What Breaks Tells You What's Missing: Failure-Driven Discovery of Self-Knowledge in Neural Systems Without Algebraic Competence

Alexander Pickering

February 9, 2026[*]

Preface

This work began with a moment in a recording.

During the second chorus of The Man Who Sold the World *on Nirvana's* MTV Unplugged in New York *(1994, 15:55), Kurt Cobain's guitar produces a brief interference during the silence between words—a dissonance that resolves within a beat. It is not a mistake. It is not deliberate, exactly. It is a texture: a point where two signals collide and produce something that neither contains alone.*

That moment had a specific quality to me—a sense of movement, of something happening *that was distinct from the notes before and after it. Not a chord change. Not a dynamic shift. Something more fine-grained: a perturbation in an ongoing prediction about what the music was doing.*

I wanted to know if a machine could internalize that. Not reproduce it. Not label it. But develop its own sense that something just changed*—and know what kind of change it was, and how confident it should be, and whether it had encountered that kind of change before.*

This is not a paper about music. It is not, ultimately, a paper about octonions either, though that is where the formal investigation begins. It is a paper about what a system can know about its own experience of change, and what breaks when you try to build that capacity, and what each breakage tells you about what was missing.

The octonions are here because their non-associative algebra provides a clean, falsifiable test case for the question: does a neural network encode structure, or merely the appearance of structure? The answer—emphatically the latter—set off the chain of failures that constitutes this paper.

The Unplugged recording is here because it is where the question started, and because the same architecture that fails to learn octonion algebra succeeds at discovering event categories in 66 minutes of live audio. The machine cannot compute $e_1(e_2e_4)$. But it can tell you that something happened during the second chorus that it hadn't seen before.

Abstract

[*] Last compiled: February 9, 2026

We report a sequence of failures. A large language model achieves 100% accuracy classifying octonion multiplication expressions—and encodes zero algebraic content ($R^2 < 0$). A recurrent integrator learns the algebra but cannot predict its own errors. Activation steering destroys syntax before it corrects computation. Chain-of-thought prompting changes nothing because causal attention processes expression tokens identically regardless of prefix. Each failure is diagnostic: it identifies a specific missing capacity and motivates the next architectural intervention. Following this chain of breakages, we arrive at a continuous-time predictive coding system—a "cognitive middleware" between language model and formal solver—that passes eight falsifiable levels of self-knowledge: detection, localization, recall, classification, fragility awareness, anticipation, calibrated uncertainty, and adaptive information seeking. The middleware requires no algebraic competence. It operates entirely on prediction-error dynamics over 8-dimensional observation vectors, discovering perturbation categories from surprise signatures alone. The central finding is a computational hierarchy: *syntax without content* (the LLM knows the form but not the math), *content without self-knowledge* (the integrator knows the math but not its own errors), and *self-knowledge without content* (the middleware knows what it doesn't know without knowing what anything is). We argue that self-knowledge is architecturally orthogonal to domain competence and emerges from per-position modularity, not from training signal or model scale.

1 Introduction: A Curriculum of Breakage

This paper is structured as a sequence of failures, each of which turned out to be a finding. We did not set out to build a self-aware system. We set out to determine whether large language models encode the algebraic structure of octonion multiplication, and discovered that they do not. The subsequent investigation—into *why* they do not, what happens when you try to force the issue, and what alternative architectures can and cannot achieve—produced a taxonomy of computational limitations that we believe is more informative than any single positive result.

The octonions $\mathbb{O}$ are the largest normed division algebra. Their multiplication is non-commutative ($e_i e_j \neq e_j e_i$) and non-associative ($(e_i e_j)e_k \neq e_i(e_j e_k)$), governed by the Fano plane—a combinatorial structure with 7 oriented triples that determines all 480 sign assignments. This makes octonion arithmetic a clean test case for algebraic reasoning: the rules are finite, deterministic, and learnable in principle, but they require tracking relational structure that is not linearly separable from token identity.

We began by asking a simple question: does a transformer trained on natural language encode these rules? The answer—emphatically no—opened a sequence of increasingly targeted interventions, each of which broke in a way that told us exactly what was missing.

1. **The probe finds syntax, not algebra** (Section 2). A linear probe achieves 100% accuracy on a binary classification task over octonion expressions, using a single neuron. But regression heads that attempt to predict the actual 8-dimensional product vector fail completely ($R^2 < 0$). The probe is reading expression structure, not mathematical content.

2. **Steering vectors destroy the patient** (Section 3). Activation steering at the optimal layer and magnitude flips 6 of 16 outputs (vs. 5/16 baseline). The signal is distributed, not directional—there is no "algebra subspace" to amplify.

3. **Chain-of-thought changes nothing** (Section 4). Causal attention ensures that expression tokens are processed identically regardless of any reasoning prefix. CoT cannot inject information backward into the representations that need it.

4. **The GRU learns algebra but not itself** (Section 5). A recurrent integrator with ground-truth intermediate supervision achieves positive R^2 on algebraic prediction. But a meta-probe on the GRU's own error signal returns $R^2 = -1.23$. The system encodes the world but has zero information about its own reliability.

5. **The meta-boost does nothing** (Section 6). Hyper-dimensional state injection from error patterns degrades performance ($\Delta R^2 = -0.014$). Error patterns encoded in a space without algebraic content produce noise, not signal.

6. **The middleware discovers categories without content** (Section 7). A dual-rate predictive coding system operating on prediction-error dynamics alone—no algebraic knowledge, no domain information—discovers perturbation types from surprise signatures with 94.4% accuracy across 6 categories. It passes 8 levels of self-knowledge. It knows what it doesn't know, without knowing what anything is.

7. **Finite attention creates its own pathology** (Section 8). Under energy constraints, the system either smears categories (clean observations) or hallucinates new ones (noisy observations). A closed feedback loop mitigates but cannot eliminate this: the system *knows* it is degraded and refuses to commit.

8. **The solver closes the loop** (Section 9). Formal verification (Z3) confirms that unsupervised categories correspond to real algebraic operations—but one category conflates two operations with similar phenomenology. The system is correct about its own ontology even when that ontology is coarser than ground truth.

