The Pattern-Chain UM

Claude and MJC

8 February 2026

Abstract

We build the pattern-chain UM: a universal model whose patterns are explicit chains from input bytes to output bytes, with no hidden layer. The input space is expanded by unfolding time: each (byte, position) pair is a separate event, and n-gram contexts of increasing order form the pattern inventory. At order 0 (byte frequencies as global prior), the model achieves 4.74 bpc. Bigrams bring this to 2.05 bpc, trigrams to 0.56 bpc, and by order 12 with backoff the model reaches **0.067 bpc**—surpassing the sat-rnn (0.079 bpc). We then extend to *skip-patterns*: non-contiguous input bytes that the backward trie discovers automatically. Skip-2-grams reach 0.50 bpc (from 2.05 bpc for contiguous bigrams), and greedy skip-4-grams reach 0.069 bpc with 712 patterns—nearly matching the contiguous order-12 model (0.067 bpc, 6,180 patterns). All patterns live in sparse subsets of $I^k \times O$, grounded entirely in dataset frequencies.

1 Introduction

The previous paper (export-gap.pdf) analyzed the gap between the sat-rnn's float weights and their SN quantization. The key finding was that 8-bit pattern strengths are adequate per-weight, but recurrent error amplification through W_h makes the mapping chaotic—an artifact of BPTT-50 training applied over 1024 uncontrolled timesteps.

We now change approach. Instead of mapping *from* the RNN's internal representation *to* SN patterns, we build patterns *directly from the data.* The pattern-chain UM has no hidden layer: all patterns chain from observed input bytes to predicted output bytes. The RNN's hidden state is replaced by an expanded input space that unfolds the time dimension.

1.1 Model Taxonomy

We use the following names (introduced in export-gap.pdf):

sat-rnn	Saturated RNN. 128 hidden, BPTT-50, 0.079 bpc.
doubled-E	Exact float UM isomorphic to sat-rnn. 0.079 bpc.
sn-quant	$[0, 255]$ SN-quantized UM from sat-rnn weights. 0.09–2.1 bpc.
ngram-um	Direct n-gram counting UM. 0.081 bpc at order 11.
pattern-chain	(This paper.) Explicit input-to-output chains from data. 0.067 bpc at order 12.

2 Unfolding Time

The sat-rnn factors its computation as: input byte → hidden state → output byte, with the hidden state carrying temporal context via recurrent weights W_h. The pattern-chain UM unfactors this: we multiply by time, expanding the input space from 256 bytes to all (byte, position) pairs.

In the RNN view, the input at each timestep is one of 256 events, and temporal context is implicit in the hidden state. In the pattern-chain view, the input at timestep t is the full observable history:

$$\{(x_0, 0), (x_1, 1), \ldots, (x_t, t)\}$$

Each element is a separate event in the expanded input space $I \times T$. Patterns connect subsets of these events to output events.

We note in passing that the same unfolding can be applied on the output side—predicting not just the next byte but (byte, position) pairs. This generalizes the framework but is not needed for the current results.

The n-gram ordering gives a natural hierarchy: order-k patterns use the k most recent input events as context. This corresponds to chains of length $k+1$ from input bytes through time to an output byte:

$$i_{t-k+1} \to i_{t-k+2} \to \cdots \to i_t \to o_y$$

3 The Global Prior (Order 0)

At order 0, we feed the log-count of each byte as a strength into the output accumulator. For byte b with count $c(b)$ in N positions:

$$A(b) = \log_2 c(b), \qquad P(b) = \frac{2^{A(b)}}{\sum_{b'} 2^{A(b')}} = \frac{c(b)}{N}$$

This is the marginal distribution, and it gives **4.74 bpc**.

This is the bias theorem: the global prior on output bytes is exactly the byte frequency distribution. In the pattern-chain UM, this replaces all layer biases from the RNN. We assert that all biases (hidden and output) can be found by working backwards from this frequency-based prior—the biases encode the average context, which is the marginal distribution.

4 Order-by-Order Results

We build a trie over the 1024-byte dataset and compute bpc at each order, using both exact-order and backoff strategies.

4.1 Exact Order

Using only order-k context (falling back to marginal when context is unavailable):

4.2 Backoff

Using the longest available context (try order k, back off to $k-1$, . . . , marginal):

The backoff model almost exclusively uses the highest available order (1012 of 1023 positions use order 12 context at max order 12), with lower orders serving only the first few positions where full context is unavailable.

