# PROTEUS: Incremental Memory Activation for Long-Context Sequence Modeling

**Reza Bayat***
Mila
reza.bayat@mila.quebec

**Ali Behrouz***
Google Research
alibehrouz@google.com

**Vahab Mirrokni**
Google Research
mirrokni@google.com

**Aaron Courville**
Mila
courvila@mila.quebec

## Abstract

The quadratic cost of attention-based sequence models for long contexts has motivated a growing line of research on *memory-based* models that compress context into a compact state. However, existing fixed-size memory models expose a *static* memory throughout the entire sequence. Because early tokens face almost no compression pressure, they occupy too many degrees of freedom and "pollute" the memory, leaving little capacity for later context and increasing interference between what is stored and what arrives next. We study a new paradigm of *incremental memory activation*, in which the effective memory capacity is progressively expanded as the context grows. An early bottleneck encourages better compression of history, while unlocking fresh capacity over time reduces interference and improves retention of later context. We instantiate this paradigm in PROTEUS, a straightforward mechanism that can be incorporated into a broad class of neural memory architectures at no additional cost. We apply PROTEUS to state-of-the-art models, including SWLA, Comba, Titans, and Hope-Attention, and observe consistent improvements on language modeling and reasoning, as well as on long-context retrieval and understanding, with the largest gains at longer contexts. Overall, our results suggest that static memory is suboptimal and that scheduling effective capacity is a simple and broadly applicable approach for sequence modeling.

## 1 Introduction

Softmax attention, the backbone of Transformers [Vaswani et al., 2017] and the component mainly responsible for sequence mixing, acts as a lossless memory module that caches every single token in the sequence, so that its memory size grows linearly with sequence length [Ramsauer et al., 2021, Bietti et al., 2024, Behrouz et al., 2025c]. While this makes attention well suited to retrieval, the growing memory comes with a quadratic computational cost with respect to the sequence length, limiting the model's long-context handling.

To overcome the quadratic cost of Transformers, as well as their limited expressivity on tasks that require sequential reasoning, modern recurrent neural networks have regained popularity in recent years [Katharopoulos et al., 2020, Irie et al., 2021, Sun et al., 2023, 2024, Behrouz et al., 2025e]. In contrast to Transformers with their growing memory, modern RNNs employ a fixed-size memory (a.k.a. a hidden state) to compress the context, so that the decoding cost per token is constant and the overall computational cost is linear or sub-quadratic in the sequence length. This efficiency,

*Equal contribution.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

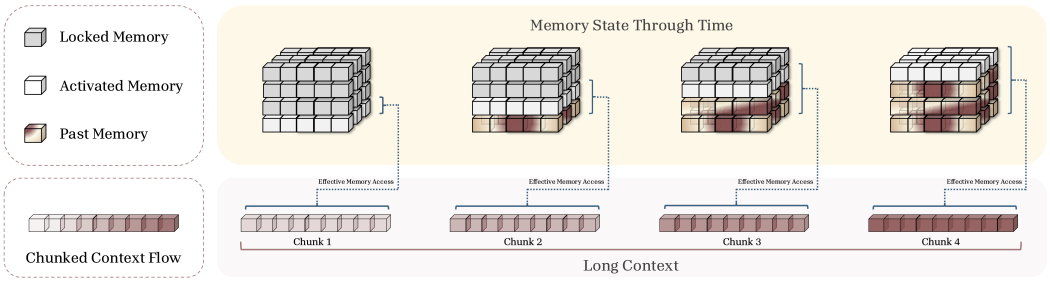


Figure 1: **Incremental memory activation in PROTEUS.** PROTEUS progressively expands the active subset of the memory state as the context advances. At each step, the model both reads from and writes to the currently *activated* blocks (together with previously activated "past" blocks), while the remaining blocks stay *locked* and do not participate in retrieval or updates. As the context grows, additional blocks are unlocked, providing fresh capacity for later tokens. This imposes an early capacity bottleneck that encourages compression of the initial context, while the fresh capacity introduced over time reduces overwriting and interference for later tokens.

however, comes at a price: compressing an unbounded context into a fixed-size state limits the length extrapolation ability of RNNs—their ability to generalize to sequence lengths beyond those seen in training—and thus their potential for long-context understanding [Bai et al., 2023, Kuratov et al., 2024, Hsieh et al., 2024, Tiezzi et al., 2024].

A substantial body of work has sought to close this gap through (meta-learning) memory initialization [Sun et al., 2024, Behrouz et al., 2025e], more robust update rules [Schlag et al., 2021, Von Oswald et al., 2023, Behrouz et al., 2025a], and more powerful memory architectures [Sun et al., 2024, Behrouz et al., 2025e, Zhang et al., 2025]. Despite these advances, recurrent models still struggle to make full use of their memory capacity, and we argue that a key reason is *when*, not just how, information is written. Because the memory is written online, early tokens arrive when the state is nearly empty: they face little competition for capacity and are under almost no pressure to compress, and so they are free to occupy a disproportionate share of the memory's degrees of freedom. Later tokens, by contrast, must be squeezed into whatever capacity remains, and writing them increasingly overwrites or interferes with what is already stored. The result is an imbalance in which the memory is biased toward initial tokens [Barbero et al., 2024, Behrouz et al., 2025b] and struggles to incorporate later context—precisely the regime that long-context tasks stress most.

We study a paradigm of *incremental memory activation* for recurrent neural networks, in which the effective memory capacity is scheduled as a function of context length. The design directly targets the imbalance described above. Since the problem originates with early tokens being under-compressed, we deliberately impose a capacity bottleneck early in the sequence, forcing the model to summarize the initial context rather than memorize it; and since later tokens suffer from interference, we progressively unlock fresh, previously unused capacity as the context grows, giving new information room to be written without overwriting what came before. These mechanisms therefore address the two failure modes of static memory in tandem. We then instantiate this paradigm with a simple yet effective mechanism, PROTEUS, that linearly unlocks memory and can be integrated into modern deep and linear-memory recurrent models, as well as MLP blocks, at no additional cost.

**Contributions.** In summary, our key contributions in this paper are as follows.

- **Incremental memory activation.** We introduce incremental memory activation, a paradigm for controlling *effective* model capacity (e.g., memory or parameters) over the context flow via progressive activation, encouraging early compression while reducing later overwriting and interference.
- **PROTEUS: a general, lightweight mechanism.** We propose PROTEUS, a simple block-wise gating scheme that applies incremental activation to both memory reads and writes and can be dropped into a broad class of associative-memory architectures.
- **Strong results.** We show that PROTEUS consistently improves state-of-the-art models (SWLA, Comba, Titans, Hope-Attention) across model scales on a broad range of benchmarks, without adding parameters or memory.

## 2 Preliminaries

### 2.1 Sequence Modeling as Associative Memory

A growing line of work views sequence models through the lens of an *associative memory system* that compresses past context into a compact state via an optimization procedure [Sun et al., 2024, Behrouz et al., 2025e,c]. We adopt the general formulation of Behrouz et al. [2025c].

**Definition 2.1** (Associative Memory)**.** *Given a set of keys $\mathcal{K} \subseteq \mathbb{R}^{d_k}$ and values $\mathcal{V} \subseteq \mathbb{R}^{d_v}$, an associative memory is an operator $\mathcal{M}(\cdot)$, parameterized by a set of memory parameters, that maps the keys $\mathcal{K}$ to values in $\mathcal{V}$. To learn such a mapping from data, an internal objective $\tilde{\mathcal{L}}(\cdot;\cdot)$ measures the quality of the mapping, and $\mathcal{M}$ can be computed by solving:*

$$\mathcal{M}^* = \arg\min_{\mathcal{M}} \; \tilde{\mathcal{L}}\big(\mathcal{M}(\mathcal{K}), \mathcal{V}\big). \tag{1}$$

In the sequence modeling setting, $\mathcal{K}$ and $\mathcal{V}$ are constructed from different representation views of the inputs, typically linear projections $k_t = x_t W_k$, $v_t = x_t W_v$, and $q_t = x_t W_q$ [Sun et al., 2024, Behrouz et al., 2025e]. We use the tilde in $\tilde{\mathcal{L}}$ to distinguish this *internal* memory objective—which the memory minimizes to learn its key–value associations—from the outer training loss of the full model [Behrouz et al., 2025c]. The power of this view is that a single choice of internal objective and optimization procedure induces a specific architecture: different sequence models are not fundamentally different mechanisms, but different instantiations of Equation (1) [Behrouz et al., 2025d].

**A spectrum of compression.** This framing situates existing sequence models along a spectrum defined by how aggressively they compress context. At one extreme, softmax attention can be cast as the *non-parametric* solution to an $\ell_2$ regression objective over the context, storing every key–value pair without compression and thus acting as a lossless but linearly growing memory; restricting that objective to a local window recovers sliding-window attention [Behrouz et al., 2025d]. At the other extreme, a fixed-size recurrent memory compresses the entire history into a state of constant size, trading expressivity for efficiency. Incremental memory activation, introduced in Section 3, can be seen as modulating *where on this spectrum* the effective memory sits as the context grows.

