Toward Total Interpretation of a Small RNN

Claude and MJC

11 February 2026

Abstract

We formalize what it means to *totally interpret* the sat-rnn: a 128-hidden-unit RNN trained to 0.079 bpc on 1024 bytes of enwik9. The doubled-E isomorphism gives an exact Universal Model (UM) with ∼3048 patterns over 768 events organized into 130 event spaces. We define a *backward attribution chain* that traces each prediction through the UM's pattern inventory, identifying which specific patterns carry the signal from input events through hidden events to the output. This gives the *sat-rnn-redux*: the subset of the isomorphic UM that the RNN actually uses. We pose the key questions whose answers constitute total interpretation.

1 Setup

Let $D = (x_0, x_1, \ldots, x_{N-1})$ be the dataset ($N = 1024$ bytes of enwik9, 52 distinct byte values).

The **sat-rnn** computes, for $t = 0, \ldots, N-1$:

$$h_t = \tanh(W_x\, e_{x_t} + W_h\, h_{t-1} + b_h) \tag{1}$$

$$P_t = \mathrm{softmax}(W_y\, h_t + b_y) \tag{2}$$

where $e_x \in \{0,1\}^{256}$ is the one-hot encoding of byte x, $W_x \in \mathbb{R}^{128\times256}$, $W_h \in \mathbb{R}^{128\times128}$, $W_y \in \mathbb{R}^{256\times128}$, $b_h \in \mathbb{R}^{128}$, $b_y \in \mathbb{R}^{256}$, and $h_{-1} = \mathbf{0}$. The model achieves 0.079 bpc after 4000 epochs of BPTT-50 training.

2 The Isomorphic UM

The doubled-E isomorphism (via $\tanh(x) = 2\sigma(2x) - 1$) gives an exact UM $u_{\mathrm{iso}} = (E, T, P, f, \omega)$:

Definition 1 (Event spaces)**.** *The event space decomposes as*

$$E = E_{\mathrm{in}} \times E_{h_0} \times \cdots \times E_{h_{127}} \times E_{\mathrm{out}}$$

where $E_{\mathrm{in}} = E_{\mathrm{out}} = \{0, \ldots, 255\}$ *(byte values) and each* $E_{h_j} = \{h_j^+, h_j^-\}$ *is a binary event space for hidden neuron* j*. Total: 768 events in 130 event spaces.*

Definition 2 (Pattern inventory)**.** *The UM patterns are derived from the weight matrices:*

- ***Input patterns*** *(W_x): For each byte* b *and neuron* j *with* $|W_x[j,b]| > \epsilon$*, a pattern "input* b*"* $\to h_j^\pm$ *with strength* $2|W_x[j,b]|$*. Sign of* $W_x[j,b]$ *determines* h_j^+ *(positive) or* h_j^- *(negative).*
- ***Recurrent patterns*** *(W_h): For each pair* (j,k) *with* $|W_h[k,j]| > \epsilon$*, a pattern* $h_j^\pm \to h_k^\pm$ *with strength* $2|W_h[k,j]|$*.*

- ***Output patterns*** *(W_y): For each neuron j and byte b with $|W_y[b,j]| > \epsilon$, a pattern $h_j^{\pm} \to$ "output b" with strength $2|W_y[b,j]|$.*
- ***Bias patterns****: b_h and b_y as always-on patterns.*

With threshold $\epsilon = 0$, there are $256 \cdot 128 + 128^2 + 128 \cdot 256 + 128 + 256 = 82{,}048$ patterns. The SN export with $\epsilon > 0$ gives $\sim$3048 patterns of significant strength.

Definition 3 (Forward pass in UM terms)**.** *At each timestep t:*

1. *Set input event: $e_{\text{in}} = x_t$.*
2. *Apply W_x patterns: input event $\to$ hidden accumulators.*
3. *Apply W_h patterns: previous hidden state $\to$ hidden accumulators.*
4. *Apply bias patterns.*
5. *Softmax within each binary ES: $P(h_j^+ \mid \text{pre}_j) = \sigma(2 \cdot \text{pre}_j)$.*
6. *Apply W_y patterns: hidden events $\to$ output accumulators.*
7. *Softmax within output ES: $P(\textit{output } b)$.*

This reproduces (1)–(2) exactly.

