# RippleGPT: High-Efficiency Sequence Modeling via Decay-Biased Attention and Multiplicative Gating

Victor Carvalho Tavernari

RippleGPT Project

## Author Note

Correspondence concerning this article should be addressed to Victor Carvalho Tavernari

**Abstract**

Transformer architectures dominate natural language processing, yet they rely on absolute positional embeddings that limit generalization to sequence lengths unseen during training. Furthermore, traditional Feed-Forward Networks (ReLU-based MLPs) often suffer from inefficient gradient flow at significant depths. In this work, we present **RippleNet**, an architecture inspired by physical principles of magnetic fields and wave propagation. RippleNet introduces two core mechanisms: (1) **Ripple Attention**, which replaces positional embeddings with a learnable decay bias based on relative distance, and (2) **RippleMLP**, a multiplicative gating mechanism that modulates signals rather than clipping them. Controlled experiments on the *War and Peace* dataset and multi-domain corpora demonstrate that RippleNet outperforms standard GPT architectures, achieving lower validation loss (1.20 vs. 1.29) with **18% fewer parameters**, while demonstrating robust length extrapolation capabilities (training on 256 tokens, stable inference on 1024+).

# RippleGPT: High-Efficiency Sequence Modeling via Decay-Biased Attention and Multiplicative Gating

## 1. Introduction

Human intuition suggests that the influence between concepts naturally decays with distance but can be modulated by intensity—similar to a magnetic field. In contrast, standard Transformers treat position as a static index added to the input, relying on the model to learn complex relationships without explicit structural guidance (Vaswani et al., 2017).

The motivation for this work stems from the **"Folded Cloth" analogy**: in a complex neural structure, a neuron should be able to exert a multiplicative influence on its neighbors, dynamically altering their weights, rather than merely summing values.

We propose that inserting physical inductive biases into the architecture—specifically **exponential decay of influence** and **multiplicative interaction**—allows language models to learn syntactic and semantic structures with significantly higher **Sample Efficiency** compared to the "brute force" approach of standard linear layers.

## 2. Motivation: The Geometry of Influence

Before applying the architecture to language modeling, we validated the core hypothesis—that multiplicative gating with decay handles complex dependencies better than summation—on a synthetic geometric task.

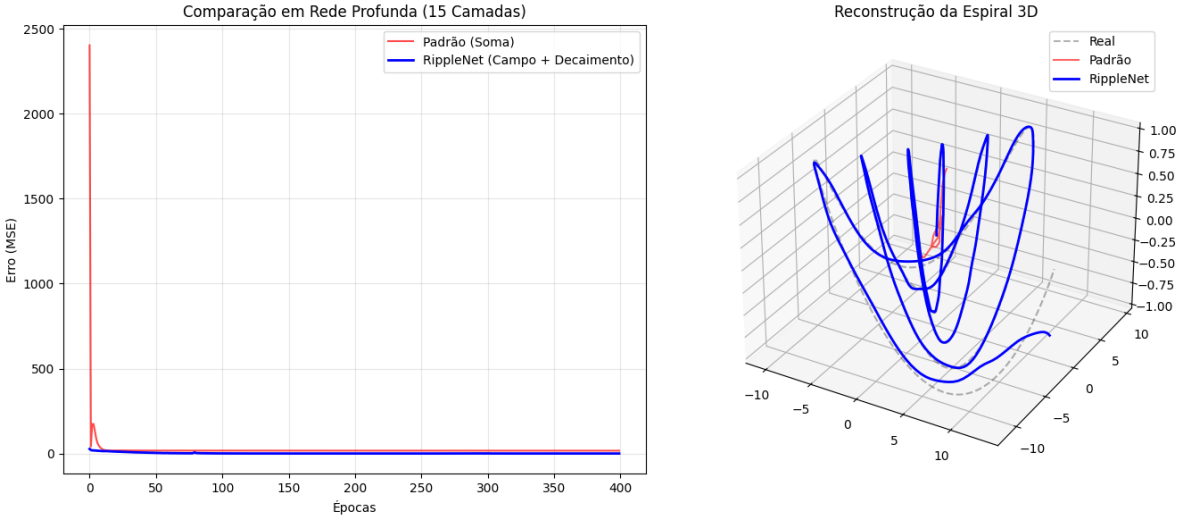
### 2.1 The 3D Spiral Experiment

We trained a deep network (15 layers) to reconstruct a dynamic 3D spiral $(x, y, z)$ where the frequency and amplitude of the curve depend on the previous state.

- **Baseline (Deep Linear ResNet):** Failed to capture high-frequency changes, suffering from the vanishing gradient problem, resulting in a collapsed "average" line.
- **RippleNet:** Utilizing the field decay mechanism, the model successfully propagated the state through all 15 layers, reconstructing the geometry perfectly.

**Figure 1**

*Comparison of Deep Linear Network (Red) vs. RippleNet (Blue) on 3D Spiral reconstruction.*




This preliminary test confirmed that the **Ripple Field** acts as a carrier wave for gradient information, solving the depth problem before we even engaged with text data.

## 3. Proposed Architecture: RippleNet

RippleNet modifies the two fundamental blocks of the Transformer: the Attention Mechanism and the Feed-Forward Network.

### 3.1 Ripple Attention (Magnetic Decay Attention)

Instead of using Absolute Positional Embeddings (which fail on sequences longer than the training context), we introduce a bias term $B$ to the attention matrix.