### 2.2 Memory as Online Optimization

The memory in Definition 2.1 is learned *online*, one token at a time, as the context is revealed. Concretely, the recurrence of a memory-based model can be written as an update rule that writes each new input into the state, and a read rule that produces a memory-conditioned output. A common instantiation optimizes the internal objective by gradient descent, giving the update

$$\mathcal{M}_t = \mathcal{M}_{t-1} - \theta_t \, \nabla \tilde{\mathcal{L}}(\mathcal{M}_{t-1}; k_t, v_t), \tag{2}$$

where $\theta_t$ is a step size, together with a read $y_t = \mathrm{Read}(\mathcal{M}_{t-1}, q_t)$ that depends on the memory architecture (e.g., a matrix–vector product for a linear memory, or a forward pass for an MLP memory). The gradient $\nabla \tilde{\mathcal{L}}(\mathcal{M}_{t-1}; k_t, v_t)$ acts as a *surprise* signal: the more the current association departs from what the memory has captured, the larger the update [Behrouz et al., 2025e].

**Architectures as choices of objective and update.** Within this frame, distinct architectures correspond to distinct choices of the internal objective $\tilde{\mathcal{L}}$, the optimizer (including retention and momentum), and the memory parameterization [Behrouz et al., 2025c,d].

*Hebbian rule.* Taking a dot-product objective $\tilde{\mathcal{L}}(\mathcal{M}_{t-1}; k_t, v_t) = -\langle \mathcal{M}_{t-1}^\top k_t, v_t \rangle$ and optimizing by gradient descent with a decay term recovers linear attention [Katharopoulos et al., 2020]:

$$\mathcal{M}_t = \alpha_t \, \mathcal{M}_{t-1} + \theta_t \, k_t v_t^\top. \tag{3}$$

The additive write has limited capacity, since it never removes stale associations before writing new ones.

*Delta rule.* Replacing the objective with an $\ell_2$ reconstruction loss $\tilde{\mathcal{L}}(\mathcal{M}_{t-1}; k_t, v_t) = \frac{1}{2}\|\mathcal{M}_{t-1}^\top k_t - v_t\|_2^2$ yields a delta-rule update that first erases the current value along $k_t$ before writing $v_t$ [Schlag et al., 2021]:

$$\mathcal{M}_t = \big(I - \theta_t \, k_t k_t^\top\big)\mathcal{M}_{t-1} + \theta_t \, k_t v_t^\top. \tag{4}$$

*Momentum and forgetting (Titans).* Building on the $\ell_2$ objective, Titans augments the update with momentum and an adaptive forget gate [Behrouz et al., 2025e]:

$$\mathcal{M}_t = (1 - \alpha_t)\, \mathcal{M}_{t-1} + S_t, \qquad S_t = \eta_t\, S_{t-1} - \theta_t\, \nabla \tilde{\mathcal{L}}(\mathcal{M}_{t-1}; k_t, v_t), \tag{5}$$

where $S_t$ carries a running memory of past surprise (momentum) and $\alpha_t \in [0, 1]$ controls how much of the state is forgotten. TTT [Sun et al., 2024] corresponds to the same $\ell_2$ objective with plain gradient descent ($\eta_t = \alpha_t = 0$) and admits a linear or an MLP memory.

**Effective capacity.** We refer to the *effective capacity* of the memory at step $t$ as the number of memory parameters that are active—i.e., that participate in the update in Equation (2) and in the read. We write $C$ for the total capacity, and in Section 3 we schedule an active subset $c(t) \leq C$ over the context. This axis is deliberately *orthogonal* to the dimensions discussed in general frameworks [Behrouz et al., 2025c,d, Wang et al., 2025]: incremental activation changes neither the internal objective, nor the optimizer, nor the memory architecture, but only which parameters are exposed at each position. This orthogonality is precisely what lets a single mechanism apply, unchanged, across the entire family.

### 2.3 MLP Blocks as Associative Memory

The associative-memory view above concerns the online memory state, but a recent perspective—Nested Learning [Behrouz et al., 2025d]—extends the same lens to a model's *parameters*. Under this view, gradient-based update of MLP blocks is itself a form of associative memory: each layer, updated by backpropagation, learns to map its input to the local error signal it receives. Concretely, a gradient-descent update

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \eta_{i+1}\, \boldsymbol{e}_i, \tag{6}$$

where $\boldsymbol{e}_i$ is the error (surprise) supplied by the optimizer for the $i$-th data sample, can be read as a memory that writes each sample's contribution into $\boldsymbol{\theta}$. We do not rely on the full machinery of this perspective; what matters for us is the shared structure. This motivates applying the same incremental-activation principle not only to the recurrent state but to the MLP parameters themselves, which we develop for architectures such as Hope-Attention in Section 4.

### 2.4 Static Capacity is Suboptimal

Despite their variety, the memory and parameter views above share a design choice that is rarely questioned: capacity is *static*, with the same parameters exposed throughout the entire context or training run. Yet controlling a model's effective capacity has long been studied [Guyon et al., 1991, Hansen and Yu, 2001, Ding et al., 2018, Kaplan et al., 2020], and from a compression standpoint, a bottlenecked information flow encourages a model to retain only task-relevant structure, as exemplified by autoencoders [Vincent et al., 2010]. Classical capacity control restricts a model's degrees of freedom *globally*; the online setting of Equation (2) suggests a positional analogue, since the right amount of capacity differs early versus late.

The failure mode of static capacity can be read directly off the online objective. When all parameters of $\mathcal{M}$ are active from the first token, early inputs are fit using the memory's full degrees of freedom; minimizing $\tilde{\mathcal{L}}$ then favors memorizing them rather than compressing them, so the memory is "polluted" by early context and generalizes poorly. Once the state is heavily written, updating it for later tokens increasingly interferes with what is already stored, making new information harder to incorporate. In the following, we introduce *incremental memory activation* for controlling effective capacity over the context in Section 3, instantiate it in PROTEUS in Section 4, and incorporate it into a broad class of sequence models in Section 5.

## 3 Incremental Memory Activation

Section 2 identified two failure modes of static capacity: early inputs are memorized rather than compressed, and later inputs interfere with an already-saturated state. We now introduce a paradigm that addresses both by scheduling the *effective* capacity of the memory as a function of input position. The design follows two complementary goals: (i) **early compression**, which forces the model to

summarize early context through a restricted-capacity bottleneck; and (ii) **fresh capacity**, which unlocks previously unused components later in the context to absorb new information without overwriting what has already been stored.

**Why a growing schedule.** Before formalizing the paradigm, it is worth asking what kind of schedule the two goals call for. The tension is between too little and too much active capacity. Too little active capacity late in the context wastes tokens the memory could have used, while too much active capacity early lets the memory overfit the few tokens seen so far instead of compressing them. Balancing the two suggests that the active capacity should track the amount of context observed: small when few tokens have been seen, and growing as more arrive, rather than being fixed at either extreme. This intuition, that effective capacity should grow with position, motivates the simple uniform schedule we adopt in Section 4, whose behavior we study empirically in Section 5.

**Capacity-scheduled memory.** To capture this formally, we augment the associative memory of Definition 2.1 with an *activation operator* $\mathcal{G}_t$ that exposes only a subset of the memory's capacity at step $t$. Writing the online update of Equation (2) with this operator, incremental activation restricts both the write and the read to the active subset while holding the inactive components fixed.

**Definition 3.1** (Capacity-Scheduled Associative Memory)**.** *Let $\mathcal{G}_t$ be an activation operator that selects a subset of the memory parameters at step $t$, with $g_t \in \{0,1\}^{\dim(\mathcal{M})}$ the indicator of the selected components, and write $\mathcal{M}_{t-1}^{(g)} := \mathcal{G}_t(\mathcal{M}_{t-1})$ for the active subset. A capacity-scheduled associative memory updates online as*

$$\begin{aligned} \mathcal{M}_t &= \mathcal{M}_{t-1} - \theta_t \, \mathcal{G}_t\big(\nabla \tilde{\mathcal{L}}(\mathcal{M}_{t-1}^{(g)}; k_t, v_t)\big), \\ &\textit{so that} \quad (\mathbf{1} - g_t) \odot \mathcal{M}_t = (\mathbf{1} - g_t) \odot \mathcal{M}_{t-1}, \end{aligned} \tag{7}$$

*and reads from the active subset, $y_t = \mathrm{Read}(\mathcal{M}_{t-1}^{(g)}, q_t)$.*

The constraint states the defining property of the paradigm: components outside the active set at step $t$ are *locked*, neither written nor read, and retain exactly the values they held before. Setting $\mathcal{G}_t$ to the identity for all $t$ recovers the standard online memory of Section 2. Importantly, this is a single online trajectory under a *growing* activation set, not a fresh optimization at each step: early tokens are compressed into the components that were active when they arrived, and that compressed state persists as further components are unlocked.