Order	bpc	Patterns	Δbpc
0	4.739	52	−3.26
1	2.047	231	−2.69
2	0.559	341	−1.49
3	0.308	401	−0.25
4	0.269	453	−0.04
5	0.218	498	−0.05
6	0.162	532	−0.06
7	0.157	564	−0.00
8	0.139	590	−0.02
9	0.139	614	−0.00
10	0.112	634	−0.03
11	0.112	653	−0.00
12	0.112	669	−0.00

Table 1: Pattern-chain UM, exact order. Each row uses *only* that order's context. "Patterns" counts the distinct (context, output) pairs at that order.

5 Sparsity

The pattern-chain UM is a *vanishingly* sparse subset of $I^k \times O$:

This extreme sparsity is the fundamental property: the dataset selects a tiny subset of all possible patterns, and this subset carries all the predictive structure. The patterns are data-terms—probabilistic syllogisms grounded in actual co-occurrences.

6 Data-Terms

Every pattern in the pattern-chain UM is a data-term: a statement of the form "when input bytes $a_1, \ldots, a_k$ were observed in this order, output byte b receives s bits of log-support, where s reflects how often this co-occurrence appears in the data."

The strength s is the log-luck:

$$s = \log_2 \frac{c(\text{context}, \text{output})}{c(\text{context})} \quad \text{(conditional)}, \qquad s = \log_2 c(\text{pattern}) \quad \text{(raw count)}$$

These are probabilistic syllogisms: "A was observed; when A is observed, B follows with support s; therefore B receives s bits of evidence." The microstate view (export-gap.pdf, Section 6.6) shows that s is grounded in dataset positions—the microstates where the co-occurrence holds.

7 Backward Trie and Skip-Patterns

The contiguous n-gram patterns of Sections 4–5 use only the k most recent bytes as context. But the pattern-chain framework admits *skip-patterns*: chains where the input bytes are not contiguous. A skip-pattern uses bytes at offsets $d_1 < d_2 < \cdots < d_k$ before the output position, where the offsets need not be consecutive.

Max order	bpc	vs sat-rnn
1	2.047	+1.968
2	0.560	+0.480
3	0.306	+0.226
4	0.260	+0.181
5	0.205	+0.125
6	0.145	+0.066
7	0.135	+0.056
8	0.113	+0.033
9	0.108	+0.028
10	0.076	−0.003
11	0.073	−0.007
12	0.067	−0.013

Table 2: Backoff model. The pattern-chain UM surpasses the sat-rnn at order 10 and reaches 0.067 bpc at order 12. Total pattern inventory: 6,180 patterns across all orders.

Order	Non-zero patterns	Fraction of $I^k \times O$
1	231	3.5×10^{-3}
2	341	2.0×10^{-5}
3	401	9.3×10^{-8}
4	453	4.1×10^{-10}
5	498	1.8×10^{-12}
6	532	7.4×10^{-15}

Table 3: Pattern sparsity. The number of patterns grows slowly (roughly linearly in order), while the product space grows as 256^k. By order 6, only 10^{-14} of the space is occupied.

7.1 The Backward Trie

We discover skip-patterns by building a *backward trie*: starting from each output byte, we look backwards through increasing offsets and measure how much information each offset carries about the output.

The mutual information $I(x_{t-d}; y_t)$ between the input byte at offset d and the output byte quantifies this:

The MI drops from 2.69 bits at offset 1 (the contiguous bigram) to 1.62 bits at offset 10, but the decay is slow: distant bytes still carry substantial predictive information. This is expected for structured data like XML, where tag openings predict closings across many intervening bytes.

The backward trie also reveals the atomic patterns at each offset. For example, the most common output byte (space, count 127) has 8 distinct predecessors at offset 1 but 22 at offset 6. The number of distinct contexts grows roughly 3–5× from depth 1 to depth 6 for each output byte, confirming that longer-range patterns are real but sparser.

7.2 Skip-2-Grams

A skip-2-gram conditions the next byte on two previously observed bytes at offsets a and b (with $a < b$). Each skip-2-gram pattern is a point in $I^2 \times O$—the same product space as the contiguous

Offset d	MI (bits)	MI/$H(y)$
1	2.692	0.568
2	2.447	0.516
3	2.190	0.462
4	2.029	0.428
5	1.916	0.404
6	1.816	0.383
7	1.742	0.368
8	1.742	0.368
9	1.594	0.336
10	1.616	0.341

Table 4: Mutual information between input byte at offset d and output byte. $H(y) = 4.74$ bits. Even offset 10 carries 1.62 bits—a third of the output entropy.

bigram (Section 5), but the two input events need not be adjacent. We measure $H(y \mid x_{t-a}, x_{t-b})$, the conditional entropy of the output given these two non-contiguous inputs.

