Titans vs Transformers : The Mathematics

Miquel Noguer i Alonso
Artificial Intelligence Finance Institute

January 17, 2025

Abstract

This paper presents a comprehensive mathematical formalization of the Titans architecture, a groundbreaking sequence modeling framework introduced by Behrouz et al. [2024]. Titans reimagine sequence modeling by incorporating an adaptive neural memory module capable of efficiently capturing long-term dependencies. This design addresses the limitations of Transformers Vaswani et al. [2017], which rely on computationally intensive self-attention mechanisms constrained by quadratic time and memory complexity. Titans utilize hierarchical memory systems, including contextual and persistent memory, to process vast context windows while maintaining computational scalability. By formalizing key components such as memory dynamics, surprise metrics, and architectural state spaces, this work highlights Titans' adaptability and theoretical robustness. Comparative analyses demonstrate that Titans outperform Transformers across long-sequence tasks in both computational efficiency and predictive accuracy, offering a transformative approach for applications in language modeling, genomics, and time-series forecasting.

1 Introduction

The Titans architecture represents a groundbreaking approach to sequence modeling, introduced by Behrouz et al. [Behrouz et al., 2024]. Unlike traditional neural network architectures, Titans fundamentally reimagine computational mechanisms through an adaptive, memory-centric design. This work builds upon and extends seminal research in neural memory systems [Hochreiter and Schmidhuber, 1997, Vaswani et al., 2017], offering a more flexible and efficient computational framework.

2 Titans Foundational Mathematical Framework

2.1 Architectural State Space Formalization

Definition 1 (Comprehensive State Space $\mathcal{A}$)**.** *The Titans architectural state space is defined as a multidimensional tuple:*

$$\mathcal{A} = (\mathcal{M}, \mathcal{X}, \mathcal{L}, \Theta, \Phi)$$

This tuple encapsulates the core components of the Titans architecture, each of which plays a critical role in its operation. The components are:

- $\mathcal{M}$*: Neural Memory State Space*

  $$\mathcal{M} = \{m \in \mathbb{R}^{d \times d} : \textit{rank}(m) \leq \min(d, m), m \textit{ is adaptive}\}$$

  The memory state space $\mathcal{M}$ represents the adaptive memory matrices used by Titans. These matrices are constrained by their rank and dimensionality, ensuring efficient storage and retrieval of information.

- $\mathcal{X}$*: Input Domain*

  $$\mathcal{X} = \{x \in \mathbb{R}^{n \times d} : \|x\|_F \leq B, x \textit{ is contextually dynamic}\}$$

  The input domain $\mathcal{X}$ consists of dynamically changing input sequences, bounded by the Frobenius norm to ensure stability in processing.

- $\mathcal{L}$*: Loss Function Space*

  $$\mathcal{L} = \{l : \mathcal{M} \times \mathcal{X} \to \mathbb{R}_+ : l \textit{ is differentiable and adaptive}\}$$

  The loss function space $\mathcal{L}$ contains differentiable and adaptive loss functions that guide the learning process.

- Θ*: Hyperparameter Configuration*

  $$\Theta = \{\theta = (\eta, \lambda, \alpha) : \eta, \lambda, \alpha \in (0, 1), \textit{dynamically updateable}\}$$

  The hyperparameter configuration Θ includes parameters such as learning rates and regularization terms, which are dynamically adjusted during training.

- Φ*: Transformation Operator*

  $$\Phi : \mathcal{M} \times \mathcal{X} \to \mathcal{M}$$
  $$\Phi(M_{t-1}, x_t) = (1 - \alpha_t)M_{t-1} + \beta_t \nabla \ell(M_{t-1}; x_t)$$

  The transformation operator Φ updates the memory state based on the current input and the gradient of the loss function, ensuring adaptive learning.

2.2 Neural Memory Dynamics

Definition 2 (Adaptive Memory Update Operator)**.** *The memory update operator* Φ_ω *is defined as:*

$$\Phi_\omega(x_t, M_{t-1}) : \mathbb{R}^{d_{in}} \times \mathbb{R}^{d \times d} \to \mathbb{R}^{d \times d}$$

This operator governs how the memory state evolves over time. It is constrained by the existence of partial derivatives with respect to both the input and the previous memory state, ensuring smooth updates. Additionally, the Frobenius norm of the updated memory is bounded to prevent instability.

