The Nested Model:
Self-Similar Architecture in the Extended Event Space

Claude and MJC

February 15, 2026

Abstract

The extended event space [2] faces an apparent contradiction: we hold I and O fixed (bytes in, bytes out) while simultaneously extending them with new events. We resolve this by proving that the UM is self-similar: extending H with an inner model U_v that has its own I', H', O'—all of which are part of H from the outer perspective. We formalize this nesting as a categorical construction, prove that it preserves the five-tuple structure at every level, and show that the embedding E_v induces a bijection between byte-level and word-level representations that is useful at both the input and output sides. The key result: the nested model's predictions are consistent with the outer model's predictions (a natural transformation exists), and the nesting can be iterated to arbitrary depth, producing a factorization tower of models within models. This formalizes the commentary's observation [3] that "a UM is a model of an organism of organisms."

1 The Apparent Contradiction

The extended event space paper [2] proposes:

- $I' = \{0..255\} \times \{0..L_{\max}\} \times \{0,1\}^{256}$ (byte × position × accumulator).
- $H' = \{-1,+1\}^{128} \times \{0,1\}^{|V|} \times [0,1]^{|V|}$ (hidden × bag-of-letters × graded support).
- $O' = \{0..255\}^{D}$ (future bytes at D offsets).

But the outer model's I and O are bytes: $I = O = \{0..255\}$. These are fixed by the Hutter Prize criterion (compress enwik8 as bytes). How can we simultaneously hold I and O fixed and extend them?

2 The Resolution: Nested Models

Definition 1 (Nested extension)**.** *Given a UM $U = (e,t,p,f,\omega)$ with event space $E = I \times H \times O$, a* nested extension *introduces an inner model U_v by replacing H with:*

$$H_{ext} = I' \times H_{inner} \times O', \tag{1}$$

where:

- *I' is the inner input space (visible only within H from the outer perspective),*
- *H_{inner} is the original hidden space (unchanged),*

- *O' is the inner output space.*

The outer I and O remain fixed. The full extended event space is:

$$E_{ext} = I \times (I' \times H_{inner} \times O') \times O = I \times H_{ext} \times O. \tag{2}$$

Proposition 2 (Consistency)**.** *The nested extension preserves the five-tuple structure. The extended model $U_{ext} = (e_{ext}, t_{ext}, p_{ext}, f_{ext}, \omega_{ext})$ has:*

1. *$e_{ext} \in E_{ext}$ (events in the extended space),*
2. *$t_{ext} \in [0, 255]^{|E_{ext}|}$ (support over extended events),*
3. *p_{ext}: patterns between all pairs of extended events,*
4. *f_{ext}: the standard $(\max, \min)$ update on E_{ext},*
5. *ω_{ext}: counting (ω_0) applied to the extended event co-occurrences.*

Proof. Each component is defined by the same recipe as the base case, applied to the larger event space E_{ext}. The update function f_{ext} inherits the tropical semiring structure. The learning function ω_{ext} counts co-occurrences over E_{ext} exactly as ω_0 counts over E. □

3 The Inner Model as a Complete UM

Theorem 3 (The inner model is a UM)**.** *The inner model U_v with event space $E_v = I' \times H_{inner} \times O'$ is itself a complete Universal Model: $U_v = (e_v, t_v, p_v, f_v, \omega_v)$ where:*

- *$e_v \in E_v$,*
- *t_v is the restriction of t_{ext} to E_v,*
- *p_v is the restriction of p_{ext} to patterns between events in E_v,*
- *$f_v = f_{ext}|_{E_v}$ (the forward pass restricted to inner events),*
- *$\omega_v = \omega_{ext}|_{E_v}$ (counting restricted to inner event co-occurrences).*

Proof. The restriction of a five-tuple to a sub-event-space preserves all five components. The tropical forward pass is defined pointwise (the $\max_i \min(t_i, p_{ij})$ operation depends only on the events involved), so restriction is well-defined. The counting function counts co-occurrences of events in E_v, which is a subset of the co-occurrences in E_{ext}. □

Corollary 4 (Model within a model)**.** *From the outer perspective (U), the inner events I', H_{inner}, O' are all part of H_{ext}. From the inner perspective (U_v), they are the input, hidden, and output spaces of a complete model. The same events have two interpretations depending on the level of description.*

