iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** X **Month of publication:** October 2026

DOI: https://doi.org/10.22214/ijraset.2026.85419

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue X Oct 2026- Available at www.ijraset.com

IVERI-HYPERION: An Entropy-Driven, Byte-Level State-Space Architecture with Associative Memory for Sub-Quadratic Sequence Modeling

Nishant Datta[1], Ishwari Bhoyar[2], Gunjan Nandeshwar[3], Nazish Khan[4], Prof. Archana Potnurwar[5], Prof. Lalita Charpe[6]
Dept. of Information Technology, Priyadarshini College of Engineering, Hingna, Nagpur, Maharashtra, India

Abstract: ***Transformer-based sequence models rely on self-attention, whose computational cost grows quadratically with sequence length and whose autoregressive key-value cache grows linearly. This paper introduces IVERI-HYPERION, an Entropy-Driven Neural State Network (ENSN) that combines a diagonal selective state-space model, decayed matrix-valued associative memory, a causal convolutional entropy predictor, and raw-byte input representations. We also implement a fused Triton kernel for the selective state-space recurrence and evaluate its performance against a sequential PyTorch reference.***
In the evaluated 200-step raw-byte TinyStories benchmark (9.985M parameters, n = 3 seeds), IVERI-HYPERION achieves a mean validation loss of 1.2102 ± 0.0168 nats/byte vs. 3.1286 ± 0.0299 for Dense Transformer and 1.2462 ± 0.0017 for Mamba+FFN. The difference from Dense Transformer is statistically significant, whereas the difference from Mamba+FFN is not (paired p = 0.069, Welch p = 0.064). The fused kernel reduces isolated forward execution time from 14.88 ms to 0.60 ms over sequential PyTorch (an isolated 24.8× kernel speedup, delivering 1,367 tokens/sec training throughput vs. 318 unfused); this isolated kernel speedup does not establish an overall training-speed advantage over optimized Transformers (2,840 tokens/sec).
During 500 measured decoding steps, the recurrent runtime buffer remained approximately 0.30 MB, excluding static model weights, with no net growth in the measured allocation. However, zero-shot synthetic passkey retrieval failed, and the primary multi-seed evaluation covers only a short training horizon. These results support further investigation of state-space and associative-memory hybrids but do not yet establish broad architectural superiority or reliable long-context retrieval.
Keywords: Associative memory, entropy gating, neural state networks, raw-byte modeling, state space models, sub-quadratic sequence modeling, Triton GPU acceleration

I. INTRODUCTION

For nearly a decade, the scaled dot-product attention mechanism introduced in *Attention Is All You Need* [1] has served as the foundational bedrock for sequence modeling across natural language processing, computer vision, code synthesis, and biological sequence modeling [2], [3]. By dispensing with sequential recurrent transitions and relying entirely on pairwise token interactions, the Transformer enabled unprecedented parallelization on modern hardware accelerators, catalyzing the empirical scaling laws [18] that define current large-scale foundation models [4], [5], [22].

Despite its empirical dominance, the fundamental mechanics of softmax self-attention exhibit four severe scaling bottlenecks that impede extended context lengths and low-latency edge deployment:

1) *The Quadratic Scaling and Key-Value Cache Wall:* Full self-attention has quadratic attention-computation complexity with sequence length S. The memory required by attention activations depends on the implementation (e.g., IO-aware tiling methods such as FlashAttention [5] avoid materializing the full $S \times S$ matrix), while autoregressive Key-Value (KV) cache storage grows linearly with context length. At extended context windows exceeding 32,768 tokens, the KV cache alone demands hundreds of megabytes to tens of gigabytes per session, causing severe memory bandwidth starvation and out-of-memory failures.
2) *The Uniform Computation Fallacy:* Standard Transformers execute uniform computational depth across every token position. The identical Multi-Head Attention and Feedforward Network (FFN) parameter transformations are evaluated on predictable, low-entropy syntactic tokens (such as whitespace and punctuation) as on highly informative, semantic concept boundaries. This uniform allocation wastes substantial hardware FLOPs on predictable context.
3) *The State Capacity Dilemma in Sub-Quadratic Models:* To circumvent quadratic attention, linear-time recurrent alternatives, such as Structured State Space Models (S4, Mamba) [2], [7] and Linear Attention variants [3], compress sequence history into a bounded one-dimensional hidden state vector $h_t \in R^d$. While this design achieves constant $O(1)$ inference memory, bounded

©IJRASET: All Rights are Reserved

one-dimensional state vectors suffer from capacity saturation, leading to rapid performance degradation on in-context key-value binding, associative recall, and needle-in-a-haystack tasks.