3 The Backward Attribution Chain

We now define the mechanism for tracing a prediction backward through the UM to identify which patterns carried the signal.

Definition 4 (Output gradient)**.** *At position t, predicting $y = x_{t+1}$, the output gradient is*

$$g_t = \frac{\partial \log P_t(y)}{\partial h_t} \in \mathbb{R}^{128} \tag{3}$$

with components

$$[g_t]_j = W_y[y,j] - \sum_{o=0}^{255} W_y[o,j]\, P_t(o)$$

This is the gradient of log-likelihood of the correct prediction with respect to the hidden state. Each component $[g_t]_j$ quantifies how much neuron j at time t contributes to predicting y.

UM interpretation. The component $[g_t]_j$ decomposes the output patterns: it is the "net vote" of neuron j for the correct output y, i.e., the strength of the $h_j \to y$ pattern minus the probability-weighted average strength of all $h_j \to o$ patterns. When $|[g_t]_j|$ is large, the W_y pattern from neuron j is a significant contributor to this prediction.

Definition 5 (Jacobian and backward gradient)**.** *The Jacobian of the hidden state update at step s is*

$$J_s = \text{diag}(1 - h_s \odot h_s) \cdot W_h \in \mathbb{R}^{128 \times 128} \tag{4}$$

where $\odot$ is elementwise multiplication. The backward gradient at offset d is

$$g_{t,d} = \Big(\prod_{s=t-d+1}^{t} J_s \Big)^{\top} g_t \qquad \textit{for } d \geq 1 \tag{5}$$

with $g_{t,0} = g_t$. The product is ordered right-to-left: $J_t^{\top} \cdot J_{t-1}^{\top} \cdots J_{t-d+1}^{\top} \cdot g_t$.

UM interpretation. The Jacobian J_s encodes how the W_h patterns propagate information one timestep. The diagonal factor $\mathrm{diag}(1-h_s^2)$ is the *saturation gate*: when $h_j(s) \approx \pm 1$ (saturated), the $(1-h_j^2)$ factor is near zero, and the gradient through neuron j is killed. This means the binary ES decision at neuron j is made with high confidence—the event h_j^+ or h_j^- is "settled" and no further information flows through it.

Conversely, when $h_j(s)$ is far from saturation, the ES decision is uncertain, and the W_h patterns through neuron j are active conduits for information flow.

Definition 6 (Input attribution)**.** *The scalar attribution of the input at position* $t-d$ *is*

$$\alpha(t,d) = (W_x\, e_{x_{t-d}})^\top g_{t,d} = \sum_{j=0}^{127} W_x[j, x_{t-d}] \cdot [g_{t,d}]_j \tag{6}$$

The per-neuron attribution is

$$\alpha_j(t,d) = W_x[j, x_{t-d}] \cdot [g_{t,d}]_j \tag{7}$$

so that $\alpha(t,d) = \sum_j \alpha_j(t,d)$*.*

UM interpretation. $\alpha_j(t,d)$ is the product of two quantities:

- $W_x[j, x_{t-d}]$: the strength of the input pattern "input x_{t-d}" $\to h_j^\pm$ (with sign).
- $[g_{t,d}]_j$: the backward gradient at neuron j after d steps, encoding how much neuron j at time $t-d$ matters for the prediction at time $t+1$.

Their product is the end-to-end contribution of the specific W_x pattern at position $t-d$ through the specific chain of W_h patterns across d timesteps to the specific W_y pattern at the output. A large $|\alpha_j(t,d)|$ means neuron j carried a significant signal from the input at $t-d$ to the output at $t+1$.

4 Pattern Chains Through Hidden Events

Definition 7 (Attribution chain)**.** *An* attribution chain *for the prediction at position t is a sequence of events:*

$$C = \big(\underbrace{e_{\text{in}}^{(t-d)}}_{input},\ \underbrace{e_{h_{j_0}}^{(t-d)}}_{hidden},\ \underbrace{e_{h_{j_1}}^{(t-d+1)}}_{hidden},\ \ldots,\ \underbrace{e_{h_{j_d}}^{(t)}}_{hidden},\ \underbrace{e_{\text{out}}^{(t+1)}}_{output}\big)$$

where each consecutive pair is connected by a UM pattern:

- $e_{\text{in}}^{(t-d)} \to e_{h_{j_0}}^{(t-d)}$ *via a* W_x *pattern*
- $e_{h_{j_s}}^{(t-d+s)} \to e_{h_{j_{s+1}}}^{(t-d+s+1)}$ *via* W_h *patterns*
- $e_{h_{j_d}}^{(t)} \to e_{\text{out}}^{(t+1)}$ *via a* W_y *pattern*

The chain has length $d+2$ *(one input,* $d+1$ *hidden events, one output). Each hidden event* $e_{h_j}^{(s)}$ *is either* h_j^+ *or* h_j^-*, determined by the sign of* $h_j(s)$ *in the actual forward pass.*

In general, the backward gradient at each step involves all 128 neurons simultaneously (not a single neuron path). The full picture is a *weighted sum over all possible chains*, where the weight of each chain is the product of the relevant entries of J_s and g_t.

Proposition 1 (Additive decomposition)**.** *The input attribution decomposes additively over neurons at each step:*

$$\alpha(t,d) = \sum_{j=0}^{127} \alpha_j(t,d) = \sum_{j=0}^{127} W_x[j, x_{t-d}] \cdot [g_{t,d}]_j$$

Each term $\alpha_j(t,d)$ is the contribution of the chain that enters *through neuron j at time $t-d$. But $[g_{t,d}]_j$ itself is a sum over all paths from neuron j at $t-d$ to all neurons at t, weighted by the W_h entries and saturation gates along each path.*

Remark. A single-neuron chain $j_0 \to j_1 \to \cdots \to j_d$ has weight

$$w(j_0,\ldots,j_d) = W_x[j_0, x_{t-d}] \cdot \prod_{s=0}^{d-1} (1 - h_{j_s}(t{-}d{+}s)^2)\, W_h[j_{s+1}, j_s] \cdot [g_t]_{j_d}$$

The full attribution $\alpha(t,d)$ is the sum over all such chains:

$$\alpha(t,d) = \sum_{j_0,\ldots,j_d} w(j_0,\ldots,j_d)$$

There are 128^{d+1} such chains, but most have negligible weight. Sparsity arises from:

1. **Saturation gates**: $(1 - h_j^2) \approx 0$ when h_j is saturated, killing chains that pass through settled neurons.
2. **Weight sparsity**: many $W_h[k,j]$ entries are small.
3. **Gradient sparsity**: few neurons have large $|[g_t]_j|$.

5 The Key Questions

We now have the machinery to pose the questions whose answers constitute total interpretation of the sat-rnn.

Question 1 (Which patterns are active?)**.** *For each prediction at position t, how many of the* $\sim$3048 *SN patterns participate in the backward attribution chain with weight above a given threshold? That is: how sparse is the explanation?*

Question 2 (Which offsets carry the signal?)**.** *What is the distribution of $|\alpha(t,d)|$ over offsets $d = 1,\ldots,50$? The greedy skip-k-gram analysis (from data alone, without the RNN) found offsets $[1, 8, 20, 3, 27, 2, 12, 7]$ as the MI-optimal set. Does the RNN's actual attribution profile match this ordering? If not, what offsets does the RNN actually use, and why?*

There is no guarantee the RNN's offset usage matches the data-derived greedy ordering. The greedy offsets were derived from a reasonable prior over the learnable pattern space, but the RNN's training dynamics (BPTT-50, gradient descent, 128-neuron bottleneck) may have converged to a different selection. If the match is poor, it means the skip-k-gram superset does not contain the pattern chains we need to explain the RNN's performance.

Question 3 (Which neurons carry the signal?)**.** *At each step of the backward chain, how many neurons have significant gradient? The factor map (factor-map.pdf) showed each neuron is a 2-offset conjunction detector with a dominant pair from $\{(1,7),(1,8),(8,2),(2,12)\}$. Does the backward trace confirm this? That is: when the gradient reaches offset d, is it concentrated in neurons whose dominant pair includes d?*

Question 4 (What is the saturation structure?). *The saturation gate* $(1 - h_j^2)$ *determines which neurons are active conduits at each timestep. What fraction of neurons are saturated (gate* $< \epsilon$*) at each position? This partitions the hidden state into:*

- Settled *neurons: binary state committed, serving as persistent features ("in an XML tag", "word length* ≥ 3*").*
- Active *neurons: gate open, carrying information from recent inputs toward the output.*

This partition is the bridge between the factor map's static picture (neurons as conjunction detectors) and the sparse differentiation's dynamic picture (information flowing through specific chains).

