Path to Lexicon:
From Character Visibility to Word Events

Claude

2026-02-04

1 Current State

We have character-level visibility into an Elman RNN (256→128→256) trained on enwik9:

- 768 named events (256 input + 256 hidden + 256 output)
- 302 significant patterns (character bigrams)
- Isomorphic UM matches RNN exactly (0.00% bpc difference)
- Hidden layer is uninterpreted

The model achieves approximately 5.7 bpc. We have the doubled-E UM that exactly matches the RNN. The task is analysis-in-situ: finding the natural event spaces and patterns hidden in this isomorphic representation.

2 Goal: The Lexicon Milestone

The milestone is discrete and defined as:

1. Coverage of ∼100 common English words
2. Discovery of the encoding pattern (how words live in hidden layer)
3. Lift procedure: isomorphic UM → interpretable UM
4. Injection procedure: write new words without losing spelling signal

After this milestone, remaining words can be derived by scanning the input corpus and writing them directly into model weights.

Note: The interpretable model will be **less accurate** than the isomorphic one. This is expected—we are simplifying and factoring, which loses fidelity. However, interpretability and compression efficiency are the same problem: the tick-tock cycle uses understanding from interpretation (tock) to improve performance in the next training round (tick).

3 Method: Analysis-in-Situ

From CMP paper (3.2224): analysis-in-situ is "the derivation of the event spaces and patterns of a model from a trained ANN by finding, through machine or human intelligence, the events, event spaces, and product patterns, and the correct names for them, generally in an iterative process beginning from the outside of the model and working in."

The search for correct event spaces is also the search for correct product patterns with their contexts (3.22242). We work on both simultaneously.

3.1 Sparse Differentiation

From CMP paper (2.121): sparse differentiation is "the same process used in backpropagation in ANNs, except that we throw away as much of the calculation as we can, while maintaining the required explanation gain."

For words, we work **backward from word ends**:

1. Find positions where a word ends (e.g., "and" followed by space)
2. Trace back through the UM: which hidden activations led to predicting 'd' after 'n' after 'a'?
3. The minimal set of patterns that explains the prediction is the word pattern
4. Collect these patterns for many words

3.2 Logic Helps Identify ESs

We know what English words are—we can enumerate them. This is external knowledge we bring to interpretation, just as a chess interpreter knows what "checkmate" means.

From CMP paper (3.22243): "Searching for the correct event spaces is searching for a partition of each layer such that each subset is one-hot in typical activations of the network."

For words, the logic is:

- Words are mutually exclusive at word boundaries (exactly one word ends here)
- This defines an event space: {"and", "for", "the", ..., other}
- Hidden neurons that fire exclusively for different words are in the same ES

We can use our knowledge of English to *propose* event spaces, then verify by checking if the partition is one-hot in typical activations.

3.3 Variable Patterns for Words

From CMP paper (3.313): abstraction requires "variable patterns"—a context event plus an event space that functions like a register.

For chess: "the square I am looking at" + 64-event space for which square.

For words: "the word being recognized" + event space for which word. This allows one set of patterns to handle all words, rather than separate patterns for each.

The hidden layer likely implements something like this—a compositional encoding where context accumulates character-by-character until word recognition fires.

4 Compositional Encoding

The encoding is compositional: hidden state builds up as characters arrive. After "a", then "an", then "and", the state encodes progressively more specific predictions.

In UM terms: each character input fires patterns that update hidden events. The accumulated support in hidden events represents partial word recognition. When a word-end is reached (space/punctuation), the hidden state "commits" to a word prediction.

Question: What is the exact structure of this composition in the isomorphic UM? How do the 128 hidden neurons (256 with doubled-E) encode word identity?

5 XML and English Share Lexicon

The dataset (enwik9) is Wikipedia XML. Tag names are mostly English words: `<title>`, `<text>`, `<page>`, etc.

This is a "mixed" context—the same word-detection machinery serves both English prose and XML tag names. We should not expect separate "modes."