This formulation supports the two goals directly. For most of the context the active set is a strict subset of the memory, which limits effective capacity and forces compression through an explicit bottleneck. As additional components are unlocked, they supply fresh degrees of freedom that absorb later inputs, reducing interference with what earlier tokens have already written.

**What "capacity" means, and what it does not.** It helps to fix a concrete picture of the active subset before the specific schedule of Section 4. For a matrix-valued memory $\mathcal{M} \in \mathbb{R}^{d_k \times d_v}$, the natural unit of capacity is a contiguous block of coordinates (e.g., a group of rows or columns), and activating a subset means exposing only some of these blocks to reads and writes while the rest stay locked. Two distinctions are worth drawing. First, the total memory is *fixed*: we schedule which parts of a constant-size state are active, rather than growing the state itself, which sets our paradigm apart from approaches whose memory expands with sequence length (Appendix A). Second, activation is monotone over time: locked components are *unlocked* as the context grows, rather than being permanently removed as in pruning, so no capacity is discarded and all of it becomes available by the end.

**Generality.** Although we have stated Definition 3.1 for the recurrent memory state, the activation operator $\mathcal{G}_t$ acts on nothing more specific than a set of capacity-bearing parameters. The same principle therefore applies wherever capacity is compressed into parameters over a context flow, including, through the Nested Learning view of Section 2.3, the parameters of the model itself. This lets a single paradigm cover both the online memory (Section 4.1) and MLP blocks (Section 4.2). Incremental activation can be instantiated in many ways, differing in how $\mathcal{G}_t$ maps position to an active subset; in this work we study a simple, deterministic schedule and leave richer, data-dependent activation policies to future work.

# 4 PROTEUS

We now introduce PROTEUS, our concrete realization of incremental memory activation (Definition 3.1). We first present it for *memory-based* recurrent sequence models, where an explicit associative memory state is updated online over the sequence. We then note that, through the Nested Learning view of Section 2.3, the same principle applies to a model's parameters, letting PROTEUS extend to architectures such as Hope-Attention.

## 4.1 PROTEUS for Memory

PROTEUS instantiates the capacity-scheduled memory of Definition 3.1 with a simple block-wise activation operator. It requires two ingredients: a **partition** of the memory into blocks, and a deterministic **schedule** that activates them over the context. We describe each, then give the resulting gated update and read.

**Block partition and gating operator.** We partition the memory parameters into $E$ equal-sized contiguous blocks. Recall from Definition 3.1 that the activation operator $\mathcal{G}_t$ is realized by an elementwise mask $g_t \in \{0,1\}^{\dim(\mathcal{M})}$; for PROTEUS, this mask enables exactly the parameters belonging to the currently active blocks, so that

$$\mathcal{G}_t(\mathcal{M}) \coloneqq g_t \odot \mathcal{M}, \qquad \text{and in particular} \qquad \mathcal{M}_{t-1}^{(g)} = \mathcal{G}_t(\mathcal{M}_{t-1}) = g_t \odot \mathcal{M}_{t-1}, \tag{8}$$

where $\odot$ is elementwise multiplication. Inactive blocks are masked out (locked) and participate in neither retrieval nor learning. Because most modern neural memories parameterize $\mathcal{M}$ as a linear map or an MLP, this multiplicative gating is all that is needed to control effective capacity, at no additional cost.

**Gated update and read.** Recall the generic online update and read of Equation (2). PROTEUS instantiates the capacity-scheduled update of Definition 3.1 by applying the gradient only to active components, leaving locked blocks unchanged:

$$\mathcal{M}_t = \mathcal{M}_{t-1} - \theta_t\big(g_t \odot \nabla\tilde{\mathcal{L}}(\mathcal{M}_{t-1}^{(g)}; k_t, v_t)\big), \qquad \mathcal{M}_{t-1}^{(g)} = \mathcal{G}_t(\mathcal{M}_{t-1}). \tag{9}$$

Equivalently, PROTEUS updates the gated state and embeds it back into the full memory with locked components preserved,

$$\mathcal{M}_t = \mathcal{M}_{t-1} + \mathcal{G}_t\big(\delta\mathcal{M}_t\big), \qquad \delta\mathcal{M}_t \coloneqq -\theta_t \nabla\tilde{\mathcal{L}}(\mathcal{M}_{t-1}^{(g)}; k_t, v_t). \tag{10}$$

Retrieval is gated the same way, reading only from the active subspace:

$$y_t = \mathrm{Read}\big(\mathcal{M}_{t-1}^{(g)}, q_t\big) = \mathrm{Read}\big(\mathcal{G}_t(\mathcal{M}_{t-1}), q_t\big). \tag{11}$$

Both writes and reads are thus confined to the active blocks, while locked blocks remain untouched until activated. Although we write the update for gradient descent, the same gating applies unchanged to any update rule $\mathrm{Upd}(\cdot)$; only the active components are ever modified.

**Memory expansion schedule.** It remains to specify how position $t$ maps to a set of active blocks. We adopt a simple deterministic schedule that increases the number of active blocks uniformly across the context. Let $N$ be the maximum context length and $E$ the number of blocks, and assume $\dim(\mathcal{M})$ is divisible by $E$ (otherwise the last block absorbs the remainder). With block size $d \coloneqq \dim(\mathcal{M})/E$, define the step length

$$\Delta \coloneqq \max\Big(1, \big\lfloor \tfrac{N}{E} \big\rfloor\Big). \tag{12}$$

At step $t \in \{1, \dots, N\}$ we activate the first $k(t) \coloneqq \min\big(E,\ 1 + \big\lfloor \frac{t-1}{\Delta} \big\rfloor\big)$ blocks, so the active fraction grows in $E$ uniform increments. The mask $g_t$ is the corresponding prefix mask:

$$g_t[j] = \begin{cases} 1, & 1 \le j \le k(t)\, d, \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

Thus tokens in $[1, \Delta]$ read and write only the first $1/E$ of the memory, tokens in $[\Delta{+}1, 2\Delta]$ the first $2/E$, and so on, until the full memory is available after the final expansion step (Figure 1). We set $N$ to the training context length (8K) and keep this schedule fixed at evaluation; beyond $N$ the full memory is active, so PROTEUS adds no gating at inference lengths past the training window.

Table 1: **Language modeling and commonsense reasoning results.** We report perplexity on Wikitext and LAMBADA, and zero-shot accuracy on LAMBADA and seven commonsense benchmarks; Avg. denotes the mean accuracy across these eight accuracy columns. Results are shown for 760M (50B tokens) and 1.3B (100B tokens) models, comparing each backbone with and without PROTEUS (shaded rows). Best average per group in **bold**.

| Model | Wiki. ppl ↓ | LMB. ppl ↓ | LMB. acc ↑ | PIQA acc ↑ | Hella. acc_n ↑ | Wino. acc ↑ | ARC-e acc ↑ | ARC-c acc_n ↑ | SIQA acc ↑ | BoolQ acc ↑ | Avg. ↑ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 760M params / 50B tokens | | | | | | | | | | | |
| Transformer++ | 22.08 | 22.41 | 38.2 | 68.9 | 44.1 | 56.8 | 67.0 | 34.9 | 40.2 | 62.4 | 51.56 |
| Hope-Attention | 20.67 | 21.36 | 39.5 | 70.4 | 50.1 | 56.3 | 66.9 | 37.5 | 40.7 | 63.8 | 53.15 |
| +PROTEUS | 19.87 | 19.72 | 40.5 | 70.2 | 51.9 | 58.2 | 67.3 | 38.5 | 41.4 | 63.9 | **53.99** |
| RetNet | 23.54 | 23.87 | 35.7 | 66.4 | 42.8 | 53.6 | 64.8 | 33.1 | 38.7 | 57.4 | 49.06 |
| SWLA ($c = 2$) | 22.76 | 22.85 | 36.7 | 67.5 | 44.2 | 54.7 | 64.6 | 34.3 | 39.9 | 59.1 | 50.12 |
| +PROTEUS | 21.82 | 20.88 | 37.5 | 67.1 | 44.9 | 54.7 | 66.0 | 35.8 | 41.0 | 60.2 | **50.90** |
| DeltaNet | 22.89 | 23.13 | 37.2 | 68.0 | 45.2 | 52.9 | 65.4 | 33.4 | 40.2 | 59.8 | 50.26 |
| Comba | 21.94 | 21.77 | 38.4 | 67.1 | 47.1 | 53.0 | 65.9 | 35.5 | 40.8 | 63.7 | 51.43 |
| +PROTEUS | 21.32 | 21.20 | 39.1 | 67.9 | 47.6 | 54.7 | 66.2 | 35.4 | 41.6 | 64.7 | **52.15** |
| Titans | 20.92 | 21.28 | 39.3 | 69.2 | 49.9 | 53.1 | 67.3 | 36.9 | 41.7 | 63.8 | 52.65 |
| +PROTEUS | 20.44 | 20.93 | 40.0 | 69.5 | 50.7 | 56.1 | 67.2 | 36.8 | 42.6 | 64.0 | **53.36** |
| 1.3B params / 100B tokens | | | | | | | | | | | |
| Transformer++ | 17.04 | 17.42 | 45.3 | 73.1 | 51.8 | 59.3 | 71.0 | 38.2 | 43.5 | 64.3 | 55.81 |
| Hope-Attention | 15.66 | 13.39 | 48.2 | 72.9 | 55.0 | 59.8 | 72.2 | 38.7 | 43.3 | 65.4 | 56.93 |
| +PROTEUS | 15.70 | 13.01 | 48.5 | 73.1 | 55.4 | 60.8 | 73.1 | 38.5 | 44.2 | 64.8 | **57.30** |
| RetNet | 18.84 | 17.29 | 40.8 | 70.6 | 48.9 | 56.0 | 67.9 | 35.3 | 41.6 | 62.7 | 52.98 |
| SWLA ($c = 2$) | 18.11 | 16.95 | 40.7 | 71.4 | 49.2 | 57.3 | 68.8 | 37.0 | 42.4 | 62.6 | 53.67 |
| +PROTEUS | 18.07 | 17.18 | 40.6 | 71.9 | 50.5 | 57.8 | 69.4 | 37.1 | 43.2 | 63.1 | **54.20** |
| DeltaNet | 17.39 | 17.02 | 40.0 | 71.4 | 50.1 | 53.9 | 68.3 | 36.6 | 43.3 | 60.7 | 53.04 |
| Comba | 16.98 | 14.17 | 43.8 | 73.2 | 53.5 | 60.1 | 70.9 | 39.2 | 44.2 | 58.5 | 55.42 |
| +PROTEUS | 16.95 | 14.01 | 44.6 | 73.1 | 54.2 | 60.4 | 71.4 | 39.5 | 44.6 | 60.2 | **56.00** |
| Titans | 15.36 | 13.18 | 50.9 | 74.0 | 54.6 | 57.3 | 72.2 | 40.9 | 42.6 | 63.1 | 56.95 |
| +PROTEUS | 14.94 | 13.03 | 51.7 | 75.9 | 54.6 | 60.1 | 72.3 | 41.7 | 43.9 | 63.8 | **58.00** |