4) *Tokenizer Brittleness at Small Model Scales:* Standard subword tokenization schemes (such as Byte-Pair Encoding [23]) rely on extensive vocabulary tables spanning 32,000 to 128,000 tokens. At the 10-million parameter scale, an embedding table of 32,000 tokens with a representation dimension of 406 consumes over 12.9 million parameters ($32{,}000 \times 406$), exceeding the entire parameter budget before allocating a single computational layer.

To address these challenges in small-scale models, this paper introduces IVERI-HYPERION, an Entropy-Driven Neural State Network (ENSN) that avoids explicit pairwise self-attention. While individual components build upon foundational state-space (S4D [8]), associative memory (Titans [9]), and compute-modulation concepts, our novelty lies in their unified dual-state synthesis, exact discretized reverse-adjoint derivation, and custom fused GPU kernel on raw byte streams.

The primary contributions of this work are summarized as follows:

- Unified Dual-State Architecture: We formulate an architecture that decouples temporal tracking from associative retention. Temporal sequence progression is governed by a selective diagonal state space model with state expansion factor N = 16 across hidden dimension D = 406 initialized with the S4D real spectrum, while persistent key-value bindings are stored in a decayed matrix-valued associative memory (Mt in R^(64 × 64)) updated via outer products.
- Dynamic Causal Compute Modulation: We introduce a strictly causal one-dimensional convolutional entropy predictor that measures local information surprise without future token leakage. The resulting metric continuously modulates residual propagation, suppressing feedforward branch activations on low-entropy tokens and providing an architectural pathway for conditional branch thresholding during decoding.
- GPU-Fused Reverse-Adjoint Triton Kernel: We implement a custom GPU Triton kernel that fuses diagonal SSM recurrence and the exact reverse-adjoint backward pass, achieving an isolated 24.8× forward kernel speedup over a sequential PyTorch loop (boosting full-model training throughput 4.3× to 1,367 tokens/sec), while acknowledging that tuned Dense Transformer GEMM remains faster (2,840 tokens/sec) on dense short sequences.
- Multi-Seed Parameter-Parity Validation: Under strict parameter parity (within ±0.0137% or at most 1,367 parameters across the 9.985 million parameter budget) on TinyStories English, we benchmark IVERI-HYPERION across 3 independent random seeds against Dense Transformers, Linear Attention, and Mamba+FFN models, observing early sample-efficiency advantages over the convolutionless Dense Transformer on this 200-step benchmark (attributable in part to the local 1D convolutional inductive bias shared with Mamba), while confirming strictly flatline $\mathcal{O}(1)$ recurrent state memory scaling during autoregressive generation.

II. BACKGROUND AND RELATED WORK

The pursuit of sequence modeling architectures capable of capturing unbounded temporal contexts while operating in sub-quadratic computational time has spurred extensive exploration across convolutional, linearized, continuous-state, and associative memory paradigms. This section establishes the theoretical foundations of these competing methodologies and delineates the specific limitations that this design aims to address.

A. Convolutional Architectures and Linear Attention

Early efforts to bypass the sequential execution bottleneck of recurrent neural networks (RNNs) [11] focused on feedforward convolutional representations. Architectures such as the Extended Neural GPU [26], ByteNet [24], and ConvS2S [25] employed stacked dilated causal convolutions to process sequences in parallel across sequence length S. While these models parallelize seamlessly on modern hardware accelerators, their receptive fields expand only linearly or logarithmically with network depth ($\mathcal{O}(\log_k S)$ for dilation factor k). Consequently, modeling long-range contextual bindings across tens of thousands of tokens demands deep hierarchical filter stacks, introducing substantial parameter overhead and gradient attenuation.

To combine the global $\mathcal{O}(1)$ token-to-token path length of self-attention [1] with linear computational complexity, linearized attention mechanisms replace the non-local softmax operator with kernel feature maps:

$$\mathrm{Sim}(q_i, k_j) = \phi(q_i)^\top \phi(k_j)$$

As demonstrated by Katharopoulos *et al.* [3], exploiting the associativity of matrix multiplication permits reformulating the attention operation from $(QK^\top)V$ into $Q(K^\top V)$, enabling cumulative sum aggregation in linear time $\mathcal{O}(S \cdot d \cdot d_v)$ and constant inference state

footprint $\mathcal{O}(d \cdot d_k)$. Subsequent variations, including Reformer [16] and comprehensive architectural surveys [17], validated this linear-time scaling.