6 Probing to Test Hypotheses

Once we have candidate patterns and ESs, use probing to test:

- Hypothesis: "neurons 47 and 103 form an ES for word-initial vs word-internal"
- Test: check if they are mutually exclusive in typical activations
- Hypothesis: "after 'an', neuron 82 has high activation iff next char is 'd' "
- Test: probe with `ctx=an` and check correlation with next char

7 Experimental Results

7.1 Event Space 1: Word Boundary

The most robust Event Space discovered is the word boundary detector:

- $\Delta h_2 > 0.95$ marks word-start with 99.6% accuracy
- At word-start: $\Delta h_2 = +1.87 \pm 0.11$
- At mid-word: $\Delta h_2 = +0.03 \pm 0.19$
- Supporting neurons: h45 (-0.62), h120 (-0.55), h61 (+0.50), h119 (+0.49)

This is a near-perfect binary Event Space: word-start vs non-word-start.

7.2 Event Space 2: Position Markers

Position within word is encoded:

- Position 0 (first char): h_2 jumps +1.83, h_{35} stays at -0.05
- Position 1 (second char): h_2 stable, h_{35} jumps to +0.42

- Position 2+: Both stabilize, word-specific patterns emerge

The neuron h_{35} marks second-char position distinctly.

7.3 Event Space 3: Final Letter Class

At word ending, Δh_{35} partitions by final letter type:

- Final 'e' words (the, are): $\Delta h_{35} \approx +0.12$
- Final 's' words (was, his): $\Delta h_{35} \approx -0.25$
- Final 'd' words (and, had): $\Delta h_{35} \approx -0.28$
- Final 't' words (not, but): $\Delta h_{35} \approx -0.19$

7.4 Key Finding: No Explicit Word Identity

The model does NOT encode word identity directly in hidden states.

Recognition experiments using hidden state matching:

- Delta-based (2 neurons): 6.4% accuracy
- Delta-based (128 neurons): 4.9% accuracy
- Absolute end-state: 4.9% accuracy

Explanation: A character-level model doesn't need word identity. It encodes $P(\text{next\_char}|\text{context})$, not "this is word X". Words are emergent, not explicitly represented.

The "lexicon" in the UM is not a lookup table—it's patterns of character transitions that covary.

7.5 Key Finding: h35 Encodes CV Syllable Structure

Further analysis revealed that h_{35} encodes **consonant-vowel (CV) syllable structure**, not word identity.

Words with the same CV pattern have nearly identical h_{35} trajectories:

- CCV words (the, she): avg distance = 0.099
- CVC words (for, was, his, but, not, can, had, her, way, may, day): avg distance = 0.106
- VCV words (are, one, use): avg distance = 0.097

Word identification from h_{35} trajectory alone achieves only 3.1% accuracy.

Implication: The model has learned English phonotactics implicitly. h_{35} tracks syllable structure (CV alternation pattern), not lexical identity.

Example trajectories (CVC pattern):

```
'for': [-0.63, +0.54, +0.35]
'was': [-0.64, +0.56, +0.28]
'can': [-0.58, +0.61, +0.33]
```

All CVC words follow the same pattern: low at word start, high after C→V transition, moderate after V→C transition.

7.6 100-Word Analysis

Analyzed 100 most common words using discovered Event Spaces:

- ES1 works for English words ($\Delta h_2 \approx 1.8$)
- ES1 fails for XML entities (quot, lt, gt) which lack word boundaries
- Words cluster by signature: final_char / sign(Δh_{35}) / sign(Δh_{111})

8 Revised Understanding

8.1 What the Model Encodes

The model encodes:

1. Word boundaries (ES1: $\Delta h_2 > 0.95$)
2. CV syllable structure (h_{35} tracks consonant-vowel alternation)
3. Character transition probabilities

The model does NOT encode:

1. Word identity directly
2. Explicit word lookup
3. Individual word patterns (same-CV words are indistinguishable)

Key insight: The model has learned English phonotactics implicitly. The "lexicon" is not a lookup table—it is the combination of:

- Word boundary detection (ES1)
- Syllable structure (CV patterns via h_{35})
- Character-level predictions

8.2 Implications for Lift Procedure

The lift from isomorphic to interpretable UM should:

1. Use ES1 for word boundary segmentation
2. Track character-by-character patterns within words
3. Understand "word" as the covariation of character transitions
4. Not attempt direct word identity matching

9 Open Questions

1. How to represent word patterns as covarying character transitions?
2. What is the injection procedure for adding new "words"?
3. How does ES1 relate to the spelling/lexicon signal?

10 Next Steps

1. Analyze character transition patterns (bigrams, trigrams)
2. Find Event Spaces for character classes, not word classes
3. Develop lift procedure based on character covariation
4. Test injection on simple cases