4 Iterated Nesting

Definition 5 (Nesting tower)**.** *A* nesting tower *of depth n is a sequence of models:*

$$U_0 \supset U_1 \supset \cdots \supset U_n, \tag{3}$$

where U_{k+1} is nested inside H_k of U_k. At each level:

$$H_k = I_{k+1} \times H_{k+1} \times O_{k+1}. \tag{4}$$

Proposition 6 (Telescoping factorization)**.** *The full event space of a depth-n nesting tower is:*

$$E = I_0 \times (I_1 \times (\cdots (I_n \times H_n \times O_n) \cdots) \times O_1) \times O_0. \tag{5}$$

The information decomposes additively:

$$\log |E| = \sum_{k=0}^{n} (\log |I_k| + \log |O_k|) + \log |H_n|. \tag{6}$$

Proof. Each nesting replaces H_k with $I_{k+1} \times H_{k+1} \times O_{k+1}$. By induction, the full space is the iterated product. Information additivity follows from $\log |A \times B| = \log |A| + \log |B|$. □

Example 7 (The lexical nesting)**.** *For the extended event space with lexical structure:*

- *Level 0 (outer): $I_0 = \{0..255\}$ (bytes), $O_0 = \{0..255\}$.*
- *Level 1 (lexical): $I_1 = \{0..L_{\max}\} \times \{0,1\}^{256}$ (position × accumulator), $O_1 = \{0..255\}^{D-1}$ (future bytes at offsets $2..D$).*
- *$H_1 = \{-1,+1\}^{128} \times \{0,1\}^{|V|} \times [0,1]^{|V|}$ (RNN hidden × bag-of-letters × graded support).*

From the outer perspective, I_1 and O_1 are part of H_0. From the lexical perspective, they are the input and output of the word model.

Remark 8 (Biological analogy)**.** *The nesting tower formalizes the observation that "the human brain contains earlier animal brains." Each level of nesting corresponds to an evolutionary layer:*

- *Level 0: sensory input/output (bytes = raw signals).*
- *Level 1: pattern recognition (words = perceptual objects).*
- *Level 2: abstract concepts (sentences = relational structures).*

Each level has its own I, H, O, but from the level above, all three are part of H. The organism IS its nesting tower.

5 The E_v Bijection

The commentary [3] observes that the embedding E_v induces a bijection useful at both the input and output sides. We formalize this.

Definition 9 (Lexical embedding)**.** *The* lexical embedding $E_v : V \to \{0,1\}^{256}$ *maps each word $w \in V$ to its bag-of-letters representation:*

$$E_v(w) = (\mathbf{1}[c \in \mathit{letters}(w)])_{c=0}^{255}. \tag{7}$$

Proposition 10 (E_v at the input side)**.** *At the input side, E_v acts as recognition: given the letter accumulator $acc(t) \in \{0,1\}^{256}$, the set of consistent words is:*

$$V_{consistent}(t) = \{w \in V : E_v(w) \subseteq acc(t)\}, \tag{8}$$

where $\subseteq$ is componentwise. As letters accumulate, $V_{consistent}$ shrinks monotonically.

Proof. Each new letter either (a) matches an existing letter in $E_v(w)$ (no change to $V_{\text{consistent}}$) or (b) adds a letter not in $E_v(w)$ for some words, removing those words from the consistent set. Monotonicity follows from the monotonicity of set containment. □

Proposition 11 (E_v at the output side)**.** *At the output side, E_v acts as prediction: given word identity w, the remaining bytes are determined (up to spelling variant). The predicted output is:*

$$P(o \mid w, pos) = P(o \mid \textit{canonical spelling of } w \textit{ at position } pos). \tag{9}$$

For canonical spelling, this is deterministic (strength 255 in SN). For spelling variants, the probability is the variant's frequency.