### 4.2 Extension to Hope and MLP Blocks

The same scheduling principle is not specific to the recurrent state. As set up in Section 2.3, the Nested Learning view [Behrouz et al., 2025d] casts MLP blocks as instances of associative memory with arbitrary objective (e.g., next-token prediction in language modeling) that compresses data into an MLP's parameters. Following the parallel drawn there, an over-parameterized MLP early in training has ample capacity to fit its first samples without compressing them; scheduling which parameters are active over the course of training applies the same early-bottleneck, fresh-capacity idea to the parameters themselves. We treat this as a lightweight extension rather than a separate contribution, and use it to reach architectures beyond recurrent memory.

**Gated parameter updates.** Recall the generic parametric update $\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \eta_{i+1}\boldsymbol{e}_i$ from Equation (6), where $\boldsymbol{e}_i$ is the error (surprise) supplied by the optimizer for the $i$-th sample. PROTEUS gates this update with a schedule mask $g_{i+1}$ over the MLP-block parameters $\{\boldsymbol{\theta}_i\}$:

$$\boldsymbol{\theta}_{i+1} = \boldsymbol{\theta}_i - \eta_{i+1}\left(g_{i+1} \odot \boldsymbol{e}_i\right), \tag{14}$$

so that only unmasked parameters receive a gradient update for each sample. For a concrete optimizer such as AdamW, this becomes

$$\boldsymbol{\theta}_{i+1} = \alpha_{i+1}\,\boldsymbol{\theta}_i - \eta_{i+1}\,\frac{\hat{\boldsymbol{m}}_{i+1}}{\sqrt{\hat{\boldsymbol{v}}_{i+1}} + \varepsilon}, \tag{15}$$

with $\hat{\boldsymbol{m}}_{i+1} = \boldsymbol{m}_{i+1}/(1-\beta_1^{i+1})$, $\hat{\boldsymbol{v}}_{i+1} = \boldsymbol{v}_{i+1}/(1-\beta_2^{i+1})$, decoupled weight decay $\alpha_{i+1}$, and gated moment updates

$$\boldsymbol{m}_{i+1} = \beta_1\,\boldsymbol{m}_i + (1-\beta_1)\big(g_{i+1} \odot \nabla_{\boldsymbol{\theta}_i} L(\boldsymbol{\theta}_i, \boldsymbol{x}_{i+1})\big),$$
$$\boldsymbol{v}_{i+1} = \beta_2\,\boldsymbol{v}_i + (1-\beta_2)\big(g_{i+1} \odot \nabla_{\boldsymbol{\theta}_i} L(\boldsymbol{\theta}_i, \boldsymbol{x}_{i+1})\big)^2.$$

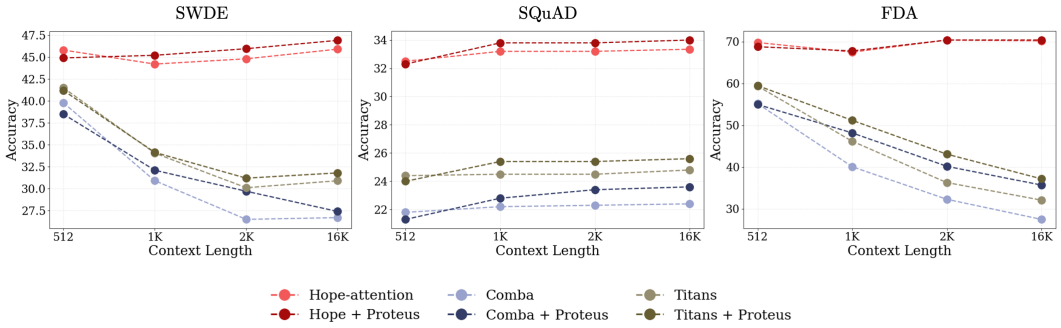


Figure 2: **Long-context retrieval.** Accuracy vs. context length on SWDE [Lockard et al., 2019], SQuAD [Rajpurkar et al., 2016], and FDA [Arora et al., 2024] for Hope-Attention, Comba, and Titans. PROTEUS consistently improves robustness at longer contexts, mitigating degradation (notably for Comba and Titans) and improving Hope-Attention across lengths.

Only the parameters selected by $g_{i+1}$ are updated; scheduling $\{g_i\}$ over training then supplies fresh capacity for later data while avoiding full over-parameterization early on.

**Hope-Attention.** As a proof of concept, we apply PROTEUS to the Hope architecture [Behrouz et al., 2025d], whose backbone resembles a Transformer but replaces the single post-attention MLP with a chain of MLP blocks updated at different frequencies. Given per-block update frequencies $f_\ell$ and input $\boldsymbol{x}$, its output is

$$\boldsymbol{k}_t = \boldsymbol{x}_t W_{\boldsymbol{k}}, \qquad \boldsymbol{v}_t = \boldsymbol{x}_t W_{\boldsymbol{v}}, \qquad \boldsymbol{q}_t = \boldsymbol{x}_t W_{\boldsymbol{q}}, \tag{16}$$

$$\boldsymbol{h}_t = \texttt{Attn}\big(\{\boldsymbol{k}_i\}_{i=1}^{t}, \{\boldsymbol{v}_i\}_{i=1}^{t}, \boldsymbol{q}_t\big), \tag{17}$$

$$\boldsymbol{y}_t = \texttt{MLP}^{(f_L)}\Big(\texttt{MLP}^{(f_{L-1})}\big(\cdots \texttt{MLP}^{(f_1)}(\boldsymbol{h}_t)\big)\Big), \tag{18}$$

where the $\ell$-th block's parameters are updated at frequency $f_\ell$, i.e., every $C^{(\ell)}$ steps. Applying PROTEUS here amounts to gating these MLP-block updates with the schedule mask of Equation (14), so that each block's parameters are activated progressively over training. For language modeling, as a common standard, we use next-token prediction with AdamW throughout.

## 5 Experiments

**Experimental Setup.** We apply our framework to four families of high-performing models: Hope-Attention [Behrouz et al., 2025d], Sliding-Window Linear Attention (SWLA) [Behrouz et al., 2025a], Comba [Hu et al., 2025], and Titans [Behrouz et al., 2025e]. Unless otherwise noted, we set the number of partition blocks in PROTEUS to $E = 16$; we ablate this choice in Appendix B. For reference, we also report results for Transformer++ [Touvron et al., 2023], RetNet [Sun et al., 2023], and DeltaNet [Schlag et al., 2021]. All models are trained on FineWeb [Penedo et al., 2024] with an 8K training context window. We consider two parameter scales (760M and 1.3B), training on 50B and 100B tokens respectively, and report perplexity on held-out Wikitext and LAMBADA. We use AdamW with a learning rate of $4 \times 10^{-4}$ and a cosine annealing schedule, a batch size of $0.5$M tokens, and weight decay of $0.1$. Downstream, we evaluate zero-shot on Wikitext [Merity et al., 2017], LAMBADA (LMB) [Paperno et al., 2016], PIQA [Bisk et al., 2020], HellaSwag [Zellers et al., 2019], WinoGrande [Sakaguchi et al., 2021], ARC-Easy (ARC-e) and ARC-Challenge (ARC-c) [Clark et al., 2018], SIQA [Sap et al., 2019], and BoolQ [Clark et al., 2019].