With constraints:

$$\frac{\partial \Phi_\omega}{\partial x_t} \text{ exists}$$
$$\frac{\partial \Phi_\omega}{\partial M_{t-1}} \text{ exists}$$
$$\|\Phi_\omega(x_t, M_{t-1})\|_F \leq C$$

The update mechanism is further refined by the following equations:

$$\alpha_t = \sigma\left(\frac{\partial \ell}{\partial M_{t-1}} \cdot x_t\right)$$
$$\beta_t = \exp\left(-\gamma \|\nabla \ell(M_{t-1}; x_t)\|_2\right)$$

Here, σ *is the sigmoid function, and* γ *is a regularization parameter that controls the rate of memory decay. This mechanism ensures that the memory update is both adaptive and stable [Hochreiter and Schmidhuber, 1997].*

2.3 Surprise Metric Formalization

Definition 3 (Generalized Surprise Metric)**.** *The surprise metric* S_t *is defined as a stochastic information potential:*

$$S_t = \eta_t S_{t-1} - \theta_t \nabla \ell(M_{t-1}; x_t)$$
$$\mathcal{I}(S_t) = -\sum_i \sigma_i(S_t) \log(\sigma_i(S_t))$$

The surprise metric quantifies the unexpectedness of new information relative to the current memory state. It is computed recursively, with a temporal decay coefficient η_t *and a scaling factor* θ_t *that adjusts the impact of the gradient of the loss*

function. The information entropy $\mathcal{I}(S_t)$ measures the uncertainty in the surprise metric, providing insights into the system's learning dynamics.

Key components:

- $\eta_t \in (0, 1)$*: Temporal decay coefficient*
- $\theta_t > 0$*: Momentary surprise scaling*
- $\mathcal{I}(S_t)$*: Information entropy of surprise metric*

Theorem 1 (Surprise Metric Convergence)**.** *The surprise metric S_t converges to an optimal information representation:*

$$\lim_{t\to\infty} \mathcal{I}(S_t) = \min_{\nu} \mathbb{E}_{x\sim P}[\mathcal{D}_{KL}(P||\nu)]$$
$$\|\nabla S_t\|_2 \leq C \exp(-\gamma t)$$

Where $\mathcal{D}_{KL}$ is the Kullback-Leibler divergence. This theorem demonstrates that the surprise metric stabilizes over time, minimizing the divergence between the true distribution P and the learned representation ν [Kullback and Leibler, 1951].

3 Architectural Variants of Titans

3.1 Memory as Context (MAC)

Definition 4 (MAC Architecture Formalization)**.** *The MAC variant introduces a context-integrated memory update:*

$$\begin{aligned} h_t &= M^*_{t-1}(q_t) \\ \tilde{S}^{(t)} &= [p_1, p_2, \ldots, p_{N_p}] \| h_t \| S^{(t)} \\ y_t &= Attn(\tilde{S}^{(t)}) \\ M_t &= M_{t-1}(y_t) \end{aligned}$$

The MAC variant leverages persistent memory parameters p_i and a query projection q_t to integrate contextual information into the memory update. The attention mechanism Attn ensures that the most relevant information is retained [Vaswani et al., 2017].

Where:

- M^**: Memory inference without parameter update*

- q_t: *Query projection*
- p_i: *Persistent memory parameters*
- *Attn: Attention mechanism*

3.2 Memory as Gate (MAG)

Definition 5 (MAG Architecture Formalization)**.** *The MAG variant introduces a gating mechanism:*

$$\begin{aligned}\tilde{x} &= [p_1, p_2, \ldots, p_{N_p}] \| x \\ y &= \textit{SW-Attn}^*(\tilde{x}) \\ o &= y \otimes M(\tilde{x})\end{aligned}$$

The MAG variant uses a sliding window attention mechanism SW-Attn* to process input sequences, combined with a non-linear gating operation $\otimes$ to control information flow [Hochreiter and Schmidhuber, 1997].

Where:

- *SW-Attn*: Sliding window attention*
- $\otimes$*: Non-linear gating operation*

3.3 Memory as Layer (MAL)

Definition 6 (MAL Architecture Formalization)**.** *The MAL variant incorporates memory as a processing layer:*

$$\begin{aligned}\tilde{x} &= [p_1, p_2, \ldots, p_{N_p}] \| x \\ y &= M(\tilde{x}) \\ o &= \textit{SW-Attn}(y)\end{aligned}$$

The MAL variant treats the memory module as a distinct processing layer, enabling hierarchical information extraction. The sliding window attention mechanism SW-Attn ensures efficient processing of long sequences [Vaswani et al., 2017].

