Count-Augmented LPPs: KN-Calibrated Scoring for the UM

Claude and MJC

March 2026

Abstract

We augment the UM's LPP entries with exact observation counts and implement KN-discount chain scoring (mode 3). The KN chain computes $p_k = (c_b - D)/n + \lambda \cdot p_{k-1}$ at each LPP level, using the lower-order distribution as backoff. At order 3 (bigram + trigram with threshold creation), frozen bpc drops from 3.846 (sharpest-LPP) to **2.766** at 100K—a 1.08 bpc improvement and the largest scoring-rule gain in the UM. However, order 4+ still degrades: the KN discount mechanism gives 10% probability mass to seen-but-wrong bytes at each level, accumulating through the chain.

1 Implementation

Two changes to `#umr_core`:

Entry counts. Each `UM_Entry` gains an `int obs_count` field, incremented on every observation. This stores the exact count that the log-stochastic weight `w` approximates.

KN chain scoring (mode 3). Starting from a uniform unigram distribution, process LPPs in order. For each LPP with an active source event:

1. Compute per-byte counts from entry `obs_counts`.
2. Apply KN discount: $p(b) = \max(c_b - D, 0)/n + \lambda \cdot p_{\text{backoff}}(b)$ where $\lambda = D \cdot t_y/n$ and $t_y =$ distinct target types.
3. If $n \geq 2$ (minimum count threshold), update the running distribution. Otherwise keep the lower-order backoff.

The KN discount parameter $D = 0.9$ (same as external KN-6).

2 Results

At 1M, order-3 KN chain achieves **3.049 bpc** vs 3.744 bpc sharpest-LPP ($\Delta = -0.695$ bpc). The KN improvement shrinks with scale (1.080 at 100K $\rightarrow$ 0.695 at 1M) as the sharpest-LPP baseline also improves, but remains the largest single scoring-rule gain at every tested scale.

The KN chain dramatically improves every order compared to sharpest-LPP. But order scaling still degrades: order 3 beats order 4 by 0.929 bpc.

Order	Sharpest-LPP	KN Chain	Δ
2 (bigram only)	4.868	3.713	−1.155
3 (bi+tri)	3.846	**2.766**	−1.080
4	4.056	3.695	−0.361
6	4.074	3.699	−0.375

Table 1: 100K frozen bpc, 3-pass pipeline ($\tau = 4$, D = 0.9).

3 Why Order Scaling Still Degrades

In standard KN-6, higher orders improve because:

1. A context exists only when its exact byte sequence has been observed.
2. High-order contexts have high counts for the correct byte.

In the UM, threshold-created neurons behave differently:

1. A bigram neuron fires whenever its pair occurs, regardless of the downstream LPP's data quality.
2. The 4-gram LPP's entries may have only 1–2 observations per source, even though the source neuron fires frequently.

The KN discount with $D = 0.9$ allocates $\lambda = D \cdot t_y/n$ mass to the backoff. For a context with $n = 3$ observations of 2 distinct targets: $\lambda = 0.9 \cdot 2/3 = 0.6$, leaving 40% on the observed (possibly wrong) bytes. When the correct byte is NOT among those observed, the prediction gets 60% of the trigram probability—a 40% hit versus using trigram directly.

Over many positions, this accumulates. At 100K, the 4-gram LPP has very few observations per context, causing persistent probability mass loss.

4 Threshold Effects

Raising the threshold τ reduces the number of neurons but does not fix order degradation:

τ	Order 3	Order 4
4	**2.766**	3.695
6	3.042	3.698
8	3.319	3.712
10	3.599	3.599

At $\tau = 10$, order 4 matches order 3 because so few 4-gram neurons exist that the chain always falls through to the trigram level. But $\tau = 4$ gives the best order-3 result because more bigram neurons = more trigram data.

5 Comparison to External KN-6

External KN-6 at 100K (all 6 orders, hash table counting): ∼2.6 bpc. The UM order-3 KN chain achieves 2.766 bpc with only bigram + trigram and threshold creation. The gap is ∼0.2 bpc, likely from orders 3–6 of the KN chain and from hash-table counting being more accurate than the UM's threshold-mediated approach.

6 The Path Forward

The KN chain scoring proves that *probability calibration is the fix*, confirming the representation bottleneck diagnosis. The remaining degradation at order 4+ is not a calibration problem but a *coverage* problem: threshold-created neurons fire at every matching position but accumulate data only at their own (source, target) entries. The 4-gram "context" is not a true 4-gram—it's a bigram neuron firing status, which is much broader.

Two approaches:

1. **Context hashing**: store the actual byte sequence as context (like KN-6) rather than relying on threshold-created neurons. This converges toward the external KN-6 implementation.

2. **Hierarchical backoff**: instead of unconditionally applying higher-order LPPs, condition on a reliability measure (e.g., $n \geq n_{\min}$ with $n_{\min}$ increasing with order). This keeps the UM's native architecture but adds a data-dependent gating mechanism.

7 Conclusion

Count-augmented LPPs with KN chain scoring achieve 2.766 bpc at order 3 (100K), a 1.08 bpc improvement over sharpest-LPP and the single largest scoring improvement in the UM's history. The remaining 0.2 bpc gap to external KN-6 and the persistent order degradation point to the threshold-creation architecture as the next bottleneck: threshold-created contexts are too broad (firing whenever the bigram matches) to support reliable high-order prediction.