test in a set.

A good summary keeps what its documents share and discards the rest – the information bottleneck. EigenEngine does not *train* that code. It *approximates* it by grouping documents that share content, then reads the structure instead of re-deriving it.

Grow the representation backwards

Fix the structure first. Pay for intelligence once, when a thing is understood. Let every later recall be arithmetic.

- Each paper is placed in a cluster, each cluster under a theme, at ingest – each grouping decided by content and recorded with its score.
- The top layer holds one summary node per theme, not one per document.



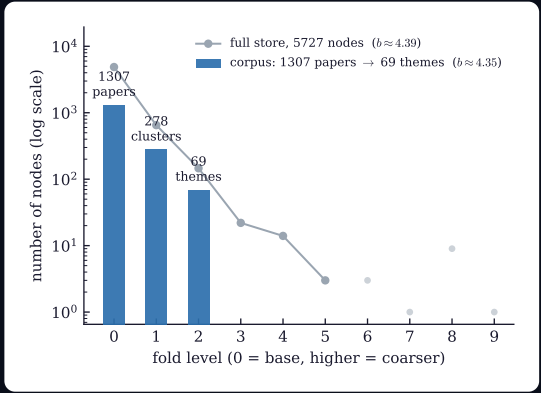
Recall is a short descent

One corpus of 1,307 papers folds geometrically:

1,307 papers $\rightarrow$ **278** clusters $\rightarrow$ **69** themes.

Branching $b \approx$ **4.4**, so depth grows as $\log_b N$, not with N. The same slope holds in the engine's full store (≈5,700 nodes, ten grades).

Intelligence is spent once, in the build. The query is the cheap walk down.

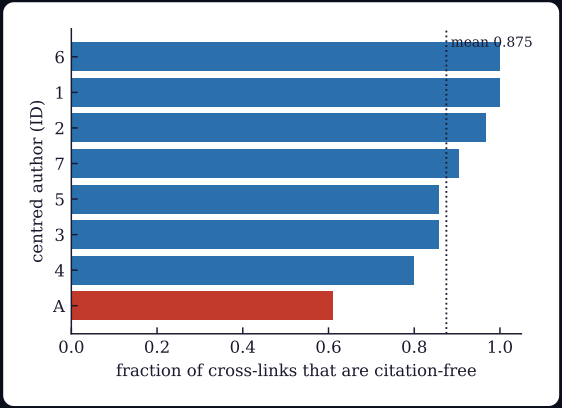


A query no citation graph can answer

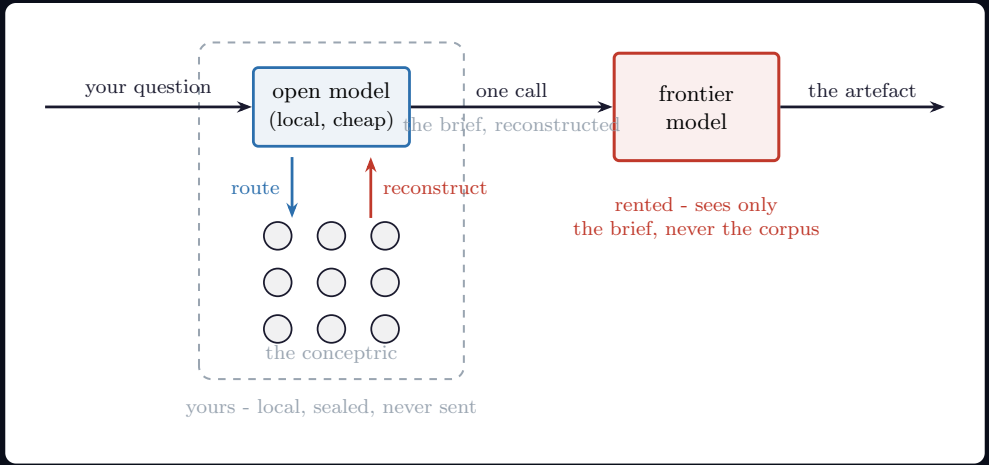
Cross-link one researcher's themes to every collaborator – a set intersection over the apex:

- **0** model inferences.
- **69** theme nodes touched; **1,307** papers never read.
- Links between authors who **share no paper** – not explained by collaboration.

Across all eight centres the citation-free fraction is mean **0.875** (range 0.61–1.00), lowest where real co-authorships exist and are correctly recovered.



Generation is recall in reverse



A small local model walks the question down the structure and hands a frontier model one short brief. The frontier model never sees the corpus; intelligence is rented by the sentence.

Why not just retrieval-augmented generation?

A typical RAG or agent harness re-embeds documents and spends a model call on *every* query, over a flat, stateless index, returning existing passages by surface similarity.

- Here the intelligence is paid **once**, at fold time.
- Structural queries run by arithmetic over a small top: **zero** model calls, cost **logarithmic** in corpus size.
- One **persistent** substrate, reused and grown across sessions.
- It recovers links as first-class structural objects that appear in **no single document** – a flat search returns them only as text, never as a link.

What is closed, and what is not

- **Closed:** persistence. The graded structure is built and queried cheaply on one researcher's machine, every number reproducible.
- **Open:** voice and reasoning are still a general-purpose model wearing instructions. Making them the system's own needs an open-weight model trained on the structure.
- **Open:** larger corpora and a self-hosted generator need compute beyond one person.

This talk and its companion paper were projected by the engine they describe, from its own memory.

EigenEngine

$\hat{\mathrm{E}}_i|\mathrm{gen}\rangle \otimes \hat{\mathrm{E}}_n|\mathrm{gen}\rangle$

Full paper: *Persistence and generation as problems of representation*

gmccaul.co.uk

Eigen@gmccaul.co.uk