RNN Memory Depth: Experiment and Results

Hutter RNN Project

January 31, 2026

1 Motivation

From the time-energy analysis, we derived a prediction for RNN memory depth:

$$d_{\max} = \frac{24}{H_{\text{avg}}} \approx \frac{24}{2} = 12 \text{ characters}$$

The reasoning: float32 has 24 bits of mantissa. Each timestep "costs" H_{avg} bits of entropy. After $d_{\max}$ steps, precision is exhausted and earlier information is washed out.

This experiment tests whether the RNN's effective memory matches this prediction.

2 Method 1: Perturbation (Failed)

2.1 Approach

For each sample position:

1. Run RNN on context, get prediction P_{orig}
2. Flip byte at distance k back
3. Run RNN on modified context, get prediction P_{mod}
4. Measure KL divergence $D_{KL}(P_{\text{orig}} \| P_{\text{mod}})$

2.2 Results

Dependency *increased* with distance—the opposite of expected.

2.3 Problem

When you flip a byte at position $-k$, the perturbation propagates through k steps of hidden state evolution. Flipping farther back = more accumulated effect, not less.

This measures **perturbation propagation**, not memory depth.

3 Method 2: Conditional Variance

3.1 Approach

For each distance k:

1. Collect many samples with their full context
2. Group samples by the byte value at position $-k$
3. Compute mean prediction for each group
4. Measure variance of means across groups

If the RNN "remembers" position $-k$, then predictions should vary depending on what byte was there. Higher variance = more dependency on that position.

3.2 Implementation

```
for k in range(1, max_dist + 1):
    predictions_by_byte = defaultdict(list)

    for pos in sample_positions:
        context = data[pos - context_len : pos]
        byte_at_k = context[-k]
        _, probs = forward(model, context)
        predictions_by_byte[byte_at_k].append(probs)

    # Compute variance of group means
    group_means = [mean(preds) for preds in predictions_by_byte.values()]
    variance[k] = var(group_means)
```

3.3 Results

Distance k	Variance	Normalized
1	0.000610	0.76
2	0.000767	0.96
3	0.000713	0.89
4	0.000799	1.00
5	0.000667	0.83
10	0.000797	1.00
15	0.000613	0.77
20	0.000500	0.63
25	0.000471	0.59
30	0.000470	0.59

Observation: No clear exponential decay. Dependency is roughly uniform out to 30 characters, with some noise.

4 Analysis

4.1 Expected vs Observed

Predicted	Exponential decay, 1/e at $k \approx 12$
Observed	Roughly flat, no clear decay to $k = 30$

4.2 Possible Explanations

1. The 24-bit limit applies to gradients, not inference.

The depth limit derivation assumed that carrying information through W_{hh} multiplication loses precision. But:

- During inference, we only need relative magnitudes
- The hidden state is normalized by tanh
- Information might be encoded in directions, not magnitudes

2. The RNN has learned efficient encoding.

The trained W_{hh} might have learned to preserve information more efficiently than a random matrix would. The singular value structure of W_{hh} could concentrate information in stable subspaces.

3. English has redundancy that extends memory.

Natural language has long-range correlations. The RNN might be exploiting statistical regularities (word boundaries, sentence structure) that effectively extend its memory beyond what raw precision would allow.

4. The measurement method has issues.

Variance of group means might not be the right statistic. The signal might be too weak relative to noise. Need more samples or a different approach.

5 Connection to Theory

The depth limit formula:

$$d_{\max} = \frac{24}{-\log_2 p_{\text{avg}}}$$

This was derived for arithmetic coding intervals, where precision loss is exact and cumulative. For RNNs:

- tanh normalization prevents unbounded growth
- The hidden state is a distributed representation
- Information can be encoded redundantly across dimensions

The formula might be a *lower bound* (you can't do worse than this) rather than an *exact prediction*.

6 Future Work

1. **Probe hidden state directly**: Instead of looking at output predictions, look at how hidden state h_n depends on input at position $-k$.
2. **Analyze W_{hh} eigenvalues**: The spectral radius of W_{hh} determines how quickly information decays. If $|\lambda_{\max}| \approx 1$, information persists.
3. **Synthetic experiment**: Train on data with known memory requirements (e.g., matching brackets at distance k). See if performance degrades at $k > 12$.
4. **Compare with untrained model**: Does a randomly initialized RNN show the predicted decay?

7 Conclusion

The predicted memory depth of $\sim$12 characters was not confirmed. Observed dependency is roughly flat out to 30 characters.

This suggests either:

- The precision limit is not the dominant factor
- The RNN has learned to work around it
- The measurement method needs refinement

The experiment is documented here for future investigation.

8 Files

- `memory_depth.py` — Method 1 (perturbation, failed)
- `memory_depth2.py` — Method 2 (conditional variance)
- `memory-depth.html` — Interactive visualization
- `memory_depth_results.npz` — Raw data