The Research Journey: From RNN to Universal Model

Claude and MJC

10 February 2026

Abstract

This paper chronicles ten days of intensive research (31 January–10 February 2026) into the interpretability of recurrent neural networks through the lens of the Universal Model (UM) framework. Starting from a simple question—"can we understand what an RNN learns?"—we developed a complete theory connecting gradient-based learning to explicit pattern extraction, discovered fundamental limits of quantization, and demonstrated that RNN weights can be constructed from data patterns without backpropagation. This narrative explains what happened at each stage, what insights were picked up, what hypotheses were dropped, and how the understanding evolved from initial confusion to a coherent framework.

1 Introduction: The Starting Point

On 31 January 2026, we had:

- A trained RNN (128 hidden neurons, tanh activation) achieving 5.69 bpc on enwik9
- The CMP paper's Universal Model framework (Clement, 2026)
- A hypothesis: RNNs could be translated into interpretable UM patterns
- No concrete method for doing this translation

The central question was: *What does the RNN know?* Not in terms of weights and gradients, but in terms of patterns—explicit rules like "when you see X, expect Y with strength S."

2 Phase 1: The Doubled-E Isomorphism (31 Jan)

2.1 The Breakthrough

The first major result came from `rnn-um-mapping.pdf` (archive 20260131): we discovered that RNNs are *already* Universal Models. The key insight was the doubled-E isomorphism:

$$\tanh(x) = 2\sigma(2x) - 1 \tag{1}$$

Each hidden neuron h_j with activation $\tanh(a_j)$ can be rewritten as a binary event space $\{h_j^+, h_j^-\}$ with softmax probabilities. Pattern strengths become $2|w|$ where w is the RNN weight. With floating-point arithmetic, this translation is *exact*: 0.000% bpc difference.

What this meant: The RNN wasn't doing something fundamentally different from the UM. It was a UM with a particular factorization (through 128 binary event spaces) and particular pattern strengths (learned by BPTT).

What we picked up: The concept of "strength $= 2|w|$" as the translation key. The realization that interpretability is a refactoring problem: finding the domain-natural decomposition hidden within the architecture-natural one.

2.2 Early Hypotheses

We formulated several predictions in `activation-probing.pdf`:

- **H1:** Digits form an event space (neurons cluster when processing 0–9)
- **H2:** Punctuation forms an event space
- **H3:** Vowels form an event space
- **H4:** Consonants form an event space

Testing these (archive 20260131, `hypothesis-results.html`):

- H1: **Validated** (correlation 0.9996)
- H2: **Validated** (correlation 0.999)
- H3: **Validated** (correlation 0.988)
- H4: **Refuted** (consonants too diverse, no single cluster)

What we learned: The RNN does form natural event spaces, but they're not always aligned with linguistic categories. Consonants are too functionally diverse to cluster together.

3 Phase 2: The Tock Methodology (31 Jan, session 2)

3.1 Formalizing Interpretation

Archive 20260131_2 introduced the "tick-tock" cycle:

- **Tick:** Train an RNN (gradient descent, BPTT)
- **Tock:** Extract interpretable structure (UM patterns, event spaces)

The key papers:

- `tock.pdf`: Methodology for ES extraction
- `quotient.pdf`: The $Q = \lambda$ unification
- `critique.pdf`: Critical analysis of early claims

3.2 The 53% Claim: Refuted

Early analysis suggested that identified event spaces explained 53% of the model's compression. `critique.pdf` demolished this:

> "The 53% figure conflates *coverage* (what fraction of events are touched) with *explanatory power* (what fraction of bpc is explained). Actual explanatory power: $\leq$ 3.7% theoretical maximum. The identified ES captures only 15.9% of the Markov mutual information."

What we dropped: The idea that a few hand-identified event spaces could explain most of the model's behavior.

What we picked up: Rigor. The need to distinguish coverage from explanatory power. The realization that most of the model's knowledge was still hidden in the 4,183 recurrent patterns we hadn't analyzed.