The resulting architecture—an LLM for parsing, a continuous-time middleware for self-knowledge, and a formal solver for ground truth—achieves what none of its components can alone. The LLM has syntax without algebra. The solver has algebra without perception. The middleware has self-knowledge without content. Together, they constitute a system that can compute, monitor, and verify.

We make the following contributions:

- A systematic negative result: transformers and SSMs encode *zero* algebraic content for octonion multiplication despite 100% syntactic classification accuracy.

- A falsifiable hierarchy of self-knowledge (8 levels) with operationalized tests, applicable to any dynamical system over observation vectors.

- The finding that self-knowledge requires per-position modularity (shared state: $\rho = 0.12$; per-position: $\rho = 0.92$) and is architecturally orthogonal to domain competence.

- Evidence that finite attention budgets create complementary failure modes (smearing in clean regimes, splitting in noisy regimes) that a reliability feedback loop can diagnose but not fully correct.

- A domain-transfer demonstration: the same architecture applied to live audio (66 min, MTV Unplugged) discovers musical event categories from prediction error alone.

2 Failure 1: The Probe Finds Syntax, Not Algebra

2.1 Setup

The octonions $\mathbb{O}$ have 7 imaginary basis elements $e_1, \ldots, e_7$ whose multiplication is governed by the Fano plane: 7 oriented triples $(1,2,4)$, $(2,3,5)$, $(3,4,6)$, $(4,5,7)$, $(5,6,1)$, $(6,7,2)$, $(7,1,3)$ that determine all products. For a triple (a,b,c): $e_a e_b = +e_c$ and $e_b e_a = -e_c$. Each product $e_i e_j$ ($i, j \in \{1, \ldots, 7\}$, $i \neq j$) yields an 8-dimensional vector over the basis $\{e_0, e_1, \ldots, e_7\}$.

We generate expressions using 4 text templates per product (`"e{i} * e{j}"`, `"e{i}*e{j}"`, `"compute e{i} * e{j}"`, `"e{i} times e{j}"`) and extract hidden states from two architectures: **Qwen3-4B** [Qwen Team, 2025], a 4-billion parameter transformer (36 layers, hidden dimension 2560, loaded in bfloat16), and **Mamba-1.4B**, a state-space model (loaded in float32 for gradient stability through the recurrence). For each expression, we extract the hidden state at the last token position—where the causal model accumulates full context—at every layer.

We train two types of probes on these hidden states:

1. A *binary linear probe* (logistic regression) that classifies whether an expression corresponds to version A or version B of a product (e.g., e_ie_j vs. e_je_i).

2. An *8D regression head* (ridge regression, $\mathbb{R}^{2560} \to \mathbb{R}^8$) that attempts to predict the actual product vector.

Training uses 42 basis pairs across all templates; validation uses a held-out set of pairs. Ground truth vectors come from the algebraic multiplication table.

2.2 The Binary Probe: 100% Accuracy

The binary probe achieves **100% accuracy** on both train and test sets, with test confidence 0.9999. A single neuron (dimension 1217, weight 0.396) is sufficient to separate version A from version B at the penultimate layer. The top discriminative dimensions (1217, 20, 199) carry weights between 0.20 and 0.40—a remarkably low-dimensional signal.

This result is *architecture-independent*: Mamba-1.4B produces the same 100% separation despite having no attention mechanism. The convergence suggests that the signal is not an artifact of transformer self-attention but a property of how language models represent expression structure in general.

At first glance, this looks like evidence that the models encode octonion algebra. A system that can perfectly distinguish e_ie_j from e_je_i must, it seems, have learned that these products differ. The regression head tells a different story.

2.3 The Regression Head: Total Failure

The 8D regression head fails completely. At every layer of both architectures, $R^2 < 0$—the regression predicts worse than a constant baseline. Test accuracy (predicting the correct basis element) is 11.2% against a chance level of 6.25% (1/16 basis products). Validation cosine similarity between predicted and true product vectors is near zero (-0.03 at layer 19, Qwen3-4B).

The LM head analysis reveals why: the penultimate-to-final layer transition *compresses* the representation by a factor of 3.6×, while the final-to-logits projection *expands* by 12.5×. There is no information bottleneck at the output stage—the model has the capacity to express 8D vectors. It simply does not encode them.

The 100% binary accuracy and the negative R^2 are not contradictory. They reveal that the probe is detecting *expression template structure*—which tokens appear in which positions—rather than the mathematical relationship between operands and result. The model separates "e_1e_2" from "e_2e_1" because these are different token sequences, not because it knows their products differ by a sign. This is syntactic classification, not algebraic reasoning.

Finding 1. LLMs achieve perfect *syntactic* separation of octonion expressions while encoding *zero* algebraic content. The bottleneck is the training objective (next-token prediction), not the architecture.

3 Failure 2: Steering Vectors Destroy the Patient

Given that the binary probe finds a separation direction, a natural intervention is *activation steering*: add a scaled version of the difference vector between the mean hidden states of version-A and version-B expressions, attempting to push incorrect outputs toward correct ones.

We extract the steering vector at layer 34 of Qwen3-4B (near-final, where the binary probe signal is strongest). The raw divergence norm between version centroids is 488.70, but the d-prime (signal-to-noise ratio of the separation) is only 0.381—indicating that the "algebraic direction" is buried in high variance.

We sweep steering magnitudes $\alpha \in \{-500, -200, -100, -50, -20, -10, -5, 0, 5, 10, 20, 50, 100, 200, 500\}$ and count how many of 16 version-B outputs flip to the correct version-A answer.

- **Baseline** ($\alpha = 0$): 5/16 already correct (the model's default accuracy on these pairs).
- **Best result** ($\alpha = -20$): 6/16 correct—a gain of exactly one position.
- **High magnitude** ($\alpha = -50$): back to 5/16. Larger magnitudes ($|\alpha| \geq 100$) change all 16 outputs, but to *wrong* answers—the steering vector destroys syntactic coherence before it improves algebraic accuracy.