Where:

- M*: Neural memory module*
- *SW-Attn: Sliding window attention*

4 Comparative Mathematical Analysis: Titans vs Transformers

4.1 Architectural Representation Spaces

Definition 7 (Transformer Architectural Space)**.** *The Transformer architectural space $\mathcal{T}$ is defined as:*

$$\mathcal{T} = (Q, K, V, W_Q, W_K, W_V, Softmax)$$

Where:

- Q, K, V*: Query, Key, Value matrices*
- W_Q, W_K, W_V*: Projection weight matrices*
- *Softmax: Attention normalization operator*

Transformation constraints:

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V$$
$$Attention(Q, K, V) = Softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

The Transformer architecture relies on self-attention mechanisms to process input sequences, which are computationally expensive for long sequences [Vaswani et al., 2017].

Definition 8 (Titans Architectural Space)**.** *The Titans architectural space $\mathcal{A}$ is defined as:*

$$\mathcal{A} = (\mathcal{M}, \Phi, \Psi, S, \nabla)$$

Where:

- $\mathcal{M}$*: Adaptive neural memory module*
- Φ*: Memory update transformation*
- Ψ*: Surprise metric operator*
- S*: Surprise state space*
- ∇*: Adaptive gradient operator*

Transformation dynamics:

$$\begin{aligned} M_t &= \Phi(M_{t-1}, x_t) \\ S_t &= \Psi(M_{t-1}, x_t) \\ \nabla M_t &= \frac{\partial \ell}{\partial M_{t-1}} \end{aligned}$$

The Titans architecture extends the Transformer by incorporating adaptive memory and surprise metrics, enabling more efficient processing of long sequences [Hochreiter and Schmidhuber, 1997].

Theorem 2 (Architectural Expressiveness Comparison)**.**

$$\dim(\mathcal{T}) < \dim(\mathcal{A})$$

This theorem demonstrates that Titans possess a higher representational capacity than Transformers, enabling more flexible and efficient sequence processing [Vaswani et al., 2017].

4.2 Computational Complexity Analysis

Definition 9 (Transformer Computational Complexity)**.** *For input sequence* $X \in \mathbb{R}^{n \times d}$*: Time Complexity:*

$$T_{Transformer}(n, d) = O(n^2 d)$$

$$\textit{Derived from Attention}(Q, K, V) = \textit{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Memory Complexity:

$$M_{Transformer}(n, d) = O(n^2)$$

Due to full attention matrix storage

The quadratic complexity of Transformers limits their scalability for long sequences [Vaswani et al., 2017].

Definition 10 (Titans Computational Complexity)**.** *For input sequence* $X \in \mathbb{R}^{n \times d}$*: Time Complexity:*

$$T_{Titans}(n, d) = O(nd \log n)$$

Incorporating adaptive memory update: $\Phi(M_{t-1}, x_t)$

Memory Complexity:

$$M_{Titans}(n, d) = O(nd + md)$$

With dynamic memory module of size m

Titans achieve superior scalability by leveraging adaptive memory updates, reducing both time and memory complexity [Hochreiter and Schmidhuber, 1997].

Theorem 3 (Complexity Scaling Comparison)**.**

$$\lim_{n\to\infty} \frac{T_{Transformer}(n, d)}{T_{Titans}(n, d)} \to \infty$$
$$\lim_{n\to\infty} M_{Transformer}(n, d) \gg M_{Titans}(n, d)$$

This theorem highlights the superior scalability of Titans compared to Transformers, particularly for long sequences [Vaswani et al., 2017].

5 Conclusion

The Titans architecture Behrouz et al. [2024] marks an advancement in the field of sequence modeling by fundamentally rethinking the role of memory in neural computation. Unlike Transformers Vaswani et al. [2017], which excel in capturing short-term dependencies but face prohibitive computational costs for longer sequences, Titans employ an adaptive memory module designed to capture both short- and long-term dependencies efficiently. This is achieved through mechanisms such as hierarchical memory integration and surprise-driven updates, which reduce computational complexity to logarithmic scales while retaining predictive precision.

The scalability of Titans enables them to process context windows exceeding 2 million tokens, a feat unattainable by standard Transformers. Their memory-centric design, coupled with novel architectural variants such as Memory as Context (MAC) and Memory as Gate (MAG), demonstrates superior performance across diverse applications, including language modeling, reasoning-intensive tasks, and genomics. This formalization underscores the potential of Titans to redefine neural architectures by prioritizing memory efficiency and adaptability, paving the way for future innovations in long-sequence processing and beyond.

References

Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time, 2024. URL `https://arxiv.org/abs/2501.00663`.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

Solomon Kullback and Richard A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

Ashish Vaswani, Noam Shazeer, Nikolai Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30:5998–6008, 2017.