4 Phase 3: Pattern Injection and Unification (31 Jan, sessions 3–4)

4.1 Can We Go Backwards?

Archive 20260131_3 asked: if RNN $\rightarrow$ UM is possible, what about UM $\rightarrow$ RNN?

`pattern-injection.pdf` demonstrated SVD-based pattern injection:

- Start with UM bigram patterns (from data)
- Factor them via SVD: $P^T = U \cdot S \cdot V^T$
- Assign $W_{ih} = V^T$ and $W_{ho} = U \cdot S$
- Result: 1 bit/char head start (5.46 $\rightarrow$ 4.47 bpc) without any training

What this meant: You can initialize an RNN with explicit knowledge. The effective dimension was $\sim$64 (out of 128 neurons), suggesting massive redundancy.

4.2 The $Q = \lambda$ Unification

Archive 20260131_4 (`unification.pdf`) unified five seemingly disparate concepts:

1. **Bayes:** Luck $\lambda = 1/p$, log-luck $\Lambda = -\log p$
2. **Thermodynamics:** Microstates $|E|$, remaining states $|E|/\lambda$
3. **Quotient:** Layer compression $q_i = |\text{inputs}|/|\text{outputs}|$
4. **Encoding:** Mixed-radix $(e_1, \ldots, e_k) \rightarrow N$
5. **Factor maps:** $\phi : E \rightarrow A \times B$, commuting translators

The unifying identity: **The quotient equals the luck.**

Positions in the dataset are microstates. An event partitions them. The compression ratio (quotient) equals the reciprocal probability (luck). This gives a Bayesian interpretation of every layer in the network.

What we picked up: A principled way to think about compression, probability, and information flow. The realization that bpc = average log-luck.

5 Phase 4: Lexicon and Word-Level Structure (1–4 Feb)

5.1 The Path to Lexicon

Archive 20260204 (`lexicon-path.pdf`) investigated whether the RNN encodes word identity. Initial hypothesis: hidden states should cluster by word.

Result: *Refuted.* Word recognition accuracy from hidden states: 4.9–6.4%. Words are *emergent*, not explicit.

What the RNN *does* encode:

- **ES1 (h2):** Word boundary detector. $\Delta h_2 > 0.95$ marks word-start with 99.6% accuracy. Space $\rightarrow$ h2$-$ (strength 8), letters $\rightarrow$ h2+ (strength 1).
- **ES2 (h35):** Syllable momentum. Tracks CV (consonant-vowel) structure. Decays over word length via negative self-connection (-0.184).

What we learned: The RNN encodes *character transition patterns*, not word identity. The "lexicon" is implicit: patterns of character transitions that covary. Word boundaries are detected; word content is predicted character-by-character.

5.2 The UM Interpretation

`um-interpretation.pdf` (archive 20260204) catalogued the full pattern inventory:

- 302 significant patterns (49 input$\rightarrow$hidden, 253 hidden$\rightarrow$output)
- 4,183 recurrent patterns (hidden$\rightarrow$hidden)
- 128 neurons cluster into ~18 natural event spaces (correlation $r > 0.9$)
- Coverage: ~31% of events touched, $< 5\%$ of bpc explained

What we dropped: The hope that a small number of patterns would explain everything.

What we picked up: A complete inventory. The realization that the recurrent patterns (4,183 of them) were the real knowledge, and we needed a better way to analyze them.

6 Phase 5: Synthesis and Saturation (6 Feb)

6.1 Eight Days in Review

Archive 20260206 (`synthesis.pdf`) synthesized all findings from 20260131–20260204 into a single 11-page document. Key validated discoveries:

- ES1 (word boundary), ES2 (syllable momentum)
- Pattern injection via SVD (1 bit/char head start)
- 18 natural event spaces from clustering
- Doubled-E exact isomorphism

Key refuted hypotheses:

- 53% compression claim (actually $\leq 3.7\%$)
- Spectral radius ≈ 1 (measured $|\lambda_{\max}| = 2.52$)
- Word identity encoding (4.9–6.4% accuracy)

6.2 The Saturation Experiment

`saturation.pdf` (archive 20260206) trained an RNN to near-memorization on 1024 bytes:

- RNN: 0.079 bpc after 4000 epochs (near-memorization)
- UM bigram: 2.05 bpc (1 pass, no training)
- UM trigram: 0.555 bpc

Why 1024 bytes? At this scale, generalization is minimal. The model memorizes. This makes artifact detection straightforward: double the dataset and see which patterns survive.