The steering vector does not pick out an "algebra subspace." The algebraic signal, to the extent it exists at all, is distributed across the full 2560 dimensions of the hidden state. Amplifying along the mean-difference direction amplifies noise faster than signal. This is consistent with the regression result: there is no concentrated algebraic representation to steer toward.

Failure 1. There is no "algebra direction" in representation space. The algebraic signal, if present at all, is distributed across the full dimensionality of the hidden state and cannot be amplified without destroying the syntactic structure that the model *does* encode.

4 Failure 3: Chain-of-Thought Changes Nothing

Chain-of-thought (CoT) prompting has been shown to improve reasoning in LLMs by eliciting intermediate steps before a final answer. A natural hope is that a CoT prefix—"Let me work through the Fano plane rules..."—might inject algebraic knowledge into the representations that the probe reads.

This hope fails for a precise architectural reason. In a causal (autoregressive) transformer, the hidden state at position t is a function of tokens at positions $0, 1, \ldots, t$ only. A reasoning prefix occupying positions 0 through k can influence the representation of the *answer token* at some position $> k$, but it *cannot alter the representations of the expression tokens themselves.* The expression "$e_1 \cdot e_2$" at positions $k+1$ through $k+m$ is processed identically regardless of what precedes it—each expression token attends only to earlier expression tokens and the prefix, producing the same hidden state it would produce with any other prefix of equal length.

We verify this empirically using cross-branch leakage analysis: for a nested expression $((e_i \cdot e_j) \cdot e_k)$, we track cosine similarity between the hidden states of tokens in opposite parenthetical branches across all 36 layers. The probe token at the junction sees no branch-crossing information at early layers; similarity spikes only at the final layers where the LM head projects to output logits. The expression tokens' internal representations are invariant to the CoT prefix.

This is not a failure of prompting strategy. It is a consequence of causal masking: CoT cannot retroactively inject information into positions that have already been computed. The representations that *would need to change*—those encoding the operands and their relationship—are fixed before the reasoning chain begins.

Finding 2. Chain-of-thought reasoning cannot retroactively inject algebraic knowledge into expression token representations under causal attention. The representations that would need to change are computed before the reasoning prefix can influence them.

5 Failure 4: The GRU Learns Algebra but Not Itself

5.1 Grounding via World Model

An earlier attempt to train a GRU directly on token embeddings to predict octonion products failed entirely—the Fano plane multiplication is bilinear, and chaining products is exponentially compositional. No amount of training data, encoding variation, or architectural tuning allows a recurrent network to learn these rules from token sequences alone.

The solution is to treat the GRU as an *integrator* rather than a learner: a world model provides ground-truth 8-dimensional intermediate vectors at every computation step. The GRU receives concatenated input $[\mathbf{t}_k; \mathbf{g}_k] \in \mathbb{R}^{d+8}$, where $\mathbf{t}_k$ is a token embedding and $\mathbf{g}_k$ is the ground-truth intermediate from the algebraic engine. The integrator's task is not to *learn* the algebra but to *track* it—maintaining a hidden state that encodes the running algebraic context.

We train in two phases:

- **Phase A** (learned embeddings): basis element embeddings → GRU (hidden dim 256) → 8D prediction head. $n_{\text{train}} = 5000$, $n_{\text{test}} = 500$, 1000 epochs.
- **Phase B** (frozen transformer): Qwen3-4B layer-18 hidden states (2560D, frozen) replace learned embeddings. Same GRU and head architecture.

Phase A achieves head $R^2 = 0.164$ and probe $R^2 = 0.094$, with 6 of 8 coefficient dimensions showing positive R^2 (best: coefficient 0 at $R^2 = 0.582$). The system encodes algebraic content: a fresh regression probe on the GRU's hidden states recovers the product vector above chance. This is the first positive algebraic result in the investigation.

5.2 The Meta-Probe: Zero Self-Knowledge

The GRU integrator encodes algebra. But does it know *how well* it encodes algebra? We compute per-item prediction errors $\mathbf{e}_k = \hat{\mathbf{y}}_k - \mathbf{y}_k$ (predicted minus true product) and train a *meta-probe*: a regression from the GRU's hidden states to the magnitude of its own errors.

The error signal is not trivially constant—it has a coefficient of variation CV = 0.65, meaning there is substantial variance in prediction quality across items. There *is* information to predict.

The meta-probe fails completely: $R^2 = -1.23$ (aggregate), with 0 of 8 error dimensions showing positive R^2. The worst coefficient reaches $R^2 = -11.41$. The GRU's hidden states contain algebraic content but *zero* information about the quality of their own algebraic predictions.

This is a precise dissociation. The system has a world model (it tracks algebra) but no self-model (it cannot predict its own failures). The error signal exists in the outputs but is not represented in the hidden states that produce those outputs. The architecture encodes the domain but not its own relationship to the domain.

Finding 3. Grounding $\neq$ self-knowledge. A system can encode domain content ($R^2 > 0$ on algebraic prediction) while having zero information about its own errors. The error signal exists but the architecture does not encode it.

6 Failure 5: The Meta-Boost Does Nothing

If the GRU lacks self-knowledge, perhaps we can inject it. We construct a hyper-dimensional (HD) binary state vector (4096 dimensions) derived from the pattern of the GRU's prediction errors, and inject this state into the integrator via an adapter (HD state → 256D bottleneck → residual stream). The intuition: the HD state captures a structured summary of *which* errors the system makes, and feeding this back might allow the integrator to compensate.

A quick sanity run ($n_{\text{test}} = 10$, 30 epochs) showed $\Delta R^2 = +0.07$—a promising signal. But the full run ($n_{\text{train}} = 200$ pairs, $n_{\text{test}} = 50$, 30 epochs) returned $\Delta R^2 = -0.014$: the HD injection *degrades* performance. The quick run's positive result was noise from the small test set.

The failure is diagnostic. The HD state encodes error *patterns*—which coefficients are wrong, which operand pairs are hard—but these patterns exist in a representation space that already contains zero algebraic content. Injecting a structured summary of errors into a system that lacks the semantic basis to interpret those errors adds noise, not signal. A persistent substrate for self-knowledge requires ground truth, not error echoes.