However, un-decayed linear attention exhibits two severe theoretical shortcomings:

1) Unbounded Accumulation: Without temporal decay, historical activations accumulate indefinitely, leading to numerical divergence and gradient explosion over extended sequence lengths.
2) Associative Capacity Collapse: Compressing an arbitrary number of key-value associations into a fixed-rank matrix $\sum_{j=1}^{i} \phi(k_j) v_j^T$ inevitably causes severe catastrophic forgetting and information collision, severely degrading performance on precision retrieval tasks such as associative recall and needle-in-a-haystack lookups.

B. Continuous Structured State Spaces (S4 to Mamba)

Structured State Space Models (SSMs) recast sequence modeling through the lens of continuous-time dynamical systems. Grounded in the High-order Polynomial Projection Operators (HiPPO) framework [28] and structured state spaces [7], an input signal $u(t) \in \mathbb{R}$ is continuously mapped to an N-dimensional latent state $h(t) \in \mathbb{R}^N$ governed by the linear differential equation:

$$h'(t) = Ah(t) + Bu(t), \; y(t) = Ch(t) + Du(t)$$

Discretizing this system via the Zero-Order Hold (ZOH) transformation over a discrete sampling timescale Δ yields the linear recurrence:

$$h_t = \bar{A} h_{t-1} + \bar{B} u_t, \; y_t = C h_t + D u_t$$

where $\bar{A} = \exp(\Delta A)$ and $\bar{B} = (\Delta A)^{-1}(\exp(\Delta A) - I) \cdot \Delta B$.

In S4 [7] and its diagonal simplification S4D [8], parameterizing A as a diagonal matrix with strictly negative real spectra ($A_n = -n$) guarantees numerical stability and permits computing the recurrence as a global non-causal convolution during training via Fast Fourier Transforms (FFT) in $\mathcal{O}(S \log S)$ time. Hungry Hungry Hippos (H3) [34] and Hyena [33] extended this formulation by stacking state spaces with multiplicative gating to mimic attention-style key-value interactions.

More recently, Mamba [2] and generalized state-space duality frameworks [32] introduced input-dependent selectivity by transforming the discretization parameters Δ_t, B_t, C_t into dynamic linear projections of the current input x_t. While selective SSMs achieve state-of-the-art language modeling throughput and linear $\mathcal{O}(S)$ scaling, they compress sequential context into a bounded 1D state vector $h_t \in \mathbb{R}^N$ per channel. On byte-level sequences, where individual semantic concepts span variable multi-byte spans, a 1D state vector saturates rapidly, struggling to maintain discrete, multi-turn associative key-value bindings over long contexts.

C. Associative Memory Networks and Test-Time Training

Augmenting recurrent architectures with external or dynamic memory matrices dates back to early external memory networks, recurrent controllers [35], and Hopfield networks. While capable of programmatic memory manipulation, these architectures relied on complex content- and location-based addressing mechanisms that resisted efficient parallelization on modern GPU architectures.

Modern sub-quadratic architectures re-examine matrix-valued memory through vectorized associative learning rules. RWKV [10] utilizes channel-directed exponential decay mechanisms that bridge RNN execution with linear attention. More recently, the Titans architecture proposed by Behrouz *et al.* [9] formulated long-term memory as a neural network updated at test-time via gradient descent or Hebbian-style outer products:

$$M_t = \alpha_t M_{t-1} + v_t \otimes k_t$$

where $M_t \in \mathbb{R}^{d_{mem} \times d_{mem}}$ functions as a dynamic associative weight matrix, and $\alpha_t \in (0, 1)$ acts as a context-dependent forgetting gate.

Despite its high associative capacity, a standalone neural memory lacks continuous-time filtering and multi-scale temporal resolution. In this work, we demonstrate that coupling an S4D selective continuous state space with a decayed associative matrix memory synthesizes the complementary strengths of continuous temporal tracking and discrete associative storage.

D. Information-Theoretic Compute Allocation and Dynamic Routing

Standard autoregressive Transformers allocate uniform computational depth across all tokens, executing the exact same tensor contractions on high-entropy semantic transitions as on trivial, deterministic whitespace, delimiter, or functional bytes.