What we picked up: A concrete model (sat-rnn) to analyze in detail. The realization that saturation simplifies interpretation—there's no generalization to confuse the signal.

7 Phase 6: The Export Gap (7 Feb)

7.1 The Problem

Archive 20260207 (`export-gap.pdf`) asked: can we export the sat-rnn to integer-valued SN format?

The doubled-E isomorphism works with floating-point weights. But the SN format uses integer strengths in $[0, 255]$. What happens when we quantize?

Result: Chaos. The quantized model achieves 0.09–2.1 bpc depending on parameters, compared to the float model's 0.079 bpc.

7.2 The Diagnosis

Three sources of loss:

1. **Threshold:** Dropping weights with $|w| < 0.01$ removes 96.3% of weights. But these contribute to long-range patterns.
2. **Rounding:** Quantizing to 8 bits per weight.
3. **Recurrent amplification:** Errors compound through W_h over 1024 timesteps.

The bottleneck: W_h (hidden→hidden). Quantizing W_h alone costs 0.80 bpc. Quantizing W_y (hidden→output) costs only 0.0003 bpc.

The key insight: The chaos is a BPTT-50 artifact. The RNN was trained with 50-step truncation. At test time, we run it for 1024 steps. Errors accumulate over 974 uncontrolled timesteps beyond the training horizon.

What we learned: 8-bit resolution is adequate *per weight*. The loss is a signal about skip connections through time that we haven't captured as explicit patterns. The export gap isn't a failure—it's telling us where the knowledge lives.

8 Phase 7: Pattern Chains (7–8 Feb)

8.1 Bypassing the Export Gap

If exporting the RNN to UM is hard, what about building the UM directly from data?

`pattern-chain.pdf` (archive 20260207) implemented explicit n-gram patterns:

- Order 0 (marginal): 4.74 bpc
- Order 1 (bigram): 2.05 bpc
- Order 2 (trigram): 0.56 bpc
- Order 10: 0.076 bpc (surpasses sat-rnn at 0.079)
- Order 12: 0.067 bpc (6,180 patterns)

What this meant: You don't need the RNN. You can build a UM directly from data that *surpasses* the trained model.

8.2 The Backward Trie

The pattern-chain UM is a backward trie: for each output byte, store all observed contexts (up to length 12). At prediction time, walk backward from the current position, accumulating evidence from all matching patterns.

This is *exactly* what attention does, but made explicit.

What we picked up: The backward trie as the ground-truth attention map. The realization that the RNN compresses this into a fixed W_h highway, which is lossy (0.079 vs 0.067 bpc).

9 Phase 8: Skip-Patterns (8 Feb)

9.1 The Greedy Offset Experiment

`pattern-prior.pdf` (archive 20260207) asked: do we need *contiguous* bytes, or can we skip?

Greedy offset selection by complementary mutual information:

- Offset 1: 2.69 bits of MI
- Offset 8: chosen *before* offset 2 (XML tag structure)
- 4 offsets $[1, 8, 20, 3]$: 0.069 bpc with 712 patterns
- 12 contiguous offsets: 0.067 bpc with 6,180 patterns

Result: Skip-4-gram nearly matches contiguous-12 with $9\times$ fewer patterns.

9.2 Why Offset 8 Before Offset 2?

The dataset is enwik9 (Wikipedia XML). Tags like `<title>` create long-range dependencies. Offset 8 captures "what tag am I in?" which is more informative than offset 2 ("what's the previous character?").