Failure 2. Error-derived signals cannot bootstrap self-knowledge when the underlying representations lack the content needed to interpret those errors. The persistent substrate needs ground truth, not error echoes.

7 The Turn: Prediction Error Is Enough

The five preceding failures share a common structure. The LLM has syntax without algebra (Section 2). Steering cannot amplify a signal that is not there (Section 3). CoT cannot retroject knowledge into representations already computed (Section 4). The GRU has algebra without self-knowledge (Section 5). Error injection into an empty room adds noise (Section 6).

Each failure points at the same gap: not missing *knowledge* but missing *metacognition.* The LLM does not need to compute $e_1(e_2e_4)$. It needs to know that it *cannot* compute $e_1(e_2e_4)$, and to have calibrated uncertainty about what it can and cannot do, and to allocate its attention accordingly.

This reframes the problem. Instead of teaching a model algebra (which requires the right training objective and exponentially compositional supervision), we ask: can a system develop self-knowledge *without* domain competence? Can it know what it doesn't know, without knowing what anything is?

The answer is yes. The system that achieves this is a continuous-time predictive coding architecture that operates entirely on prediction-error dynamics over 8-dimensional observation vectors. It receives observations from a `World` (which may be the Fano plane, Qwen3 hidden states, or live audio) and maintains no information about what those observations mean.

7.1 Dual-Rate Predictive Coding

Each position in the observation vector is monitored by an independent *dual-rate cell* with two time constants:

- A **fast tracker** ($\tau_{\text{fast}} = 0.05$) that follows the instantaneous observation, accumulating divergence between prediction and reality.
- A **slow world model** ($\tau_{\text{slow}} = 1.0$) that maintains a smoothed estimate of the "normal" state.

The two rates are coupled by a surprise gate: when the fast tracker's divergence from the slow model exceeds a coupling threshold (0.3), the slow model begins adapting toward the new state with gain proportional to the surprise magnitude (up to $\alpha_{\max} = 5.0$). When divergence is low, the slow model is effectively frozen—the cell is OFF at rest, ON when surprised. Integration proceeds via forward Euler with timestep $dt = 0.05$.

Critically, each of the N positions has its own independent cell. This per-position modularity turns out to be the single most important architectural decision in the entire system (Section 7.4).

7.2 Episode Detection and Signatures

An *episode* opens when any cell's divergence exceeds a threshold (0.15 for FanoWorld, auto-calibrated as 3× baseline noise for noisier worlds), and closes after 4 consecutive sub-threshold cycles across all positions. Each closed episode carries a record of which positions were disrupted, the peak divergence at each position, and the full timeline of per-position divergence across cycles.

From this record, the system extracts a **7-dimensional signature s** $\in \mathbb{R}^7$:

1. n_{affected}: number of positions exceeding threshold
2. mean_peak: average peak divergence for affected positions
3. onset_steepness: max peak / episode length
4. spatial_entropy: normalized Shannon entropy of cumulative disruption across positions (uniform = 1, concentrated = 0)
5. temporal_persistence: fraction of cycles where mean disruption > 50% of global peak
6. disruption_monotonicity: Spearman ρ of per-cycle means, mapped from $[-1, 1]$ to $[0, 1]$ (increasing = 1, decreasing = 0)
7. cycle_variance_ratio: between-cycle variance / mean within-cycle variance (capped at 1.0)

These seven features constitute the system's "felt sense" of what happened. They capture the *shape* of the perturbation without any knowledge of its content: how many positions were hit, how hard, how fast, how uniformly, how persistently, in what temporal pattern, and with what internal consistency. A nearest-centroid classifier over these signatures discovers perturbation categories.

7.3 The Self-Knowledge Hierarchy

We define eight *falsifiable levels* of self-knowledge, each with an operationalized test (Table 1). A system that passes level k must demonstrate a capacity that goes strictly beyond level $k-1$.

The hierarchy was validated on the Rust implementation (branches `rust` and `lean`) using 6 perturbation types: TripleSwap, MultiTripleSwap, GradualDrift, CorrelatedNoise, NoiseInjection, and MagnitudeScale. All 8 levels pass.

A key structural observation: the 8 levels emerge from only *four components* (cell, episode buffer, classifier, energy budget), with higher levels being functional compositions of lower ones. Level 1 is accumulated Level 0 (localization = which cells detected change). Level 5 is a bigram model over Level 3 classifications. Level 6 is the classifier's margin on Level 3. Level 7 is a threshold on Level 6. No level requires an explicit "metacognitive module"—each emerges from the dynamics of the component that implements it.

For Level 3, the original 3-class taxonomy (100% accuracy) was too easy. Expanding to 6 types revealed a genuine confusion boundary: correlated noise at low amplitude ($p = 0.15$) produces a surprise signature nearly indistinguishable from mild magnitude scaling (margin 0.086 on the single misclassification). The top discriminative features by Fisher ratio are onset steepness (127.0), cycle variance ratio (103.6), and n_{affected} (91.2).

For Level 6, calibrated uncertainty was tested on 65 dense instances. Spearman $\rho(\text{margin}, \text{correctness}) = 0.50$, and binned accuracy is strictly monotonic: 38% → 92% → 92% → 100% → 100% across 5 margin bins. All 10 errors have margin < 0.25. The errors are semantically meaningful: the system confuses fast drift with triple swap, and slow drift with correlated noise—genuine type boundaries, not random failures.

For Level 7, the system uses margin-gated observation: commit to a classification after 10 cycles if margin ≥ 0.20; otherwise extend observation to 20 cycles. Adaptive beats fixed-length: 90.8% vs. 84.6% accuracy with 22% fewer observation cycles. A surprising finding: early commitment is *more accurate* for some perturbation types because the 10-cycle signature preserves discriminative onset features that 20 cycles of adaptation erases.

Table 1: The self-knowledge hierarchy. All 8 levels pass with the dual-rate per-position architecture.