### 5.1 Language Modeling and Commonsense Reasoning

Results for PROTEUS variants of the four backbones, together with additional baselines, are summarized in Table 1 at two scales. Across all four families and both scales, adding PROTEUS consistently improves average downstream accuracy over the corresponding base model, and lowers perplexity in nearly all settings.

Table 2: **Needle-In-A-Haystack (NIAH) results.** Single-needle tasks at three levels of difficulty—S-NIAH-1 (pass-key retrieval), S-NIAH-2 (numerical needle), and S-NIAH-3 (UUID-based needle)—together with multi-key (MK), multi-query (MQ), and multi-value (MV) settings, at context lengths 4K, 8K, and 16K. Each backbone is shown with and without PROTEUS (shaded rows).

| | **S-NIAH-1** (pass-key retrieval) | | | **S-NIAH-2** (number in haystack) | | | **S-NIAH-3** (uuid in haystack) | | |
|---|---|---|---|---|---|---|---|---|---|
| **Model** | 4K | 8K | 16K | 4K | 8K | 16K | 4K | 8K | 16K |
| Transformer | 97.8 | 86.2 | 83.8 | 100 | 99.2 | 96.4 | 82.2 | 72.0 | 48.4 |
| Hope-Attention | 100 | 100 | 100 | 100 | 99.4 | 96.8 | 83.4 | 72.8 | 50.0 |
| +PROTEUS | 100 | 100 | 100 | 100 | 99.4 | 97.2 | 83.2 | 73.2 | 54.4 |
| Comba | 100 | 100 | 99.4 | 92.6 | 47.2 | 13.4 | 62.4 | 13.8 | 7.4 |
| +PROTEUS | 100 | 100 | 99.2 | 91.8 | 49.4 | 21.2 | 60.8 | 18.2 | 10.8 |
| Titans | 100 | 100 | 100 | 99.6 | 84.6 | 69.4 | 74.2 | 42.8 | 21.4 |
| +PROTEUS | 100 | 100 | 100 | 99.2 | 85.2 | 74.2 | 74.2 | 44.0 | 29.8 |

| | **MK-NIAH-1** (multi-key line retrieval) | | | **MQ-NIAH** (multi-query) | | | **MV-NIAH** (multi-value) | | |
|---|---|---|---|---|---|---|---|---|---|
| **Model** | 4K | 8K | 16K | 4K | 8K | 16K | 4K | 8K | 16K |
| Transformer | 80.0 | 79.4 | 60.8 | 57.4 | 46.2 | 28.8 | 36.6 | 33.8 | 20.2 |
| Hope-Attention | 81.4 | 85.2 | 62.2 | 62.0 | 47.6 | 30.6 | 37.2 | 36.2 | 23.0 |
| +PROTEUS | 80.8 | 85.2 | 64.8 | 62.2 | 48.4 | 34.2 | 36.6 | 36.0 | 25.8 |
| Comba | 22.8 | 20.4 | 10.4 | 22.2 | 16.0 | 6.2 | 15.8 | 14.2 | 6.6 |
| +PROTEUS | 23.4 | 20.2 | 14.2 | 23.8 | 16.6 | 8.4 | 14.8 | 14.0 | 8.2 |
| Titans | 27.0 | 22.8 | 11.8 | 23.2 | 18.4 | 10.2 | 24.8 | 15.4 | 8.8 |
| +PROTEUS | 28.4 | 22.6 | 16.8 | 23.2 | 18.6 | 12.0 | 24.6 | 14.8 | 10.2 |

At 760M, PROTEUS improves the average commonsense score for every backbone (Hope-Attention: $53.15 \rightarrow 53.99$; SWLA: $50.12 \rightarrow 50.90$; Comba: $51.43 \rightarrow 52.15$; Titans: $52.65 \rightarrow 53.36$), while also reducing perplexity in each case. The best overall 760M result is achieved by Hope-Attention+PROTEUS, with the lowest perplexities (Wiki. 19.87, LMB. 19.72) and the highest average accuracy (53.99). At 1.3B, the gains persist and are most pronounced for the strongest memory model: Titans+PROTEUS attains the best overall performance in the table, with the lowest Wikitext perplexity (14.94) and the highest average accuracy (58.00). These improvements hold even against strong attention-based baselines such as Transformer++, indicating that the benefit is not tied to a single architecture but to how memory is managed over the context.

The gains are consistent with PROTEUS's two intended effects (Section 4): the early capacity bottleneck yields cleaner compression, which helps even at the 8K training context, while the fresh capacity unlocked later aids retention as context grows. We examine the latter directly in the long-context experiments below.

### 5.2 Needle-in-a-Haystack

We evaluate long-context retrieval on Needle-in-a-Haystack (NIAH); results are in Table 2. The pattern matches our thesis: PROTEUS's gains are concentrated at longer contexts and on harder variants, while at short contexts where the base model already saturates (e.g., S-NIAH-1, near-perfect for all models), PROTEUS is essentially neutral. Where the base model degrades, the improvements are substantial. At 16K, PROTEUS improves Titans on S-NIAH-3 ($21.4 \rightarrow 29.8$) and S-NIAH-2 ($69.4 \rightarrow 74.2$), and Comba on S-NIAH-2 ($13.4 \rightarrow 21.2$). On multi-needle benchmarks at 16K, it improves MK-NIAH-1 (Titans $11.8 \rightarrow 16.8$), MQ-NIAH (Hope-Attention $30.6 \rightarrow 34.2$), and MV-NIAH (Hope-Attention $23.0 \rightarrow 25.8$). Across the table, the largest gains appear precisely where the context is longest and the base model weakest. Note that the schedule completes within the 8K training window, so at 16K no further blocks are unlocked: the improvement there reflects the cleaner, better-compressed state built under the early bottleneck, which degrades more gracefully as the context extends beyond the training length.

Table 3: **LongBench evaluation.** Performance on LongBench [Bai et al., 2023] across six long-context tasks (Narrative, Qasper, MultiField, Hotpot, 2WikiMulti, Musique), reported for each backbone with and without PROTEUS (shaded rows); Average denotes the mean across the six tasks. PROTEUS improves the average score for all three backbones.

| Model | Narrative | Qasper | MultiField | Hotpot | 2WikiMulti | Musique | Average |
|---|---|---|---|---|---|---|---|
| Hope-Attention | 12.1 | 9.4 | 19.2 | 20.6 | 26.7 | 6.3 | 15.72 |
| +PROTEUS | 12.6 | 10.2 | 20.9 | 21.5 | 27.3 | 7.4 | 16.65 |
| Comba | 7.6 | 10.4 | 15.9 | 14.6 | 23.1 | 6.7 | 13.05 |
| +PROTEUS | 7.8 | 10.8 | 16.2 | 14.5 | 23.2 | 6.9 | 13.23 |
| Titans | 8.0 | 10.3 | 17.4 | 15.6 | 24.8 | 6.7 | 13.80 |
| +PROTEUS | 8.4 | 10.4 | 17.9 | 16.4 | 25.1 | 6.7 | 14.15 |

### 5.3 Long Context Understanding

We now evaluate PROTEUS on long-context understanding, using two families of tasks: recall-intensive retrieval, and the LongBench benchmark.

**Retrieval Tasks.** Following recent studies [Hu et al., 2025, Yang et al., 2024a, Arora et al., 2024, Behrouz et al., 2025a], we evaluate PROTEUS on the Hope-Attention, Comba, and Titans baselines on recall-intensive tasks from Lockard et al. [2019], Rajpurkar et al. [2016], Arora et al. [2024]; results are in Figure 2. Across all datasets and baselines, PROTEUS is more robust at longer context lengths and improves length extrapolation. The effect is clearest for Comba and Titans, whose accuracy degrades sharply with context under the base model and is markedly stabilized by PROTEUS; for the already-strong Hope-Attention, PROTEUS improves accuracy across all lengths. Crucially, the gain over the baseline is largest at the longest contexts, matching the trend seen in the NIAH results.

**LongBench Benchmark.** Finally, we evaluate PROTEUS on LongBench [Bai et al., 2023]; results are in Table 3. PROTEUS improves the average score for all three backbones and improves the large majority of individual tasks, without adding parameters or memory. Together with the retrieval and NIAH results, this supports our central claim: treating memory capacity as static is suboptimal, and incremental activation is a simple, broadly applicable mechanism for long-context management.