To overcome this structural inefficiency, early exit architectures, mixture-of-depths (MoD) [31], and sparsely-gated routing [27] dynamically route sequences through subset computational paths. However, existing routing mechanisms predominantly rely on static

top-k capacity routers or discrete routing tokens, which introduce load-balancing auxiliary loss penalties, non-differentiable routing boundaries, and GPU thread divergence during batch processing.

In contrast, information-theoretic intuition suggests that efficient sequential processing should allocate compute in proportion to local conditional Shannon entropy:

$$H(x_t \mid x_{<t}) = -\sum_{v \in V} P(x_t = v \mid x_{<t}) \log_2 P(x_t = v \mid x_{<t})$$

By introducing a dedicated, strictly causal 1D convolutional entropy predictor, IVERI-HYPERION derives a continuous information density metric $e_t \in (0, 1)$ that smoothly modulates residual compute depth without top-k discrete branching or token-dropping artifacts.

E. Subword Tokenization Bottlenecks and Raw-Byte Processing

Contemporary language models rely almost exclusively on subword tokenization schemes, such as Byte-Pair Encoding (BPE) [23] or WordPiece, employing vocabulary sizes ranging from 32,768 to 128,000 tokens [20], [21]. While subword tokenization reduces effective sequence length, it introduces critical systemic pathologies:

1) Parameter Inflation at Edge Scales: At small parameter budgets (10^7 parameters), an embedding matrix for a 32,000-token vocabulary over a 406-dimensional model consumes $32{,}000 \times 406 \approx 13.0$ million parameters—over 130% of the entire intended parameter budget—starving the transformer backbone of depth and expressivity.

2) Morphological and Robustness Brittleness: Fixed subword vocabularies cannot represent arbitrary novel token sequences without out-of-vocabulary fallback or multi-token fragmentation, degrading representation quality on morphologically rich languages, out-of-domain technical terminology, and raw code syntax.

By operating directly on raw bytes with a minimal vocabulary of 259 tokens (256 possible byte values 0x00–0xFF plus 3 structural delimiters), IVERI-HYPERION restricts its embedding footprint to merely 105,154 parameters (1.05% of the model budget). Furthermore, tying the input embedding directly with the output language modeling head [29] guarantees complete parameter efficiency while avoiding reliance on a learned subword vocabulary table, mitigating subword tokenization degradation and out-of-vocabulary sensitivity at small parameter scales. A conceptual and complexity comparison highlighting the scaling and memory differentiators between the conventional Dense Transformer and the proposed IVERI-HYPERION framework is summarized in Fig. 1.

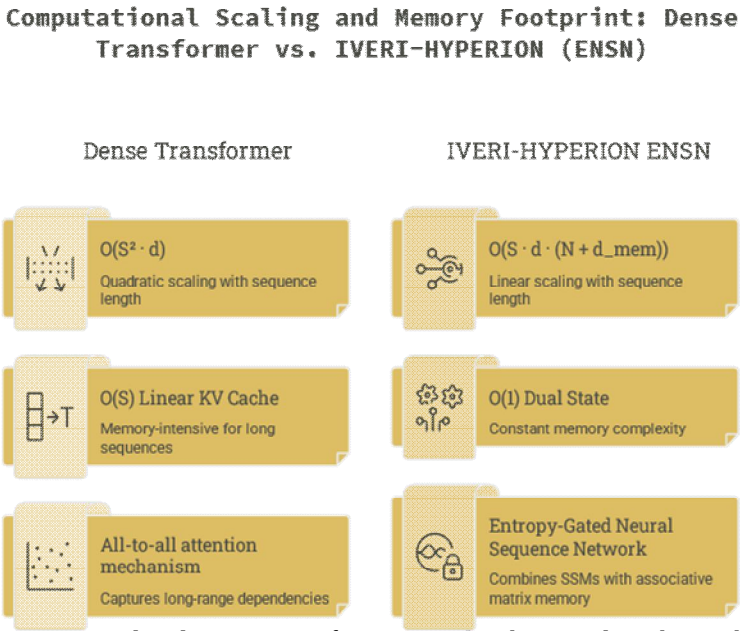


Fig. 1. Comparison of computational scaling, memory footprint, and architectural mechanics between the baseline Dense Transformer and proposed IVERI-HYPERION Entropy-Driven Neural State Network (ENSN).