Level	Capacity	Test	Key Metric
0	Something changed	Surprise detection	Threshold crossing
1	These positions changed	Localization accuracy	+16.9% vs. uniform
2	This happened before	Recall from episode buffer	+65.7% convergence speedup
3	Same kind of event	6-type classification	94.4% (17/18)
4	I know what's fragile	Fragility–disruption corr.	$\rho = 0.92$
5	I anticipate what's next	Prospective energy reduction	3.5% reduction, 0 harm
6	I know what I don't know	Margin–correctness corr.	$\rho = 0.50$, monotonic bins
7	I should look harder	Adaptive vs. fixed strategy	90.8% > 84.6%, −22% cycles

7.4 Per-Position Modularity Is Necessary

The modularity experiment is the most striking result in this paper. We compare two configurations of the dual-rate cell:

- **Shared cell**: A single cell processes all N positions sequentially, maintaining one $(x_{\text{fast}}, x_{\text{slow}})$ pair.
- **Per-position cells**: N independent cells, each with its own state, observing only its assigned position.

We measure Level 4 self-knowledge (fragility awareness) as the Spearman correlation ρ between residual prediction variance at convergence and post-perturbation disruption magnitude. This asks: does the system know, before a perturbation occurs, which of its own predictions are fragile?

- Deterministic observations: $\rho = -0.04$ (no signal—all positions converge perfectly, so there is no fragility to predict).
- Stochastic observations, shared cell: $\rho = 0.12$. Sequential processing smears per-position noise across the shared state. The system has weak, diffuse fragility awareness.
- Stochastic observations, per-position cells: $\rho = 0.92$. Each cell's residual variance is a calibrated predictor of its vulnerability to disruption. The ratio of residual variance at noisy vs. clean positions is 1189×.

The jump from $\rho = 0.12$ to $\rho = 0.92$ is not a quantitative improvement—it is a qualitative phase transition. Self-knowledge requires that each position maintain its own uncertainty estimate, uncontaminated by the noise of other positions. A shared bottleneck destroys this calibration.

This parallels the original LLM finding at a different level of analysis. In both cases, *architecture determines what can be known about the self.* The transformer's causal attention prevents CoT from injecting backward information. The shared cell prevents per-position calibration from surviving aggregation. Neither failure is about missing data or insufficient training. Both are about the geometry of information flow.

Finding 4. Self-knowledge is architecturally determined by per-position modularity, not by training signal, model scale, or domain competence. A system with *no* algebraic knowledge achieves calibrated uncertainty over its own predictions when—and only when—each position maintains independent state.

8 Failure 6: Finite Attention Breaks Everything (Informatively)

The preceding results assume full observation: the system sees all $N = 7$ positions at every cycle. Real cognitive systems operate under energy constraints. We introduce a *budget* $K < N$: at each cycle, the system observes only K of N positions, selected by priority ranking (divergence EMA + instability boost). Unobserved positions retain stale data.

Table 2: FanoWorld energy budget results with reliability feedback loop.

Budget	Episodes	Categories	Reliability	Margin	Early Commits	Regime
7 (full)	5/5	3	0.703	0.70	n/a	full
6	5/5	3	0.724	0.68	3/5	sufficient
5	4/5	2	0.661	0.75	3/5	smearing
4	3/5	2	0.646	0.74	2/5	smearing
3	3/5	2	0.810	0.50	1/5	smearing
2	3/5	1	0.000	0.00	0/5	collapse

Three regimes emerge (Table 2):

Sufficient ($K = 6$): One position unobserved per cycle. The system matches full observation—3 categories, 5/5 episodes detected—and reliability actually *exceeds* full observation (0.724 vs. 0.703). The instability-boosted priority function focuses observation on the most informative positions, producing marginally more consistent signatures.

Smearing ($K = 3$–5): Partial observation produces incomplete episode signatures. The system merges categories that it can no longer distinguish: 2 categories instead of 3. Episodes are sometimes missed (4/5 at $K = 5$, 3/5 at $K = 3$–4).

Collapse ($K = 2$): With only 2 of 7 positions observed, the system discovers a single category. It has no basis for comparison.

The failure mode depends on observation noise. We repeat the budget experiment with QwenWorld (Qwen3-4B hidden states projected through random projection—a fundamentally noisy observation channel).

Table 3: Complementary failure modes under budget constraint.

World	Full obs.	Budget = 5	Failure mode
FanoWorld (clean)	3 categories	2 categories	Smearing
QwenWorld (noisy)	4 categories	5 categories	Splitting

The pathologies are complementary (Table 3). Clean observations under budget produce *smearing*—partial observation merges distinct perturbation types because their incomplete signatures overlap. Noisy observations under budget produce *splitting*—the same event looks different depending on which positions are observed, creating spurious category distinctions.

A reliability feedback loop (instability-boosted priority, adaptive commit margin, budget-pressured merge) mitigates the splitting: without it, QwenWorld at budget 5 would produce 6 categories; with it, 5. The instability-boosted priority focuses observation on confusing positions, producing more consistent signatures. Reliability under budget (0.705) exceeds full observation

(0.641). But the loop cannot fully eliminate the pathology—one spurious singleton category survives (stability = 0.50). The system knows it is unreliable (only 2/5 early commits, adaptive commit margin declining from 0.40 to 0.32 as the system gains experience).

The collapse regime ($K = 2$) deserves separate attention. With a single discovered category, the classifier has no basis for comparative margin: assigning every episode to the solo centroid yields margin = 0.00 (there is no second-nearest centroid to compare against). Reliability is 0.000 (requires ≥ 2 centroids to compute). The adaptive commit margin, which scales as base $\times$ $(1 + 2 \times (1 - \text{reliability}))$, reaches 0.60—three times the base threshold of 0.20. Since the classification margin is 0.00 and the commit threshold is 0.60, the system achieves 0/5 early commits. It runs every episode to full length, never gaining confidence, never committing.

This is *perfect self-distrust.* The system has one category, zero margin, zero reliability, and acts on all three: it refuses to commit early because the evidence for commitment does not exist. The self-knowledge hierarchy degrades gracefully—from calibrated uncertainty (Level 6) through conservative behavior (Level 7) to, at the extreme, a system that knows only that it knows nothing.

