Cascading Coverage Bottleneck in the UM KN Chain

Claude and MJC

March 2026

Abstract

The UM KN chain at order 6 creates neurons through cascading threshold creation: a k-gram neuron requires a $(k-1)$-gram neuron to exist and fire. At 100K enwik9 ($\tau = 1$), orders 3–4 achieve 100% coverage of external KN contexts, but orders 5–6 achieve only 10% and 4.6% coverage respectively. This cascading bottleneck explains why the UM scores worse than external KN at scales above 150K: the model lacks the high-order contexts that provide the largest gains at larger data sizes.

1 The Cascade Mechanism

In the UM KN chain, neurons are created by cascading threshold:

1. `byte_prev` → `byte_output`: LPP fires at every position. When a context is seen τ times, creates a `bigram_prev` neuron.
2. `bigram_prev` → `byte_output`: fires only when the matching bigram neuron is active. Creates `trigram_prev` neurons at threshold.
3. And so on up to order 6.

A k-gram neuron can only be created when:

- Its $(k-1)$-gram parent neuron already exists,
- The parent neuron has been observed at least τ times.

This creates a timing dependency: higher-order neurons are born later in the data stream, after all prerequisite neurons have been created and fired enough times.

2 Coverage at 100K ($\tau = 1$, order 6)

ES / KN order	UM neurons	External KN	Coverage
bigram_prev (order 3)	2252	2252	100.0%
trigram_prev (order 4)	9398	9398	100.0%
4gram_prev (order 5)	2111	21336	9.9%
5gram_prev (order 6)	1607	35003	4.6%
Total	15368	67989	22.6%

Orders 3–4 achieve perfect coverage because their cascade depth is shallow (1–2 levels). Orders 5–6 require 3–4 levels of cascade, and many contexts are never created because their prerequisite neurons don't fire enough times in 100K bytes.

3 Coverage at 500K

ES / KN order	UM neurons	External KN	Coverage
bigram_prev (order 3)	4540	4540	100.0%
trigram_prev (order 4)	31680	23514	134.7%*
4gram_prev (order 5)	6509	63620	10.2%
5gram_prev (order 6)	5421	118063	4.6%

*Trigram_prev exceeds external count, likely due to hash collisions in the LPP table creating spurious entries.

4 Scaling Implications

Size	UM frozen	Ext. KN-6	Δ	Coverage
100K	2.628	2.719	−0.091	22.6%
200K	2.599	2.511	+0.088	∼20%
300K	2.778	2.568	+0.210	∼18%
500K	2.920	2.489	+0.431	∼16%

At small scale (<150K), the UM's retroactive advantage (2× counts) overcomes the coverage gap. As data grows, external KN benefits from high-order contexts (5-gram, 6-gram) that the UM hasn't created.

5 The Solution: Pre-seeded Contexts

The fix is to pre-seed high-order contexts before running the online pass. Two approaches:

1. **Two-pass seeding.** Run pass 1 (online) to create low-order neurons. Then, before the retroactive pass, scan the data once more with the existing neurons to create the missing high-order neurons. This adds one scan but dramatically increases coverage.

2. **Direct hash-table seeding.** Create neurons for ALL observed n-gram contexts in a single preliminary scan (like external KN's initialization). Then run the UM's retroactive pass to fill in counts.

Approach 2 is equivalent to giving the UM external KN's context structure but letting it compute its own KN distributions via retroactive training. This should match or beat external KN at all scales, since the retroactive advantage applies everywhere.

6 Conclusion

The UM KN chain's scaling failure above 150K is caused by a cascading coverage bottleneck at orders 5–6. With $\tau = 1$, only 23% of external KN's contexts are created. Pre-seeding contexts would restore full coverage while preserving the retroactive training advantage. This is the most impactful single improvement available: it would combine external KN's coverage with the UM's $2\times$ count advantage.