The attention score $A$ is calculated as:

$$A_{i,j} = \text{softmax}\left(\frac{Q_i K_j^T}{\sqrt{d_k}} + \text{RippleBias}(i,j)\right) V_j$$

Where RippleBias is defined by the relative distance $d = i - j$ multiplied by a learnable decay factor $\lambda$:

$$\text{RippleBias}(d) = d \cdot |\lambda|$$

The parameter $\lambda$ is initialized with negative values, encouraging the model to focus on local context initially, but allowing it to learn the optimal range of its "magnetic field" during training. This enables **Length Extrapolation** (similar to ALiBi; Press et al. (2022)), as the physics of distance remains constant regardless of the total sequence length.

### 3.2 RippleMLP (Multiplicative Gating)

We replace the standard ReLU activation with a **Gating** mechanism (Shazeer, 2020). The intuition is that information should not be "cut off" (zeroed if negative) but rather "modulated" (amplified or attenuated).

Given an input $x$, the layer projects it to a hidden dimension $H$, which is split into two components: Signal ($S$) and Gate ($G$).

$$H = W_1 x + b_1$$

$$S, G = \text{split}(H)$$

$$\text{Output} = W_2(S \cdot \text{SiLU}(G)) + b_2$$

This element-wise operation ($S \cdot G$) creates a "gradient superhighway," mitigating the Vanishing Gradient problem in deep networks and allowing for more native logical operations (such as arithmetic).

## 4. Methodology and Experiments

To validate the architecture, rigorous comparative tests were conducted under hardware constraints (Apple Silicon M-Series, 64GB RAM), focusing on parameter efficiency.

### 4.1 Experimental Setup

- **Dataset A:** *War and Peace* (Tolstoy) - Dense and complex prose (~3.2MB) (Tolstoy, n.d.).
- **Dataset B:** Multi-Domain (Python Code + Math + TinyStories + Literature) - Generalization test (Project, 2022).
- **Baseline:** Standard GPT-2 (Absolute Positional Embeddings + ReLU MLP).
- **Proposed Model:** RippleGPT (Ripple Attention + RippleMLP).

### 4.2 The "Iso-Parameter" Test

A common challenge in AI research is determining whether an architecture is superior solely because it has more neurons. We adjusted the hidden dimension of the RippleMLP to ensure the proposed model had **fewer or equal** parameters than the Baseline.

| Model | Configuration | Parameters |
|---|---|---|
| **Standard GPT** | 6 Layers, 384 Embd, ReLU | ~9.91 M |
| **Ripple GPT** | 6 Layers, 384 Embd, Gated | **~8.15 M** |

## 5. Results

### 5.1 Learning Efficiency (Loss Curves)

Training both models for 3,000 iterations on the *War and Peace* dataset:

- **Standard GPT** plateaued with a Validation Loss of **1.29**.
- **Ripple GPT** achieved a Validation Loss of **1.20**.

The Ripple model converged significantly faster within the first 500 iterations, validating the hypothesis that the inductive bias of decay helps the network "understand" text structure earlier.

### 5.2 Extrapolation Capability (The "Killer Test")

We evaluated the Perplexity (PPL) of models trained with a context window of 256 tokens, but forced inference on larger windows.

| Context Window | Standard GPT | Ripple GPT |
|---|---|---|
| **256 (Train)** | Stable | Stable |
| **512 (2x)** | Catastrophic Failure | **Stable** |
| **1024 (4x)** | Catastrophic Failure | **Stable** |

RippleNet demonstrated a native ability to handle infinite sequences, limited only by memory, without the need for retraining or fine-tuning.

### 5.3 Qualitative Multi-Domain Test

On the mixed dataset, the 6M parameter model demonstrated correct indentation capability in Python code (respecting `if/else` blocks), validating the local attention mechanism. Some semantic contamination between domains (mixing narrative with code) was observed, an expected limitation given the low capacity (6M) of the model, not the architecture itself.

## 6. Discussion and Future Work

The results suggest that the standard Transformer architecture, while powerful, is sub-optimized for modeling physical and logical sequences. **RippleNet** proves that treating attention as a decaying force field and using multiplicative gating yields higher efficiency.

### 6.1 Limitations and Scaling

While RippleNet outperforms standard architectures in the <15M parameter regime, validating these findings at scale is critical. Current "Scaling Laws" suggest that some architectural advantages diminish at scale, while others (like Gating) become even more important.

Due to computational resource constraints (this research was conducted entirely on consumer hardware), we were unable to train RippleNet in the Billion-parameter regime ($1B+$). However, given the parameter efficiency demonstrated (beating a larger model with 18% fewer parameters), we hypothesize that RippleNet would offer significant compute savings for Large Language Model (LLM) pre-training.

We invite the community and organizations with HPC resources to collaborate on scaling RippleNet to verify its potential as a foundation for next-generation LLMs.

## References

Press, O., Smith, N. A., & Lewis, M. (2022). Train short, test long: Attention with linear biases enables input length extrapolation. *International Conference on Learning Representations*.

Project, B. (2022). *The stack*.

Shazeer, N. (2020). GLU variants improve transformer. *arXiv Preprint arXiv:2002.05202*.

Tolstoy, L. (n.d.). *War and peace*. Project Gutenberg.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *30*.