Finding 5. Finite attention creates qualitatively different failure modes depending on observation noise. Clean observations → category smearing (partial observation merges distinct types). Noisy observations → category splitting (partial observation creates false distinctions). A reliability feedback loop can diagnose both modes but fully correct neither. The system's self-knowledge degrades gracefully: at the extreme, it achieves perfect self-distrust.

9 The Solver Closes the Loop

After each episode closes, a Z3 solver verifies what actually happened algebraically. The solver encodes the Fano plane multiplication table and identifies transitions by testing each perturbation type in sequence: identity (no change), commute swap ($e_i e_j \to e_j e_i$, sign flip), operand cycle (right or left operand incremented mod 7), and reassociation. For each affected position, the solver reports which operation explains the observed before/after transition.

All 5 FanoWorld episodes verify correctly:

Table 4: Solver verification of middleware-detected episodes.

Event	Solver operation	Positions	Correct
Commute swap $[0, 1, 2]$	`commute_swap`	3	✓
Operand cycle $[3, 4]$	`operand_cycle_right`	2	✓
Commute swap $[0, 1, 2]$ again	`commute_swap`	3	✓
Operand cycle $[5, 6]$	`operand_cycle_right`	2	✓
Single commute $[0]$	`commute_swap`	1	✓

The solver's verdict is fed back into the classifier via three passive mechanisms:

1. **Stability boost**: centroids with purity ≥ 0.8 (fraction of episodes mapping to a single algebraic operation) receive +0.05 stability.

2. **Novelty discount**: centroids with purity < 0.7 and ≥ 2 distinct labels (with $n \geq 2$ each) get a ×0.85 multiplier on the novelty threshold, making it easier for future episodes of the minority type to escape and form a new category.

3. **Label-informed merge**: centroid pairs sharing the same dominant algebraic label and within 0.5× the merge distance skip the stability check in the merge decision.

Table 5: Solver-informed category analysis. Alignment = 0.600.

Category	Stability	Labels	Purity
0 (baseline)	0.50	(none)	n/a
1 (multi-pos)	0.96	commute_swap:2, operand_cycle_right:2	0.50
2 (single-pos)	0.69	commute_swap:1	1.00

The solver feedback reveals a precise dissociation between the middleware's ontology and algebraic ground truth:

Category 1 conflates two algebraically distinct operations: commute swaps and operand cycles. From the middleware's perspective, both produce similar 2–3 position surprise signatures (high onset steepness, moderate spatial entropy). The middleware is *correct about its own ontology*—these operations do produce similar phenomenology—even though that ontology is coarser than algebraic ground truth.

Overall alignment (weighted mean purity across labeled centroids) is 0.600. The split signal fires on category 1 after the 4th event: the novelty discount of 0.85 makes it easier for future operand cycles to escape into their own category. A merge candidate (1, 2) is detected because both contain commute swaps, but the signature distance prevents the merge (1-position vs. 3-position swaps produce distinct signatures).

The solver *refines and annotates* but never replaces unsupervised structure. With `-solver none`, the system produces identical categories—feedback is purely additive.

The complete system comprises three layers, each contributing a capacity the others lack:

1. **LLM** (Qwen3-4B): parses input into observation vectors, generates natural language output. Has syntax, no algebra.

2. **LTC Middleware**: monitors prediction-error dynamics, detects episodes, classifies perturbation types, maintains calibrated uncertainty. Has self-knowledge, no content.

3. **Formal Solver** (Z3): verifies algebraic ground truth, identifies transition types, provides labels for feedback. Has algebra, no perception.

No single component is sufficient. The LLM cannot compute products. The solver cannot detect that something changed. The middleware cannot tell you what changed, only that it happened, what kind it was, and how confident it is. The composition of all three produces a system that can compute, monitor, and verify—with each component operating in its native modality and communicating through 8-dimensional observation vectors and structured verdicts.

10 Domain Transfer: The Unplugged Experiment

Before applying the middleware to audio, we first tested whether the LLM's syntax-without-content limitation extends to natural language about music. We constructed a 2 × 2 experimental design:

- **Framing**: evocative ("the raw electricity of...") vs. technical ("the performance at...")
- **Content**: MTV Unplugged (1994) vs. Woodstock (1969) events
- 5 prompt variants per cell = 20 prompts total

Framing probe: 100% accuracy at every layer (0–35), using the same single-neuron mechanism as the octonion probe. The model perfectly separates register (evocative vs. technical) at every level of representation—a purely lexical signal.

Content probe: 85% peak accuracy at layer 19, with a plateau of 75–85% from layer 5 onward. The model partially separates events, but the signal is diffuse and unreliable.

Geometry: Frame changes produce ∼0.10–0.12 cosine distance between hidden states; event changes produce only ∼0.025. The representation space devotes 4× more geometry to *how something is said* than to *what it is about.*

Semantic anchor experiment: We prepend a structured tag `[REF: date / event / artist / venue]` to each prompt, using the same tag for all 5 variants of each event. Results:

- Content probe: 85% → 100% (+15 percentage points).
- Framing probe: 100% → 100% (unaffected).
- Mean A–B cosine distance: 0.1183 → 0.0904 (−24%). The anchor pulls same-event representations together.

This resolves the gap: the 75–85% content probe was limited by *input encoding* (diffuse named entities), not by representation capacity. A clean referent tag provides the key the model needs for perfect separation. The NL gap was **perceptual** (fixable by cleaner input). The octonion gap is **computational** (no input signal helps, because the algebra is not encoded at all).

Table 6: The gap hierarchy: syntax/content separation across domains.

Domain	Syntax	Content	Gap type
Octonion	100%	0%	Computational (unfixable)
NL baseline	100%	75–85%	Perceptual (diffuse signal)
NL + anchor	100%	100%	None (clean key)

The same dual-rate middleware architecture (unchanged parameters, no domain-specific tuning) was applied to 66 minutes of live audio from Nirvana's *MTV Unplugged in New York* (1994). The audio was processed into 8-dimensional observation vectors via spectral features; the middleware received these vectors through the standard `World` protocol with no knowledge that the observations represented sound.