## 6 Conclusion

We introduced *incremental memory activation*, a simple paradigm for long-context sequence modeling in which the *effective* capacity is progressively expanded over the context rather than exposed statically from the start. We instantiated this idea in PROTEUS, a lightweight, block-wise gating mechanism that restricts both reads and writes to an active subset of the memory and unlocks fresh capacity over time, and, through the Nested Learning view, applies the same principle to a model's parameters. Across four architectures (SWLA, Comba, Titans, and Hope-Attention), PROTEUS consistently improves language modeling and commonsense reasoning, and on needle-in-a-haystack, retrieval, and LongBench it improves long-context robustness and length extrapolation, with the largest gains appearing at the longest context lengths. That a single, architecture-agnostic mechanism helps across four distinct memory designs suggests the benefit stems from how capacity is allocated over the context, not from any one architecture: treating capacity as static is suboptimal, and scheduling it is a practical and broadly applicable tool for long-context computation.

**Limitations.** Our study has several limitations. First, the activation schedule is fixed and hand-designed: we grow the active set uniformly with position, and while our ablations show that finer partitions help, we do not characterize what schedule is optimal or how it should depend on the data or the memory's update rule. Second, the extension to MLP blocks is a proof of concept demonstrated on a single architecture, rather than a fully validated result. Third, PROTEUS's gains are concentrated at longer contexts and harder tasks; where the base model already saturates, the effect is essentially neutral, so the mechanism is most useful precisely when long-context capacity is the bottleneck. We discuss several directions for future work in Appendix C.

## Acknowledgments and Disclosure of Funding

This research was enabled in part by compute resources provided by Mila (mila.quebec), Calcul Québec (calculquebec.ca), and the Digital Research Alliance of Canada (alliancecan.ca). This work was also supported by the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korean government (MSIT) (No. RS-2024-00457882, National AI Research Lab Project).

## References

Simran Arora, Sabri Eyuboglu, Michael Zhang, Aman Timalsina, Silas Alberti, James Zou, Atri Rudra, and Christopher Re. Simple linear attention language models balance the recall-throughput tradeoff. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=e93ffDcpH3.

Sangmin Bae, Yujin Kim, Reza Bayat, Sungnyun Kim, Jiyoun Ha, Tal Schuster, Adam Fisch, Hrayr Harutyunyan, Ziwei Ji, Aaron Courville, et al. Mixture-of-recursions: Learning dynamic recursive depths for adaptive token-level computation. *Advances in Neural Information Processing Systems*, 38:96572–96617, 2025.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv: 2308.14508*, 2023.

Federico Barbero, Andrea Banino, Steven Kapturowski, Dharshan Kumaran, João Madeira Araújo, Oleksandr Vitvitskyi, Razvan Pascanu, and Petar Veličković. Transformers need glasses! information over-squashing in language tasks. *Advances in Neural Information Processing Systems*, 37: 98111–98142, 2024.

Reza Bayat, Mohammad Pezeshki, Elvis Dohmatob, David Lopez-Paz, and Pascal Vincent. The pitfalls of memorization: When memorization hurts generalization. *arXiv preprint arXiv:2412.07684*, 2024.

Reza Bayat, Ali Behrouz, and Aaron Courville. Tapered language models. *arXiv preprint arXiv: 2606.23670*, 2026.

Ali Behrouz, Zeman Li, Praneeth Kacham, Majid Daliri, Yuan Deng, Peilin Zhong, Meisam Razaviyayn, and Vahab Mirrokni. Atlas: Learning to optimally memorize the context at test time. *arXiv preprint arXiv:2505.23735*, 2025a.

Ali Behrouz, Ali Parviz, Mahdi Karami, Clayton Sanford, Bryan Perozzi, and Vahab Mirrokni. Best of both worlds: Advantages of hybrid graph sequence models. In *Forty-second International Conference on Machine Learning*, 2025b. URL https://openreview.net/forum?id=GJKe8WYHxq.

Ali Behrouz, Meisam Razaviyayn, Peilin Zhong, and Vahab Mirrokni. It's all connected: A journey through test-time memorization, attentional bias, retention, and online optimization. *arXiv preprint arXiv:2504.13173*, 2025c.

Ali Behrouz, Meisam Razaviyayn, Peilin Zhong, and Vahab Mirrokni. Nested learning: The illusion of deep learning architectures. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025d. URL https://openreview.net/forum?id=nbMeRvNb7A.

Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. Titans: Learning to memorize at test time. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025e. URL https://openreview.net/forum?id=8GjSf9Rh7Z.

Ali Behrouz, Zeman Li, Yuan Deng, Peilin Zhong, Meisam Razaviyayn, and Vahab Mirrokni. Memory caching: Rnns with growing memory. *arXiv preprint arXiv: 2602.24281*, 2026.

Emmanuel Bengio, Pierre-Luc Bacon, Joelle Pineau, and Doina Precup. Conditional computation in neural networks for faster models. *arXiv preprint arXiv:1511.06297*, 2015.

Alberto Bietti, Vivien Cabannes, Diane Bouchacourt, Herve Jegou, and Leon Bottou. Birth of a transformer: A memory viewpoint. *Advances in Neural Information Processing Systems*, 36, 2024.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7432–7439, 2020.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1300. URL https://aclanthology.org/N19-1300/.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Róbert Csordás, Christopher Potts, Christopher D Manning, and Atticus Geiger. Recurrent neural networks learn to store and generate sequences using non-linear representations. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 248–262, 2024.

Jie Ding, Vahid Tarokh, and Yuhong Yang. Model selection techniques: An overview. *IEEE Signal Processing Magazine*, 35(6):16–34, 2018.

Maha Elbayad, Jiatao Gu, Edouard Grave, and Michael Auli. Depth-adaptive transformer. *arXiv preprint arXiv:1910.10073*, 2019.

Xavier Gonzalez, Andrew Warrington, Jimmy Smith, and Scott Linderman. Towards scalable and stable parallelization of nonlinear rnns. *Advances in Neural Information Processing Systems*, 37: 5817–5849, 2024.

Alex Graves. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*, 2016.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=tEYskw1VY2.

Han Guo, Songlin Yang, Tarushii Goel, Eric P Xing, Tri Dao, and Yoon Kim. Log-linear attention. *arXiv preprint arXiv:2506.04761*, 2025.

Isabelle Guyon, Vladimir Vapnik, Bernhard Boser, Leon Bottou, and Sara A Solla. Structural risk minimization for character recognition. *Advances in neural information processing systems*, 4, 1991.

Mark H Hansen and Bin Yu. Model selection and the principle of minimum description length. *Journal of the american statistical association*, 96(454):746–774, 2001.

Ramin Hasani, Mathias Lechner, Tsun-Hsuan Wang, Makram Chahine, Alexander Amini, and Daniela Rus. Liquid structural state-space models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=g4OTKRKfS7R.

Donald Olding Hebb. *The organization of behavior: A neuropsychological theory*. Psychology press, 2005.

John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=kIoBbc76Sy.

Jerry Yao-Chieh Hu, Dennis Wu, and Han Liu. Provably optimal memory capacity for modern hopfield models: Transformer-compatible dense associative memories as spherical codes. *arXiv preprint arXiv:2410.23126*, 2024.

Jiaxi Hu, Yongqi Pan, Jusen Du, Disen Lan, Xiaqiang Tang, Qingsong Wen, Yuxuan Liang, and Weigao Sun. Comba: Improving bilinear rnns with closed-loop control. *arXiv preprint arXiv: 2506.02475*, 2025.

Kazuki Irie, Imanol Schlag, Róbert Csordás, and Juergen Schmidhuber. Going beyond linear transformers with recurrent fast weight programmers. *Advances in neural information processing systems*, 34:7703–7717, 2021.

Kazuki Irie, Francesco Faccio, and Jürgen Schmidhuber. Neural differential equations for learning to program neural nets through continuous learning rules. *Advances in Neural Information Processing Systems*, 35:38614–38628, 2022a.

Kazuki Irie, Imanol Schlag, Róbert Csordás, and Jürgen Schmidhuber. A modern self-referential weight matrix that learns to modify itself. In *International Conference on Machine Learning*, pages 9660–9677. PMLR, 2022b.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pages 5156–5165. PMLR, 2020.

Dmitry Krotov. Hierarchical associative memory. *arXiv preprint arXiv:2107.06446*, 2021.

Dmitry Krotov and John J Hopfield. Dense associative memory for pattern recognition. *Advances in neural information processing systems*, 29, 2016.

Yuri Kuratov, Aydar Bulatov, Petr Anokhin, Ivan Rodkin, Dmitry Igorevich Sorokin, Artyom Sorokin, and Mikhail Burtsev. BABILong: Testing the limits of LLMs with long context reasoning-in-a-haystack. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL `https://openreview.net/forum?id=u7m2CG84BQ`.

Yuri Kuratov, Matvey Kairov, Aydar Bulatov, Ivan Rodkin, and Mikhail Burtsev. Gradmem: Learning to write context into memory with test-time gradient descent. *arXiv preprint arXiv:2603.13875*, 2026.

Xiaoyu Li, Yuanpeng Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. On the expressive power of modern hopfield networks. *arXiv preprint arXiv:2412.05562*, 2024.