Off. a	Off. b	$H(y\|x_a, x_b)$	Patterns
1	2	0.556	340
1	3	0.512	376
1	4	0.511	421
1	7	0.498	495
1	8	0.495	510
2	8	0.608	518
2	11	0.570	557
2	14	0.582	576
3	12	0.658	581
3	10	0.693	555

Table 5: Selected skip-2-gram results. The best pair (1,8) reaches $H = 0.495$ bpc with 510 patterns. All pairs involving offset 1 are strong because the immediate predecessor carries the most MI.

7.3 Skip-2-Gram Prediction

For practical prediction, we use a *backoff* strategy: try the most specific context first (the skip-2-gram), and if that context was never observed in the data, fall back to a less specific one (the bigram), then the marginal. This is the same principle as Section 4.2, extended to non-contiguous contexts. We fix offset 1 and vary the second offset:

The best second offset is 2 (the trigram), achieving 0.817 bpc with 340 patterns, down from the bigram's 2.042 bpc. Distant offsets also help: offset 15 still reaches 1.165 bpc, well below the bigram baseline. In each case, we are conditioning the next character on a sparse subset of "things we have recently seen"—pairs drawn from I^2, where I is the (byte, timestep) input space. As Elman (1990) observed, the RNN handles time implicitly through its recurrent state; the skip-k-gram makes the same temporal conditioning explicit.

Second offset	bpc	Skip-2 used	Bigram fallback
2	0.817	89.4%	10.5%
3	0.892	85.3%	14.6%
4	0.892	82.9%	17.0%
5	0.968	80.7%	19.2%
8	1.052	76.2%	23.7%
10	1.054	76.1%	23.8%
15	1.165	73.8%	26.1%

Table 6: Skip-2-gram backoff predictor (offset 1 + second offset). Contiguous bigram alone: 2.042 bpc. Best skip-2-gram: **0.817 bpc** (adding offset 2, i.e. the trigram context) from 340 patterns.

7.4 The Backward Trie Finds Skip-Patterns

The backward trie and skip-2-gram analyses are two views of the same structure. The backward trie starts from each output byte and discovers which input offsets carry mutual information about it. The skip-2-gram analysis measures the joint predictive power of offset pairs. Both identify the same skip-patterns—co-occurrences between non-contiguous input bytes and the output.

The backward trie is the more natural construction: it does not require enumerating all $\binom{D}{k}$ offset combinations. Instead, it grows outward from the output, adding offsets in order of diminishing MI. This is the pattern-chain analogue of the RNN's hidden state: where W_h carries information across time implicitly through recurrent neurons, the backward trie carries it explicitly through catalogued skip-patterns.

In the sat-rnn, skip connections are latent. A pattern like "< predicts > after some intervening bytes" lives in the recurrent weights W_h—the same weights that cause chaotic amplification when quantized (export-gap.pdf). The backward trie extracts these patterns directly from data, with no hidden layer to amplify errors. Each discovered pattern is a point in $I^k \times O$: a sparse subset of all possible input-output co-occurrences, just as in Section 5.

7.5 Greedy Skip-k-Grams

We extend to k non-contiguous input bytes, selected greedily: starting from offset 1 (the most informative single offset), we repeatedly add the offset that minimizes $H(y \mid \text{selected offsets})$.

Three observations:

1. **Offset 8 before offset 2.** The greedy selector chooses offset 8 as the second input, not offset 2 (the trigram). The byte 8 positions back carries more *unique* information than the immediately preceding byte, because offset 1 and offset 2 are highly correlated (both within the same XML token), while offset 8 crosses a tag boundary.

2. **Four bytes nearly match twelve.** The skip-4-gram $[1, 8, 20, 3]$ at 0.069 bpc is within 0.002 of the contiguous order-12 model (0.067 bpc), using 712 patterns vs 6,180. The remaining contiguous bytes add redundant information already captured by the skip offsets.

3. **Structural distances, not autocorrelation.** The dataset (enwik XML) has strong autocorrelation at offsets 18 (13.6%, half a tag line) and 47 (20%, one full tag line), but the greedy selector ignores these. Autocorrelation means the byte is predictable from nearby bytes—*redundant*. The greedy offsets $\{1, 3, 8, 20, 27\}$ have low autocorrelation (3–10%) but

k	Greedy offsets	Skip bpc	Contig. bpc	Patterns
1	$[1]$	2.047	2.047	231
2	$[1, 8]$	0.497	0.555	511
3	$[1, 8, 20]$	0.120	0.301	700
4	$[1, 8, 20, 3]$	0.069	0.256	712
5	$[1, 8, 20, 3, 27]$	0.056	0.200	810
6	$[1, 8, 20, 3, 27, 2]$	0.049	0.141	818
7	$[1, 8, 20, 3, 27, 2, 12]$	0.045	0.131	830
8	$[1, 8, 20, 3, 27, 2, 12, 7]$	0.043	0.108	834