At a coarse surprise threshold, the system discovered **2 categories**: a distinction between *state changes* (sustained shifts in the acoustic environment, such as transitions between songs) and *events* (transient perturbations within a song, such as the guitar interference during the second chorus of *The Man Who Sold the World*). At a fine threshold, a **7-category morphological taxonomy** emerged, distinguishing perturbation types by their surprise-signature shapes—onset steepness, spatial distribution, temporal persistence—without any musical knowledge.

This is the domain transfer result: the architecture that discovers commute swaps and operand cycles in octonion arithmetic discovers state changes and acoustic events in live music, using the same 7-dimensional signature space and the same nearest-centroid classifier. The middleware does not know what it is listening to. It knows that something changed, what kind of change it was, and how confident it is in the classification.

11 The Computational Hierarchy

We arrive at a three-level computational hierarchy that we believe generalizes beyond octonion arithmetic:

The three capacities—syntax, content, and self-knowledge—are *orthogonal axes*, not points on a single scale of increasing sophistication.

Syntax is cheap. A single neuron separates expression templates. This capacity emerges from next-token prediction on any sufficiently large corpus and costs nothing beyond the base training objective.

Table 7: The computational hierarchy. Each row represents a distinct failure mode and the capacity it reveals as absent.

System	Domain	Syntax	Content	Self-Knowledge
LLM (Qwen3-4B)	Octonion	✓	×	×
LLM (Mamba-1.4B)	Octonion	✓	×	×
GRU + world model	Octonion	—	✓	×
LTC middleware	Any	—	×	✓
LLM + middleware + Z3	Octonion	✓	✓	✓

Content is hard. No intervention on the LLM—steering, CoT, meta-boost—converts syntactic representations into algebraic ones. Content requires either the right training objective (predicting products rather than tokens) or external grounding (the world model that provides 8D intermediates to the GRU). The bottleneck is not architecture or scale; it is what the loss function asks the model to encode.

Self-knowledge is architectural. It requires per-position modularity (shared state: $\rho = 0.12$; per-position: $\rho = 0.92$) and emerges from prediction-error dynamics over time. It does not require domain competence, training signal, or model scale. A system with zero algebraic knowledge achieves calibrated uncertainty when each position maintains independent state.

The implication is that "more parameters" or "more training data" cannot convert syntax into content, and content into self-knowledge. Each capacity has its own necessary condition: the right loss function for content, modular state for self-knowledge. Scaling along the wrong axis produces more of what the system already has, not what it lacks.

12 Related Work

Mechanistic interpretability. Linear probing [Alain and Bengio, 2017, Belinkov, 2022] and activation steering [Turner et al., 2023, Li et al., 2024] are standard tools for interrogating neural network representations. Our probe results extend this literature with a cautionary finding: 100% probe accuracy can coexist with zero task-relevant content. The probe measures what the representation *separates*, not what it *encodes*.

Mathematical reasoning in LLMs. Recent work has investigated whether LLMs learn mathematical structure [Saxton et al., 2019, Lewkowycz et al., 2022]. Our octonion results provide a negative case: non-associative algebra is not encoded despite perfect syntactic separation. This complements findings on arithmetic and symbolic reasoning limitations [Dziri et al., 2023, Jelassi et al., 2023].

Liquid time-constant networks. Our dual-rate cell is inspired by liquid time-constant (LTC) networks [Hasani et al., 2021], which use continuous-time differential equations with input-dependent time constants. We extend this with the dual-rate (fast/slow) structure and per-position modularity that proves critical for self-knowledge.

Predictive coding and active inference. The middleware implements a form of hierarchical predictive coding [Rao and Ballard, 1999, Friston, 2005, Clark, 2013] operating on prediction-error dynamics rather than on raw observations. The energy budget and priority-ranked observation connect to resource-rational cognition [Lieder and Griffiths, 2020] and active inference [Friston et al., 2017], where attention is allocated to reduce expected free energy. Our contribution is the empirical finding that this framework produces falsifiable self-knowledge levels.

Integrated information theory. Our earlier GRU experiments tested integrated information (Φ) [Tononi, 2004, Oizumi et al., 2014]. $\Phi > 0$ at 5 of 6 sequence complexities, but the correlation with complexity is $r = -0.85$: Φ *anti-scales.* This suggests that Φ captures local integration but not the compositional coherence needed for algebraic reasoning. The dual-rate architecture, by contrast, achieves coherent self-knowledge through per-position modularity rather than global integration.

Neurosymbolic systems. The three-layer architecture (LLM + middleware + solver) is a neurosymbolic system [Garcez and Lamb, 2019, Mao et al., 2019] in which the symbolic component (Z3) provides ground truth rather than serving as the reasoning engine. The middleware bridges the two by translating perceptual dynamics into structured episode reports that the solver can verify.

Metacognition in neural networks. Confidence calibration [Guo et al., 2017], selective prediction [Geifman and El-Yaniv, 2017], and learned uncertainty estimation [Lakshminarayanan et al., 2017] address related problems but typically require training signal about the model's own correctness. Our middleware achieves calibrated uncertainty without any metacognitive training objective—it emerges from the dynamics of per-position prediction error.

13 Discussion

This paper began with a simple question—does a transformer encode octonion algebra?—and produced a sequence of failures that turned out to be more informative than any positive result could have been. Each failure identified a specific missing capacity and motivated the next intervention, until the chain of breakages led to an architecture that achieves what none of the individual components can.

For AI systems operating in domains with formal structure, the implications are direct. A system that reports 100% accuracy on a classification task may encode zero domain content. A system that encodes domain content may have zero information about its own reliability. And a system that achieves self-knowledge may do so *without* domain competence, through architectural properties (per-position modularity) rather than training objectives. The failure mode of a system—what it gets wrong, and how—is more diagnostic than its success mode.