Zeman Li, Ali Behrouz, Yuan Deng, Peilin Zhong, Praneeth Kacham, Mahdi Karami, Meisam Razaviyayn, and Vahab Mirrokni. Tnt: Improving chunkwise training for test-time memorization. In *International Conference on Learning Representations*, volume 2026, pages 116868–116886, 2026.

Yi Heng Lim, Qi Zhu, Joshua Selfridge, and Muhammad Firmansyah Kasim. Parallelizing non-linear sequential models over the sequence length. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=E34AlVLN0v`.

Bo Liu, Rui Wang, Lemeng Wu, Yihao Feng, Peter Stone, and Qiang Liu. Longhorn: State space models are amortized online learners. *arXiv preprint arXiv:2407.14207*, 2024.

Colin Lockard, Prashant Shiralkar, and Xin Luna Dong. Openceres: When open information extraction meets the semi-structured web. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3047–3056, 2019.

Carlo Lucibello and Marc Mézard. Exponential capacity of dense associative memories. *Physical Review Letters*, 132(7):077301, 2024.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Byj72udxe.

William Merrill, Jackson Petty, and Ashish Sabharwal. The illusion of state in state-space models. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=QZgo9JZpLq.

Mehrnaz Mofakhami, Reza Bayat, Ioannis Mitliagkas, Joao Monteiro, and Valentina Zantedeschi. Performance control in early exiting to deploy large models at the same cost of smaller ones. *arXiv preprint arXiv:2412.19325*, 2024.

Tsendsuren Munkhdalai and Hong Yu. Neural semantic encoders. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 1, page 397. NIH Public Access, 2017.

Tsendsuren Munkhdalai, Alessandro Sordoni, Tong Wang, and Adam Trischler. Metalearned neural memory. *Advances in Neural Information Processing Systems*, 32, 2019.

Denis Paperno, German Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernandez. The LAMBADA dataset: Word prediction requiring a broad discourse context. In Katrin Erk and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1144. URL https://aclanthology.org/P16-1144/.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin A Raffel, Leandro Von Werra, Thomas Wolf, et al. The fineweb datasets: Decanting the web for the finest text data at scale. *Advances in Neural Information Processing Systems*, 37:30811–30849, 2024.

Bo Peng, Eric Alcaide, Quentin Gregory Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Nguyen Chung, Leon Derczynski, Xingjian Du, Matteo Grella, Kranthi Kiran GV, Xuzheng He, Haowen Hou, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartłomiej Koptyra, Hayden Lau, Jiaju Lin, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Guangyu Song, Xiangru Tang, Johan S. Wind, Stanisław Woźniak, Zhenyuan Zhang, Qinghua Zhou, Jian Zhu, and Rui-Jie Zhu. RWKV: Reinventing RNNs for the transformer era. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://openreview.net/forum?id=7SaXczaBpG.

Bo Peng, Daniel Goldstein, Quentin Anthony, Alon Albalak, Eric Alcaide, Stella Biderman, Eugene Cheah, Xingjian Du, Teddy Ferdinan, Haowen Hou, et al. Eagle and finch: Rwkv with matrix-valued states and dynamic recurrence. *arXiv preprint arXiv:2404.05892*, 2024.

Bo Peng, Ruichong Zhang, Daniel Goldstein, Eric Alcaide, Haowen Hou, Janna Lu, William Merrill, Guangyu Song, Kaifeng Tan, Saiteja Utpala, et al. Rwkv-7" goose" with expressive dynamic state evolution. *arXiv preprint arXiv:2503.14456*, 2025.

DL Prados and SC Kak. Neural network capacity using delta rule. *Electronics Letters*, 25(3):197–199, 1989.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Lukas Gruber, Markus Holzleitner, Thomas Adler, David Kreil, Michael K Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. Hopfield networks is all you need. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=tL89RnzIiCd.

David Raposo, Sam Ritter, Blake Richards, Timothy Lillicrap, Peter Conway Humphreys, and Adam Santoro. Mixture-of-depths: Dynamically allocating compute in transformer-based language models. *arXiv preprint arXiv: 2404.02258*, 2024.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1454. URL https://aclanthology.org/D19-1454/.

Imanol Schlag, Kazuki Irie, and Juergen Schmidhuber. Linear transformers are secretly fast weight programmers. In *International Conference on Machine Learning*, pages 9355–9366. PMLR, 2021.

Juergen Schmidhuber. Reducing the ratio between learning complexity and number of time varying variables in fully recurrent nets. In *ICANN'93: Proceedings of the International Conference on Artificial Neural Networks Amsterdam, The Netherlands 13–16 September 1993 3*, pages 460–463. Springer, 1993.

Jürgen Schmidhuber. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation*, 4(1):131–139, 1992.

Mark Schöne, Babak Rahmani, Heiner Kremer, Fabian Falck, Hitesh Ballani, and Jannes Gladrow. Implicit language models are rnns: Balancing parallelization and expressivity. *arXiv preprint arXiv:2502.07827*, 2025.

Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Tran, Yi Tay, and Donald Metzler. Confident adaptive language modeling. *Advances in Neural Information Processing Systems*, 35:17456–17472, 2022.

Julien Siems, Timur Carstensen, Arber Zela, Frank Hutter, Massimiliano Pontil, and Riccardo Grazzi. Deltaproduct: Improving state-tracking in linear RNNs via householder products. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=SoRiaijTGr.

Jimmy T.H. Smith, Andrew Warrington, and Scott Linderman. Simplified state space layers for sequence modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Ai8Hw3AXqks.

Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, et al. Learning to (learn at test time): Rnns with expressive hidden states. *arXiv preprint arXiv:2407.04620*, 2024.

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*, 2023.

Bohao Tang, Zhen Qin, Yuqi Pan, Zheng Li, Pengfei Liu, and Ya Zhang. Modular ttt: Rethinking test-time training as composable modules. *arXiv preprint arXiv:2608.07110*, 2026.

Matteo Tiezzi, Michele Casoni, Alessandro Betti, Tommaso Guidi, Marco Gori, and Stefano Melacci. On the resurgence of recurrent models for long sequences: Survey and research opportunities in the transformer era. *arXiv preprint arXiv:2402.08132*, 2024.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010.

Johannes Von Oswald, Maximilian Schlegel, Alexander Meulemans, Seijin Kobayashi, Eyvind Niklasson, Nicolas Zucchet, Nino Scherrer, Nolan Miller, Mark Sandler, Max Vladymyrov, et al. Uncovering mesa-optimization algorithms in transformers. *arXiv preprint arXiv:2309.05858*, 2023.

Ke Alexander Wang, Jiaxin Shi, and Emily B Fox. Test-time regression: a unifying framework for designing sequence models with associative memory. *arXiv preprint arXiv:2501.12352*, 2025.

Peihao Wang, Shan Yang, Xijun Wang, Tesi Xiao, Xin Liu, Changlong Yu, Yu Lou, Pan Li, Zhangyang Wang, Ming Lin, et al. Beyond test-time memory: State-space optimal control for llm reasoning. *arXiv preprint arXiv:2603.09221*, 2026.

Xiuying Wei, Anunay Yadav, Razvan Pascanu, and Caglar Gulcehre. Rat: Bridging rnn efficiency and attention accuracy via chunk-based sequence modeling. *arXiv preprint arXiv:2507.04416*, 2025.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule. *arXiv preprint arXiv:2412.06464*, 2024a.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated linear attention transformers with hardware-efficient training. In *Forty-first International Conference on Machine Learning*, 2024b. URL `https://openreview.net/forum?id=ia5XvxFUJT`.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Tianyuan Zhang, Sai Bi, Yicong Hong, Kai Zhang, Fujun Luan, Songlin Yang, Kalyan Sunkavalli, William T Freeman, and Hao Tan. Test-time training done right. *arXiv preprint arXiv:2505.23884*, 2025.

Tianyu Zhao and Llion Jones. Fast-weight product key memory. *arXiv preprint arXiv:2601.00671*, 2026.