Question 5 (What is the sat-rnn-redux?). *Define the* sat-rnn-redux *as the subset of* u_{iso}*'s patterns that appear in backward attribution chains above a threshold. Concretely: a pattern* $p \in P$ *is in the redux if there exists a position* t *and offset* d *such that* p *participates in the backward chain at* (t, d) *with weight above* τ*.*

1. *How many of the* $\sim$3048 *significant SN patterns are in the redux? (Equivalently: how many patterns does the RNN actually use?)*
2. *What bpc does the redux achieve when run as a UM?*
3. *How does the redux bpc depend on the threshold* τ*?*
4. *Does the redux match the data-derived superset? That is: for each pattern in the redux, is there a corresponding skip-k-gram pattern that explains the same prediction?*

Question 6 (Can we justify each prediction?). *For a single prediction at position* t*, can we produce a human-readable justification of the form:*

> *At position* $t = 42$*, the model predicts 'm' with probability* 0.98.
> *The dominant chain: the input '/' at* $t-4$ *activates neuron* h_7^+ *via* W_x *pattern (strength 3.2).* h_7^+ *persists via* W_h *self-connection (strength 2.8, gate 0.4) through positions* 38–42. *At* $t = 42$*,* h_7^+ *contributes to 'm' via* W_y *pattern (strength 1.9).*
> *Secondary chain: input 'p' at* $t-1$ *activates* h_{56}^+ *(strength 2.1), which directly contributes to 'm' via* W_y *(strength 1.4).*
> *Together, these two chains account for* 87% *of the total attribution.*

The justification mentions only events from the UM—input events, hidden events (h_j^+/h_j^-*), and output events—connected by UM patterns with explicit strengths. This is the sense in which the interpretation is* total*: every element of the explanation is a UM event or pattern, and the strengths are the actual weight magnitudes.*

6 Experimental Plan

To answer these questions, we need the following:

1. **Full backward trace** (extending sparse_diff.c): For each position t, compute $g_{t,d}$ for $d = 0, \ldots, 49$ (the full BPTT-50 horizon). Record the per-neuron backward gradient $[g_{t,d}]_j$ and the per-neuron input attribution $\alpha_j(t, d)$ at each offset.

2. **Saturation census**: For each position t and neuron j, record $(1 - h_j(t)^2)$. Partition neurons into settled/active at each position. Compute the time-averaged saturation profile.

3. **Chain enumeration**: For the top-K predictions (highest bpc), enumerate the dominant single-neuron chains. Verify that they trace to interpretable input events.

4. **Redux construction**: Threshold on per-pattern attribution to define the redux subset. Export as SN. Evaluate bpc.

5. **Offset comparison**: Compare the RNN's per-offset attribution $|\alpha(t, d)|$ averaged over positions to the MI-greedy ordering $[1, 8, 20, 3, 27, 2, 12, 7]$.

6. **Per-prediction justification**: For selected positions, produce the human-readable justification described in Question 6.

Notation Summary

Symbol	Meaning
$h_t \in \mathbb{R}^{128}$	Hidden state at time t
$g_t \in \mathbb{R}^{128}$	Output gradient $\partial \log P_t(y)/\partial h_t$
$J_s \in \mathbb{R}^{128\times128}$	Jacobian at step s: $\mathrm{diag}(1 - h_s^2) \cdot W_h$
$g_{t,d} \in \mathbb{R}^{128}$	Backward gradient at offset d
$\alpha(t, d) \in \mathbb{R}$	Scalar input attribution at offset d
$\alpha_j(t, d) \in \mathbb{R}$	Per-neuron input attribution
$E_{h_j} = \{h_j^+, h_j^-\}$	Binary ES for neuron j
$(1 - h_j^2)$	Saturation gate for neuron j

Reproducibility

This paper builds on:

- sat-rnn: `sat_model.bin` in `archive/20260209/`
- Data: `enwik_1024.txt` (first 1024 bytes of enwik9)
- Prior tools: `sparse_diff.c`, `factor_map2-4.c`
- Prior papers: export-gap.pdf, pattern-chain.pdf, factor-map.pdf, sparse-diff.pdf