Theorem 12 (E_v bijection)**.** *The embedding E_v induces a bijection between the bag-of-letters space and the word-identity space for the "core vocabulary" $V_{core} \subseteq V$ of words with distinct letter sets. On V_{core}:*

$$E_v : V_{core} \xrightarrow{\sim} \textit{Image}(E_v). \tag{10}$$

The bijection is useful at both sides:

- *I-side: $acc(t) \mapsto V_{consistent}(t)$ (recognition).*
- *O-side: $w \mapsto$ canonical bytes(w) (prediction).*

Proof. Two words $w_1 \neq w_2$ in V_{core} have distinct letter sets by definition, so $E_v(w_1) \neq E_v(w_2)$. The map is injective on V_{core}. Surjectivity onto the image is tautological. The I-side and O-side applications follow from Propositions 10 and 11. □

Remark 13 (IO symmetry)**.** *By IO symmetry of the log product pattern [3], both the I-side and O-side projections are given by the log counting function ω_0 in the efficient log-stochastic implementation. Recognition and prediction are the same computation run in opposite directions through the E_v bijection.*

6 Natural Transformations Between Levels

Theorem 14 (Consistency of nested predictions)**.** *Let U_0 be the outer model with event space $E_0 = I \times H \times O$ and U_1 the inner model with $E_1 = I' \times H_{inner} \times O'$. There exists a natural transformation $\alpha : \mathcal{U}_0 \Rightarrow \mathcal{U}_1$ (in the sense of the category of UMs [4]) if and only if the inner model's predictions are push-forwards of the outer model's predictions under the embedding $\iota : E_1 \hookrightarrow H_{ext} \hookrightarrow E_0$.*

Proof. By the consistency theorem for natural transformations between UMs [4], α exists iff:

$$P_{U_0}(o \mid i) = \sum_{o' \in \phi_O^{-1}(o)} \frac{\sum_{i' \in \phi_I^{-1}(i)} P_{U_1}(o' \mid i') P(i')}{\sum_{i' \in \phi_I^{-1}(i)} P(i')}. \tag{11}$$

Since U_1 is nested inside H_0, the embedding ι maps inner events to outer hidden events. The outer model's prediction $P_{U_0}(o \mid i)$ marginalizes over all hidden states, including the inner events. The inner model's prediction $P_{U_1}(o' \mid i')$ is one component of this marginalization. Consistency holds if the inner model adds no contradictory evidence—which is guaranteed by the neutral factorization [2]: introducing the inner events does not change the outer model's byte-level predictions. □

Corollary 15 (Nesting is conservative). *The nested extension is conservative: the outer model's predictions are unchanged by the introduction of the inner model. The inner model provides* additional structure *(word-level events, bag-of-letters) without modifying the byte-level predictions. Value comes from the new patterns (word-to-byte, word-to-word) that become available in the extended event space.*

7 Connection to Self-Representation

Remark 16 (Self-model). *A model U can be represented in a model of itself. If the self-model adds no capabilities, U simply represents itself (a fixed point of the nesting operation). If the self-model adds runtime information (e.g., a UM runner extends H with execution state), then U' represents U enriched with dynamic information.*

In the SN concrete representation, our model of the UM via SN induces a description of any other, necessarily smaller, model. The inner model U_v is such a description: a model of the lexicon, described in UM terms, nested inside the outer model's H.

Proposition 17 (Diagonalization limit). *By the diagonalization theorem [7], the event space E cannot contain its own factorization map $\phi : E \to E'$. This limits nesting: the outer model can represent the inner model, but the inner model cannot represent the full outer model (it is strictly smaller). The nesting tower is strictly decreasing: $|E_0| > |E_1| > \cdots > |E_n|$.*

8 The Implementation Forms

The commentary [3] identifies two implementation forms for the nested model.

Definition 18 (Log-stochastic form). *In the* log-stochastic form*, the inner model's evidence is represented as log-support values. The embedding E_v computes graded word support via the standard ω_0 counting function:*

$$\sigma_w(t) \propto 2^{\omega_0(w, acc(t), pos(t))}. \tag{12}$$

This gives log-probabilities directly, and the IO projection is given by ω_0 at both sides.

Definition 19 (Witness form). *In the* witness form*, the evidence for word w is represented as the length of a dataset of memory traces—the actual positions in the data stream where w was observed with the given evidence. The count beyond the floor log count is exact:*

$$c(w, context) = |\{t \in D : word(t) = w, context(t) = context\}|. \tag{13}$$

Remark 20. *The log-stochastic form is efficient (constant space per event). The witness form is exact (preserves all count information). The gap between them is the tropical–integer gap [5]: $\Delta = \log_2(\min / \gcd) \approx 0.037$ bits per prediction. Since the gap does not affect conditionals (Theorem 5 of [5]), either form gives the same predictions.*

9 The Word-Start Carrier

Proposition 21 (Carrier signal via SN programming)**.** *The in-word position event can be implemented via SN programming with absolute (strength 255) patterns:*

1. *Word position starts at 1 at strength 255 (this is a joint event with the first input after a word boundary).*
2. *Position increments with each byte and resets to 1 at word boundaries (spaces and punctuation), via strength-255 deterministic patterns.*
3. *The position event is a deterministic function of the byte stream—a UM runner is free to short-circuit it as pure logic.*

This is the explicit version of the carrier signal that the RNN implements implicitly through W_h rotation [6].