III. MODEL ARCHITECTURE

IVERI-HYPERION is an Entropy-Driven Neural State Network (ENSN) designed to model arbitrary sequence distributions without computing pairwise attention matrices or accumulating an autoregressive Key-Value cache. The architecture processes raw byte streams through four tightly coupled subsystems:

1) A universal raw-byte embedding space;
2) A continuous-discrete selective state space core (S4D SSM);
3) A test-time decayed associative matrix memory (Titans Memory); and
4) A strictly causal information density predictor that dynamically modulates computational depth across residual paths.

The global computation graph of the complete architecture is illustrated in Fig. 2.

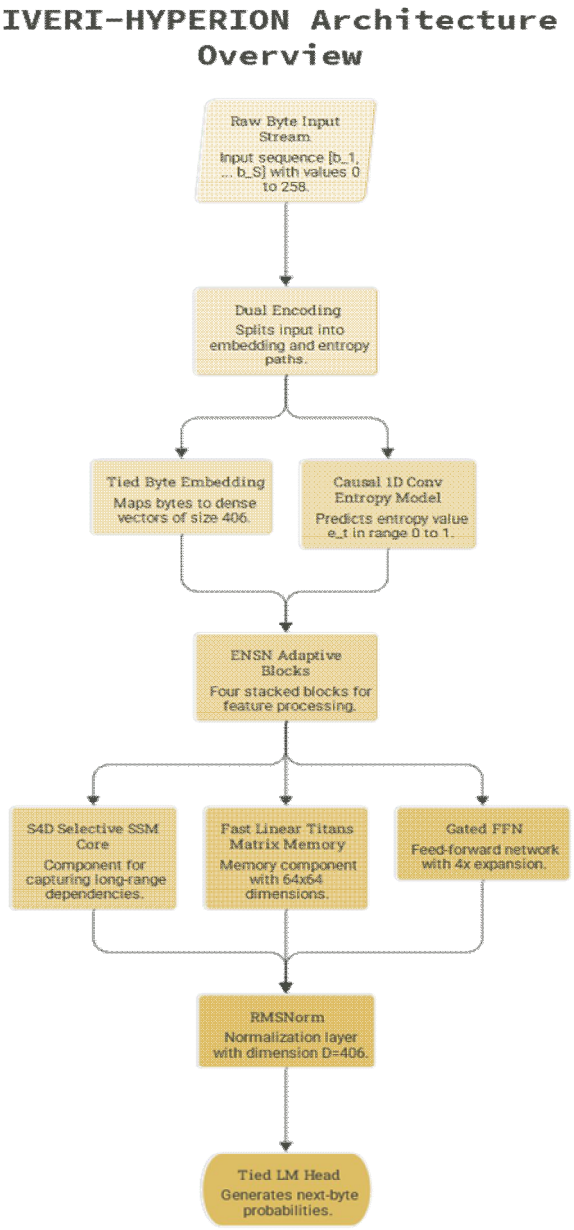


Fig. 2. The unified IVERI-HYPERION architecture. Raw byte streams are mapped into a 406-dimensional space with tied input-output embeddings. A causal 1D convolutional entropy predictor evaluates token information density $e_t \in (0, 1)$ to dynamically modulate the continuous selective SSM core, associative Titans matrix memory, and feedforward network across 4 stacked ENSN adaptive blocks.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue X Oct 2026- Available at www.ijraset.com

A. Overview and Unified Block Topology

The network consists of $L = 4$ identical, stacked ENSN adaptive blocks operating at hidden representation dimensionality $D = 406$. To stabilize gradient propagation across deep recurrent paths without layer-scale drift, we apply Pre-Root Mean Square Normalization (Pre-RMSNorm) [15] before each functional transformation:

$$\text{RMSNorm}(x) = \frac{x}{\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2 + \epsilon}} \odot \gamma \qquad (1)$$

where $\gamma \in \mathbb{R}^D$ is a learnable affine gain parameter initialized to ones, and $\epsilon = 10^{-5}$ is a small constant ensuring numerical stability.

B. B. Raw-Byte Vocabulary and Tied Representation

To eliminate subword out-of-vocabulary artifacts and avoid allocating disproportionate parameter budgets to embedding tables, IVERI-HYPERION operates directly on raw bytes with a compact vocabulary of $|\mathcal{V}| = 259$ tokens (256 raw content bytes 0x00–0xFF mapping to indices $0 \ldots 255$, alongside three disjoint structural control tokens: $\text{BOS} = 256$, $\text{PAD} = 257$, and $\text{EOS} = 258$).