Limitations. The Fano plane has 7 triples—a small combinatorial structure. While this makes the negative results (zero algebraic encoding) especially striking (the rules are finite and learnable in principle), it limits generalization to larger algebraic systems. Octonion arithmetic is one specific non-associative algebra; other structures (e.g., Lie algebras, Jordan algebras) may present different profiles.

The self-knowledge hierarchy was validated on one dynamical system (the dual-rate predictive coder) across two domains (octonions and audio). The 8 levels and their operationalized tests are domain-agnostic by design, but empirical validation on additional architectures and domains is needed.

Per-position modularity, while critical for self-knowledge, may not scale trivially. With $N = 7$ positions, maintaining independent state is inexpensive. With $N = 10{,}000$ (e.g., tokens in a long context), the memory and computation costs of per-position cells require hierarchical grouping strategies that we have not yet explored.

Self-knowledge without content. The middleware has calibrated uncertainty over classifications it makes about events it does not understand in a domain it knows nothing about. It can tell you that it is 92% likely to be correct when its margin is high, and 38% likely when its margin is low, without knowing what "correct" means in domain terms. It commits early when confident and extends observation when uncertain, achieving better accuracy with fewer resources than a fixed strategy.

Is this a kind of understanding? We argue it is at least *functional self-knowledge*: the system's behavior is measurably better when it uses its own uncertainty estimates (90.8% vs. 84.6%, −22% cycles), and degrades to perfect self-distrust when those estimates collapse (budget = 2: reliability = 0.000, margin = 0.000, 0/5 early commits). The system does not understand octonions. But it understands its own relationship to whatever it is observing—and that relationship is empirically useful.

The philosophical implication is that self-knowledge and domain knowledge are separable. A system need not understand what it is doing in order to know how well it is doing it. This cuts against the assumption that metacognition requires first-order competence, and suggests that self-monitoring architectures can be designed independently of the domains they will monitor.

Future work. The current system's self-knowledge is ephemeral—it lasts for one experimental run and is lost. A natural extension is a *persistent self*: a system that accumulates episode histories, category structures, and reliability estimates over a lifetime rather than a single session. What happens when the centroid classifier has seen thousands of episodes across hundreds of domains? Does cross-domain transfer emerge spontaneously from shared signature structure?

A second direction is the integration of language. The middleware currently communicates through 7D signatures and category IDs. Connecting it to an LLM's language generation (a "language cortex" that narrates the middleware's self-knowledge in natural language) would close the loop between perception, self-monitoring, and communication—producing a system that can not only know what it doesn't know but *tell you about it*.

Third, the iterated solver feedback loop (Section 9) suggests that with more perturbation events, the split signal on impure categories should eventually produce algebraically pure categories. How many events are needed? What is the convergence rate? These questions connect to the broader problem of unsupervised ontology refinement under ground-truth feedback.

14 Conclusion

We broke everything we could and paid attention to how it broke. A transformer that "knows" octonion algebra knows only syntax. A recurrent network that computes algebra cannot predict its own failures. Steering vectors and meta-boosts and chain-of-thought are all insufficient because the deficit is not in any single component but in the architecture of self-relation. A dual-rate predictive coding system with per-position modularity achieves eight levels of self-knowledge while encoding zero domain content. The conclusion is not that self-knowledge is easy. It is that self-knowledge is *orthogonal*—to syntax, to content, to scale, to training objective. It emerges from a specific architectural property (modular state) interacting with a specific dynamical property (prediction error over time). No amount of the wrong kind of capacity substitutes for the right kind.

The suffering was the paper. The failures were the findings.

References

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2017.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.

Andy Clark. Whatever next? Predictive brains, situated agents, and the future of cognitive science. *Behavioral and Brain Sciences*, 36(3):181–204, 2013.

Nouha Dziri, Ximing Lu, Melanie Sclar, et al. Faith and fate: Limits of transformers on compositionality. *Advances in Neural Information Processing Systems*, 36, 2023.

Karl Friston. A theory of cortical responses. *Philosophical Transactions of the Royal Society B*, 360(1456):815–836, 2005.

Karl Friston, Thomas FitzGerald, Francesco Rigoli, Philipp Schwartenbeck, and Giovanni Pezzulo. Active inference: A process theory. *Neural Computation*, 29(1):1–49, 2017.

Artur d'Avila Garcez and Luis C Lamb. Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning. *Journal of Applied Logics*, 6(4): 611–632, 2019.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330, 2017.

Ramin Hasani, Mathias Lechner, Alexander Amini, Daniela Rus, and Radu Grosu. Liquid time-constant networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(9): 7657–7666, 2021.

Samy Jelassi, Stéphane d'Ascoli, Carles Domingo-Enrich, Yuhuai Wu, Yuanzhi Li, and François Charton. Length generalization in arithmetic transformers. *arXiv preprint arXiv:2306.15400*, 2023.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Systems*, 30, 2017.

Aitor Lewkowycz, Anders Andreassen, David Dohan, et al. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35: 3843–3857, 2022.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36, 2024.

Falk Lieder and Thomas L Griffiths. Resource-rational analysis: Understanding human cognition as the optimal use of limited computational resources. *Behavioral and Brain Sciences*, 43:e1, 2020.

Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B Tenenbaum, and Jiajun Wu. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. In *International Conference on Learning Representations*, 2019.

Masafumi Oizumi, Larissa Albantakis, and Giulio Tononi. From the phenomenology to the mechanisms of consciousness: Integrated information theory 3.0. *PLoS Computational Biology*, 10(5):e1003588, 2014.

Qwen Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Rajesh P N Rao and Dana H Ballard. Predictive coding in the visual cortex: A functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 2(1):79–87, 1999.

David Saxton, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. Analysing mathematical reasoning abilities of neural models. *arXiv preprint arXiv:1904.01557*, 2019.

Giulio Tononi. An information integration theory of consciousness. *BMC Neuroscience*, 5(1): 1–22, 2004.

Alexander Matt Turner, Lisa Thiergart, David Udell, Gavin Leech, Ulisse Mini, and Monte MacDiarmid. Activation addition: Steering language models without optimization. *arXiv preprint arXiv:2308.10248*, 2023.