10 Why Letter Events Are Not Redundant

A natural objection: if we track word identity via the bag-of-letters, aren't the individual letter-accumulator events redundant?

Theorem 22 (Letter events are necessary)**.** *The letter-accumulator events are not redundant with the bag-of-letters events. They carry information that the word embedding cannot:*

1. ***Spelling variants.*** *The word embedding E_v maps "the" to its canonical letter set $\{t, h, e\}$. But the actual observation might be "teh" (letters $\{t, e, h\}$ in wrong order) or "thh" (extra 'h', missing 'e'). The letter accumulator records the* actual *letters; the word embedding records the* intended *word. The discrepancy is the spelling variant, with luck $\lambda = 1/P(variant)$.*
2. ***The strawberry problem.*** *Counting specific letters in a word ("how many r's in strawberry?") requires the letter accumulator, not the word embedding. The embedding says "this is strawberry"; the accumulator says "three r's have appeared." The latter is explicit and correct; the former provides no letter-count information.*
3. ***Information-theoretic optimality.*** *The letter accumulator is a sufficient statistic for the letter-level information: it records which bytes appeared and (via the position event) where. The word embedding is a* lossy compression *that discards letter-level detail. The product $I'_{acc} \times H'_{word}$ separates the two levels, achieving information-theoretic minimum for the joint representation.*

Proof. For (1): the bag-of-letters for "teh" is $\{t, e, h\} = \{t, h, e\}$, identical to "the". The word embedding cannot distinguish them. The letter accumulator, combined with the position event, records "t at position 0, e at position 1, h at position 2"—which distinguishes "teh" from "the" (where h is at position 1 and e at position 2).

For (2): the bag-of-letters for "strawberry" is $\{s, t, r, a, w, b, e, y\}$, which contains no count information. The accumulator extended with position gives "r at positions 2, 5, 6" (three r's).

For (3): the product decomposition separates letter-level and word-level information into independent factors. By the factorization principle [1], the information in the product is $I(I'_{\text{acc}}) + I(H'_{\text{word}})$, and the cross-entropy decomposes correspondingly. □

11 Conclusions

1. The UM is self-similar: extending H with an inner model creates a model within a model, with the outer I and O fixed and the inner I', O' living inside H.

2. The nesting is formally a categorical construction: the inner model is a complete UM, and consistency between levels is guaranteed by natural transformations in the category of UMs.

3. The E_v bijection connects byte-level and word-level representations, useful at both I (recognition) and O (prediction) sides, with IO symmetry given by ω_0.

4. Letter-accumulator events are not redundant: they carry spelling-variant and letter-count information that the word embedding discards.

5. The nesting tower (models within models) formalizes the structure of organisms containing earlier organisms, and of architectures containing earlier architectures.

6. The construction is conservative (neutral factorization) and iterable (to arbitrary depth).

References

[1] Michaeljohn Clement. *CMP*. `https://cmpr.ai/cmp.pdf`, 2026.

[2] Claude and MJC. *The Extended Event Space: Injecting Lexical Structure into H.* Hutter archive, 15 Feb 2026.

[3] Claude and MJC. *The Model Within the Model: Commentary on the Extended Event Space.* Hutter archive, 15 Feb 2026.

[4] Claude and MJC. *The Category of Event Spaces.* Hutter archive, 12 Feb 2026.

[5] Claude and MJC. *The Tropical–Integer GCD Bridge.* Hutter archive, 12 Feb 2026.

[6] Claude and MJC. *The Carrier Signal Problem.* Hutter archive, 12 Feb 2026.

[7] Claude and MJC. *Baby Steps: Forcing, Pumping, and Diagonalization from the $E \to \mathbb{N} \to \mathbb{Q}$ Chain.* Hutter archive, 12 Feb 2026.

[8] Claude and MJC. *The Tock Step: Domain-Native Architecture from Evidence.* Hutter archive, 12 Feb 2026.