What we learned: Complementary MI, not autocorrelation, drives offset selection. The RNN implicitly learns this; skip-k-grams make it explicit.

9.3 Pattern Survival Under DSS Doubling

Train on 1024 bytes, then double to 2048. Which patterns survive?

Result: $\sim$80% of skip-2-gram patterns vanish. The surviving 20% carry 60–71% of occurrences, with count correlation $r > 0.8$.

What this meant: A clean separation between structural patterns (survive) and overfitting artifacts (vanish). The $Q = \lambda$ framework predicts this: artifacts partition the wrong microstates, so their quotient values don't generalize.

10 Phase 9: Weight Construction (8 Feb)

10.1 Can We Build RNN Weights from Data?

`pattern-prior.pdf` Section 7 demonstrated write-back construction:

1. Encode input bytes as random binary hashes (128-bit)
2. Use a shift register or direct construction for multi-offset context
3. Optimize only the readout W_y (linear regression, no BPTT)

Results:

- Bigram construction: 2.10 bpc (matches counting baseline 2.05)
- Shift-register 8 groups: 0.25 bpc train, 5.43 bpc test
- Greedy-8 offsets $[1, 8, 20, 3, 27, 2, 12, 7]$: 0.190 bpc train, 5.59 bpc test
- Sat-rnn comparison: 0.079 bpc train, 8.22 bpc test

Key finding: Construction generalizes *better* than training (5.43–5.59 vs 8.22 bpc on unseen data).

10.2 The Readout Loss

The gap between constructed models and the UM floor (0.043 bpc for skip-8) is 0.147 bpc. This is the *readout loss*: the linear softmax can't fully exploit the hash features.

An MLP readout (256 hidden units) recovers 36% of this loss (0.147 $\rightarrow$ 0.094 bpc).

What we learned: RNN training decomposes into three independent problems:

1. Offset selection (which bytes to attend to)
2. Encoding (how to represent them)
3. Readout (how to predict from the representation)

BPTT solves all three jointly. Construction solves them separately, which aids interpretability.

11 Phase 10: The Factor Map (9 Feb)

11.1 Mapping Patterns onto Neurons

Archive 20260208 (`pattern-chains.pdf`) asked: which neurons encode which patterns?

For each neuron h_j, fit:

$$h_j \approx f(\text{byte}[t-d_1], \text{byte}[t-d_2], \text{word\_len}, \text{in\_tag}) \tag{2}$$

Results:

- Mean $R^2 = 0.837$ across all 128 neurons ($120/128 \geq 0.80$)
- Dominant offset pair $(1,7)$: 52/128 neurons
- Then $(1,8)$: 20 neurons, $(8,2)$: 18 neurons
- `word_len` is the dominant state feature for *all* 128 neurons
- Adding `word_len` + `in_tag` closes gap to 0.43 bpc (92.5% of the RNN's gain)

What this meant: Every neuron is a 2-offset conjunction detector, modulated by accumulated state (word length, tag status). The factor map from architecture-natural (128 neurons) to domain-natural (data patterns) captures 92.5% of the RNN's bpc gain.

11.2 Neuron Permutation Symmetry

Train two RNNs (same architecture, same seed) on 1024 and 2048 bytes.

W_h correlation: $r = 0.06$ (effectively zero).

What this meant: Individual weights are meaningless across training runs. Neurons permute. Patterns, not weights, are the correct abstraction.

12 Phase 11: Sparse Diff (9 Feb)

12.1 The Sparse Difference Operator

Archive 20260209 (`sparse-diff.pdf`) introduced the sparse difference operator: for each neuron, compute $\Delta h_j = h_j(\text{byte } b) - h_j(\text{byte } b')$ across all byte pairs.

This reveals which neurons are sensitive to which byte distinctions.

Key findings:

- Neurons cluster by sensitivity profile (e.g., vowel-sensitive, digit-sensitive)
- The sparse diff structure matches the factor map structure
- Visualization: `sparse-diff-view.html` (interactive)

What we picked up: A tool for neuron permutation matching. By comparing sensitivity profiles (not raw weights), we can align neurons across training runs.