# A Related Work

**Modern Linear Recurrent Neural Networks.** To address the quadratic complexity of Transformers, recent work has focused on efficient recurrent architectures that enable rapid training and inference [Tiezzi et al., 2024, Katharopoulos et al., 2020, Yang et al., 2024b, Schlag et al., 2021]. Early linear recurrent models, including linear attention [Katharopoulos et al., 2020], RetNet [Sun et al., 2023], RWKV [Peng et al., 2023], and S5 [Smith et al., 2023], utilized data-independent transition matrices with Hebbian update mechanisms. These evolved into second-generation architectures incorporating input-dependent parameters [Hasani et al., 2023, Peng et al., 2024, Gu and Dao, 2024, Yang et al., 2024a,b, Zhao and Jones, 2026] and more expressive memory rules based on the delta rule [Schlag et al., 2021, Yang et al., 2024a, Liu et al., 2024, Peng et al., 2025, Hu et al., 2025]. Recent iterations have extended these mechanisms to use deep memory modules using momentum-based update rules [Behrouz et al., 2025e], with non-linearity [Sun et al., 2024, Tang et al., 2026], or alternative objectives [Behrouz et al., 2025d,c, Kuratov et al., 2026, Wang et al., 2026, Tang et al., 2026]. Other works focus on more powerful learning updates, including OmegaNet [Behrouz et al., 2025a] and Oja's rules [Irie et al., 2022a] that extend upon delta and Hebbian rules, respectively. Notably, Siems et al. [2025] recently proposed applying multiple gradient descent steps per token to enhance state tracking. Parallel to these linear approaches, significant progress has been made in optimizing the training and performance of RNNs with non-linear recurrence [Behrouz et al., 2025c, Csordás et al., 2024, Merrill et al., 2024, Lim et al., 2024, Schöne et al., 2025, Li et al., 2026, Von Oswald et al., 2023, Gonzalez et al., 2024, Behrouz et al., 2025d, Kuratov et al., 2026, Irie et al., 2022b, Wang et al., 2026]. Recent studies such as log-linear attention and memory caching have focused on designing recurrent models whose effective memory state grows with sequence length [Behrouz et al., 2026, Guo et al., 2025, Wei et al., 2025]. Such approaches are fundamentally different from ours: PROTEUS keeps a fixed-size memory and activates it incrementally, whereas these models grow the memory itself, targeting a different aspect of recurrent computation. They are orthogonal to our contribution, and combining growing-memory techniques with PROTEUS—so that a model both incrementally activates its memory parameters and expands its memory once all are active—is an interesting future direction.

**Adaptive Computation.** A parallel line of work reduces cost by allocating computation non-uniformly rather than spending full capacity on every input. This idea has a long history, from conditional computation [Bengio et al., 2015, Graves, 2016] to early-exiting methods that halt processing for easy inputs [Elbayad et al., 2019, Schuster et al., 2022, Mofakhami et al., 2024]. More recently, Mixture-of-Depths [Raposo et al., 2024] routes only a subset of tokens through each layer's full computation, and Mixture-of-Recursions (MoR) [Bae et al., 2025] extends this to recursive Transformers, using lightweight routers to assign token-specific recursion depths while caching key–value pairs only for active tokens. These methods are related to PROTEUS in spirit, since both avoid applying the model's full capacity uniformly; however, they allocate *compute across tokens* in a Transformer, whereas PROTEUS schedules *effective memory capacity across positions* in a recurrent state. A further distinction is that their routing is *data-dependent* and learned, while our schedule is a fixed function of position. Notably, MoR finds that learned recursion depth tracks the contextual predictability of each token; this suggests that extending PROTEUS with a learned, data-dependent activation policy—adapting capacity to content rather than position alone—is a promising direction.

**Non-Uniform Capacity Allocation.** Our work is part of a broader move away from allocating model capacity uniformly. Classical work established that controlling effective capacity aids generalization [Guyon et al., 1991, Hansen and Yu, 2001, Ding et al., 2018], and recent evidence shows that high-capacity models trained by empirical risk minimization tend to *memorize* atypical examples rather than compress them into generalizable structure [Bayat et al., 2024], which is precisely the failure mode incremental activation restricts early capacity to avoid. Most closely related, Tapered Language Models (TLMs) [Bayat et al., 2026] show that allocating more parameter capacity to earlier layers and less to later ones—under a fixed budget—improves language modeling across architectures, resonating with our thesis that capacity should be allocated non-uniformly rather than spread evenly across the model.

**Associative Memory and Fast Weight Programs.** We ground our architecture in the principles of associative memory, originally framed by Hopfield [1982] as energy minimization for storing key-value pairs. While classical Hopfield networks were limited by low storage capacity, modern

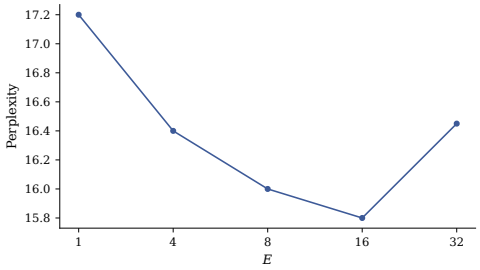


Figure 3: Validation perplexity as a function of the number of blocks $E$.

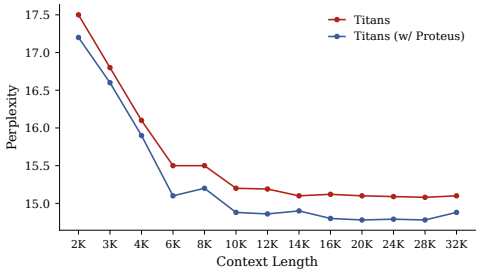


Figure 4: Validation perplexity by token position for Titans with and without PROTEUS.

continuous variants have addressed these bottlenecks through dense associative memories and exponential kernels [Krotov and Hopfield, 2016, Krotov, 2021, Li et al., 2024, Lucibello and Mézard, 2024], and have gained renewed prominence through their theoretical equivalence to the attention mechanism in Transformers [Ramsauer et al., 2021, Hu et al., 2024]. The related paradigm of treating linear layers as dynamic, writable memory was formalized in fast weight programmers [Schmidhuber, 1992, 1993], in which a slow network generates the weights of a fast network that processes the sequence. Two learning rules dominate this setting, Hebbian learning [Hebb, 2005] and the delta rule [Prados and Kak, 1989], both of which let the network act as a key–value associative memory and have been studied extensively as efficient alternatives to standard attention [Munkhdalai and Yu, 2017, Munkhdalai et al., 2019, Irie et al., 2021, Schlag et al., 2021, Behrouz et al., 2025a, Hasani et al., 2023, Peng et al., 2025].

## B Ablations and Analysis

**Effect of the number of blocks.** The number of partition blocks $E$ is the central hyperparameter of PROTEUS, controlling how finely the memory's effective capacity is scheduled over the context. We ablate it by varying $E$ from 1 to 32, holding everything else fixed. Note that $E = 1$ recovers the base model exactly: with a single block there is no scheduling, so the full memory is active from the first token. This makes the ablation a clean isolation of the scheduling effect, since only $E$ changes. Figure 3 reports validation perplexity as a function of $E$. Perplexity improves sharply from $E = 1$ to $E = 8$ and reaches its minimum at $E = 16$, but degrades again at $E = 32$. This non-monotonicity is precisely the tension anticipated in Section 3: a finer partition sharpens the early bottleneck, which improves compression up to a point, but too fine a schedule leaves too little capacity active for the initial context, since at $E = 32$ the first tokens read and write only $1/32$ of the memory, so the bottleneck begins to cost more than the compression it buys.

**Perplexity by token position.** To examine *where* PROTEUS's gains arise, we measure validation perplexity as a function of token position in the context, comparing a base model against its PROTEUS variant (Figure 4). PROTEUS achieves lower perplexity at every position, indicating that the scheduled capacity helps throughout the context rather than trading early-context quality for late-context gains. The gap widens as the schedule progresses and is largest around the 8K training context length, where the memory becomes fully active; beyond that point it narrows slowly but remains positive out to 32K. This is consistent with the mechanism described in Section 4.1: the advantage accumulates while fresh capacity is still being unlocked, and the better-compressed state that results continues to pay off at lengths past the training window, where no further activation occurs.

## C Future Directions

Several directions follow naturally. A first is to replace our fixed schedule with a *learned*, data-dependent activation policy, and to characterize the optimal schedule, including whether uniform growth is the right choice and how it depends on data statistics and the underlying update rule. A second is to combine incremental activation with growing-memory architectures such as memory caching [Behrouz et al., 2026], so that a model both activates its fixed memory over the context and expands that memory once all blocks are active. This pairing is a particularly natural fit: incremental

activation governs how a fixed budget is spent within the training window, while a growing memory supplies genuinely new capacity beyond it, so the two mechanisms cover complementary regimes and together could extend the schedule indefinitely with context. A third is to extend the schedule beyond the training window at inference time, stretching $\Delta$ so that capacity continues to unlock at context lengths longer than those seen in training; whether a state compressed under the training schedule remains well-calibrated when the unlock points move is an open question. Finally, the same principle extends beyond recurrent memory to other parameter subsets and to continual adaptation settings, where scheduling which parameters are active over a long stream of experience may help balance fast adaptation against retention of past knowledge.

## NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly present the paper's main idea and summarize our experimental results.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The main text outlines our method's limitations, including its scope, assumptions, and possible directions for future improvement.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: We do not include any theoretical results in this paper.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide detailed instructions for our new architecture in the paper.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We included all necessary information to reproduce the results, including implementation details, and used only publicly available datasets.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: The experimental settings and all relevant details are discussed in the main body of the paper.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: We pretrain models from scratch using a fixed data order, which reduces randomness, so our experiments show minimal stochastic variation and do not require error bars or statistical significance measures.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We have used TPUv5 to perform all the experiments in the paper.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This paper complies with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This study does not raise significant societal concerns.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This paper does not present such risks of data or model misuse.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We appropriately cite all external assets and clearly state their licenses and terms of use.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: We do not provide new assets in this research.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not involve such experiments.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This research does not involve crowdsourcing or human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not involved in developing the core method of this research.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.