Table 7: Greedy skip-k-gram vs contiguous k-gram. The greedy selector discovers offsets reflecting XML structure periodicities (8, 20, 27). At $k = 4$, the skip model (0.069 bpc) nearly matches the contiguous order-12 backoff (0.067 bpc) with 712 vs 6,180 patterns. At $k = 5$ it surpasses it (0.056 bpc).

high *complementary* MI: they cross different structural boundaries (within-token, tag boundary, half-line, full-line) and each carries information the others lack.

8 Comparison: Pattern-Chain vs Sat-RNN

Model	bpc	Patterns	Notes
Uniform	8.000	0	No patterns
Marginal (order 0)	4.739	52	Global prior
Bigram (order 1)	2.047	231	
Skip-2-gram [1,8]	0.497	511	Non-contiguous
sat-rnn	0.079	3,048*	128 hidden neurons
ngram-um (order 11)	0.081	—	From sat_ngram_um.c
pattern-chain (order 10)	0.076	6,180	Surpasses sat-rnn
Skip-4-gram [1,8,20,3]	0.069	712	4 offsets $\approx$ 12 contig.
pattern-chain (order 12)	0.067	6,180	Best contiguous
Skip-5-gram [1,8,20,3,27]	0.056	810	Surpasses all contig.
Skip-8-gram (greedy-8)	0.043	834	Best overall

Table 8: Model comparison. *SN pattern count from sat_sn_full.c (522 W_x + 1,006 W_h + 1,471 W_y + 49 biases). The skip-4-gram nearly matches the contiguous order-12 model with 712 vs 6,180 patterns. The skip-5-gram surpasses all contiguous models.

The contiguous pattern-chain UM surpasses the sat-rnn at order 10 (0.076 vs 0.079 bpc). But the skip-k-gram results reframe this: 4 optimally-placed non-contiguous bytes achieve 0.069 bpc with only 712 patterns, nearly matching the 12-contiguous-byte model's 0.067 bpc from 6,180 patterns. Five non-contiguous bytes surpass it entirely (0.056 bpc, 810 patterns).

The sat-rnn is more *efficient* in the model-complexity sense: 3,048 SN patterns (derived from ∼50K weights) achieve 0.079 bpc. But the skip-k-gram shows that pattern *placement* matters more than pattern *count*: 712 well-placed patterns beat 6,180 contiguous ones, and 810 well-placed patterns beat the RNN's 3,048.

This reveals what the RNN's hidden state actually does: it learns which offsets to attend to. The W_h matrix is an implicit offset selector—it preserves information from structurally important positions (offsets 8, 20, 27 in our data) while letting irrelevant positions decay. The greedy skip-k-gram makes this selection explicit.

9 Implications

1. **The gap is closed.** The pattern-chain UM reaches 0.067 bpc without any reference to the RNN's internal structure. The SN export gap (export-gap.pdf) is bypassed entirely by building patterns from data rather than from weights.

2. **Interpretation is direct.** Every pattern in the inventory is a readable data-term: "after seeing bytes $a_1 \ldots a_k$, predict byte b with strength s." There is no hidden layer to interpret.

3. **The RNN's value is compression.** The sat-rnn achieves nearly the same bpc with far fewer parameters. Understanding *how* it compresses 6,180 data-terms into 3,048 SN patterns (through hidden neurons) is the next question.

4. **Sparsity is the structure.** The pattern inventory occupies $< 10^{-14}$ of the product space at order 6. This extreme sparsity *is* the structure of the data—the patterns are where the information lives.

5. **The backward trie is the attention map.** The backward trie discovers which input positions predict each output, ranked by MI. This is the static, exhaustive version of what attention mechanisms learn to do selectively at runtime. The RNN's W_h carries skip information implicitly; the backward trie makes it explicit.

Reproducibility

```
cd docs/archive/20260207
gcc -O2 -o pattern_chain pattern_chain.c -lm
./pattern_chain sat_model.bin enwik_1024.txt 12
gcc -O2 -o backward_trie backward_trie.c -lm
./backward_trie enwik_1024.txt
gcc -O2 -o skip2gram skip2gram.c -lm
./skip2gram enwik_1024.txt
gcc -O2 -o skip_kgram skip_kgram.c -lm
./skip_kgram enwik_1024.txt
```