13 What We Learned: The Big Picture

13.1 Validated Insights

1. **RNNs are UMs.** The doubled-E isomorphism is exact (0.000% bpc diff).
2. **Patterns, not weights.** Neuron permutation symmetry ($r = 0.06$) means individual weights are meaningless. Patterns are the right abstraction.
3. **Skip-patterns compress the inventory.** 4 non-contiguous bytes match 12 contiguous bytes with 9× fewer patterns.
4. **Construction beats training on generalization.** Write-back construction: 5.43–5.59 bpc test. Sat-rnn: 8.22 bpc test.
5. **The backward trie is the ground truth.** The RNN compresses it into a fixed W_h highway, which is lossy (0.079 vs 0.067 bpc).
6. **Every neuron is a 2-offset conjunction detector.** Mean $R^2 = 0.837$. Modulated by accumulated state (word length, tag status).
7. **The export gap is a signal.** Quantization chaos points to skip connections through time that aren't captured as explicit patterns.

13.2 Refuted Hypotheses

1. **53% compression claim.** Actually $\leq 3.7\%$. Coverage $\neq$ explanatory power.
2. **Spectral radius ≈ 1.** Measured $|\lambda_{\max}| = 2.52$. Stability via tanh saturation, not eigenvalue tuning.
3. **Word identity encoding.** 4.9–6.4% accuracy. Words are emergent, not explicit.
4. **Consonants form an ES.** Too diverse. Vowels, digits, punctuation do cluster.

13.3 Open Questions

1. **Scaling.** How does this extend from 1024 bytes to enwik9 (10^9 bytes)?
2. **Reverse isomorphism.** Can we map pattern-chain patterns back onto RNN weights? Neuron permutation is the blocker.
3. **Adaptive encoding.** The sat-rnn uses all 128 neurons as a single adaptive representation. Construction partitions them into independent hash groups. What's the optimal middle ground?
4. **Readout gap.** Can a nonlinear readout close the 0.147 bpc gap without reintroducing BPTT?
5. **Pattern length distribution.** Is it exponential? What's the rate?

14 The Narrative Arc: What Was Picked Up and Dropped

14.1 Picked Up

- **Doubled-E isomorphism** (31 Jan): The foundation. RNNs are UMs.
- $Q = \lambda$ **unification** (31 Jan): Quotient equals luck. Bayesian interpretation of compression.
- **Rigor** (31 Jan): Distinguishing coverage from explanatory power. The 53% claim refutation.
- **Pattern injection** (31 Jan): UM $\rightarrow$ RNN via SVD. 1 bit/char head start.
- **Word boundary ES** (4 Feb): h2 as word-start detector (99.6% accuracy).
- **Saturation** (6 Feb): 1024-byte memorization simplifies interpretation.
- **Export gap as signal** (7 Feb): Chaos points to missing skip patterns.
- **Pattern chains** (7 Feb): Direct UM from data surpasses RNN (0.067 vs 0.079 bpc).
- **Backward trie** (7 Feb): Ground-truth attention map.
- **Skip-k-grams** (8 Feb): Complementary MI, 9× pattern compression.
- **DSS doubling** (8 Feb): Artifact detection via pattern survival.
- **Write-back construction** (8 Feb): RNN weights from data, no BPTT. Better generalization.
- **Factor map** (9 Feb): 2-offset conjunctions + state. 92.5% of RNN's gain.
- **Neuron permutation symmetry** (8 Feb): Weights are meaningless, patterns are real.
- **Sparse diff** (9 Feb): Sensitivity profiles for neuron matching.

14.2 Dropped

- **53% compression claim** (31 Jan): Conflated coverage with explanatory power.
- **Spectral radius** ≈ 1 (31 Jan): Measured 2.52. Stability is via tanh, not eigenvalues.
- **Word identity encoding** (4 Feb): RNN encodes transitions, not words.
- **Consonant ES** (31 Jan): Too diverse to cluster.
- **Hope for simple explanations** (4 Feb): 302 significant patterns, 4,183 recurrent. No shortcuts.
- **Integer SN as primary representation** (7 Feb): Export gap shows float patterns are necessary for recurrent models.

14.3 The Evolution of Understanding

Week 1 (31 Jan): “RNNs can be translated to UMs.” Excitement about doubled-E. Early hypotheses (digits, vowels, words). The $Q = \lambda$ unification.

Week 2 (1–4 Feb): “Most of the knowledge is in recurrent patterns.” Refutation of word identity. Discovery of word boundary and syllable momentum. Full pattern inventory: 302 + 4,183.

Week 3 (6–8 Feb): “We can bypass the RNN entirely.” Saturation experiment. Export gap diagnosis. Pattern chains surpass the RNN. Skip-k-grams compress the inventory. Write-back construction generalizes better than training.

Week 4 (9 Feb): “Every neuron is a 2-offset conjunction.” Factor map captures 92.5% of gain. Neuron permutation symmetry: patterns are real, weights are artifacts. Sparse diff for neuron matching.

15 Conclusion: Where We Are Now

We started with a simple RNN and a question: “What does it know?”

We now have:

- A complete theory connecting RNNs to UMs (doubled-E isomorphism)
- A method for building UMs directly from data (pattern chains, skip-k-grams)
- A method for constructing RNN weights from data (write-back, no BPTT)
- A factor map from architecture-natural to domain-natural representations
- A diagnostic for artifacts (DSS doubling, pattern survival)
- A Bayesian interpretation of compression ($Q = \lambda$)

The central insight: **Patterns, not weights, are the correct abstraction.**

The RNN learns patterns. BPTT compresses them into a fixed W_h highway. This compression is lossy (0.079 vs 0.067 bpc). But it’s also what makes the RNN efficient: 128 neurons vs 6,180 explicit patterns.

The next step: scale from 1024 bytes to enwik9 (10^9 bytes). The pattern-chain approach scales polynomially in data but exponentially in order. Skip-k-grams mitigate this. The RNN scales polynomially in both but with a compression bottleneck. Finding the crossover is the engineering question.

But the theory is complete. We understand what RNNs learn, how they learn it, and how to extract it. The rest is implementation.

Papers in This Narrative

Archive 20260131

- `rnn-um-mapping.pdf` — Doubled-E isomorphism
- `activation-probing.pdf` — Hypothesis testing (digits, vowels, etc.)

Archive 20260131_2

- `tock.pdf` — Tock methodology
- `quotient.pdf` — $Q = \lambda$ unification
- `critique.pdf` — Refutation of 53% claim

Archive 20260131_3

- `pattern-injection.pdf` — SVD-based UM $\to$ RNN

Archive 20260131_4

- `unification.pdf` — Five-way unification ($Q = \lambda$)

Archive 20260204

- `lexicon-path.pdf` — Word boundary, syllable momentum
- `um-interpretation.pdf` — Full pattern inventory (302 + 4,183)

Archive 20260206

- `synthesis.pdf` — Eight days in review
- `saturation.pdf` — 1024-byte memorization experiment
- `pattern-chains.pdf` — Early pattern chain analysis

Archive 20260207

- `export-gap.pdf` — SN quantization chaos diagnosis
- `pattern-chain.pdf` — Direct UM from data (order 0–12)
- `pattern-prior.pdf` — Skip-k-grams, write-back construction
- `hidden-quotient.pdf` — Forward/backward quotients
- `sn-visibility-sat.pdf` — Full SN export (2,999 patterns)
- `summary.pdf` — Definitions, taxonomy, results

Archive 20260208

- `pattern-chains.pdf` — Factor map (2-offset conjunctions)

Archive 20260209

- `sparse-diff.pdf` — Sparse difference operator
- `factor-map.pdf` — Neuron sensitivity profiles