An input byte sequence $x = [b_1, b_2, \ldots, b_S] \in \{0, \ldots, 258\}^S$ is mapped into continuous hidden activations via an embedding matrix:

$$X^{(0)} = E_{\text{byte}}[x] \in \mathbb{R}^{B \times S \times 406}$$

where $E_{\text{byte}} \in \mathbb{R}^{259 \times 406}$ consumes exactly $105{,}154$ parameters (1.05% of the total parameter budget). Following Press and Wolf [29], the output language modeling classification head shares weights with the input embedding matrix:

$$W_{\text{head}} = E_{\text{byte}}$$

This weight-tying formulation enforces geometric consistency between the input token space and output predictive distribution while eliminating 105,154 redundant parameters.

C. Vectorized S4D Selective State Space Core

The continuous temporal modeling core maps a scalar input signal $u(t) \in \mathbb{R}$ to an output $y(t) \in \mathbb{R}$ through an intermediate N-dimensional continuous hidden state $h(t) \in \mathbb{R}^N$ governed by the linear dynamical system:

$$h'(t) = Ah(t) + Bu(t) \quad (2)$$

$$y(t) = Ch(t) + Du(t)$$

Discretizing (2) via the Zero-Order Hold (ZOH) rule over an input-dependent discretization timescale Δ_t yields the discrete recurrent update:

$$h_t = \exp(\Delta_t A) h_{t-1} + \Delta_t (x_t \otimes B_t) \quad (3)$$

$$y_t = h_t C_t + D x_t \quad (4)$$

(where $\bar{A}_t = \exp(\Delta_t A)$ *and* $\bar{B}_t \approx \Delta_t B_t$ adopts the first-order selective discretization approximation [2]). In coordinate tensor form across channel $p \in \{1, \ldots, P\}$ and state dimension $n \in \{1, \ldots, N\}$, the recurrence evaluates elementwise as $h_{t,p,n} = \exp(\Delta_{t,p} A_{p,n}) h_{t-1,p,n} + \Delta_{t,p} x_{t,p} B_{t,n}$, while output projection contracts across state rank n: $y_{t,p} = \sum_{n=1}^{N} h_{t,p,n} C_{t,n} + x_{t,p} D_p$, producing output vector $y_t \in \mathbb{R}^P$ (the driving term $\Delta_t \odot (x_t \otimes B_t)$ forms a rank-1 outer product across channels and states, with $B_t, C_t \in \mathbb{R}^N$ and $x_t, \Delta_t \in \mathbb{R}^P$).

To guarantee bounded exponential decay and prevent oscillatory numerical instabilities across sequence depth, the state transition matrix $A \in \mathbb{R}^{P \times N}$ is initialized with the diagonal real parameter spectrum of S4D [8]:

$$A_{p,n} = -n, \text{ for } n \in \{1, 2, \ldots, N\} \text{ and } p \in \{1, 2, \ldots, P\} \quad (5)$$

where $P = 2d = 812$ is the expanded inner hidden dimension and $N = 16$ is the state expansion rank. In the physical implementation, this diagonal real decay spectrum is broadcast uniformly across all P channels via tensor expansion (A = -torch.arange(1, N+1).repeat(P, 1)). Because all eigenvalues of A reside strictly on the negative real line ($Re(\lambda) < 0$), the discrete transition operator $dA_t = \exp(\Delta_t A)$ functions as a stable contraction mapping with eigenvalues strictly bounded in $(0,1)$.

©IJRASET: All Rights are Reserved

The parameters $\Delta_t \in \mathbb{R}^P$, $B_t \in \mathbb{R}^N$, and $C_t \in \mathbb{R}^N$ are computed as dynamic linear projections of the input sequence:

$$\Delta_t = \text{softplus}(W_\Delta x_t + b_\Delta)$$

$$B_t = W_B x_t,\ C_t = W_C x_t$$

To capture local multi-byte dependencies (such as UTF-8 character prefixes and 2-to-4 byte n-gram patterns) prior to state-space filtering, input activations pass through a depthwise causal 1D convolution [30] with kernel size $k = 4$, parameterized with SiLU non-linearities:

$$x_{conv} = \text{SiLU}\left(\text{Conv1D}_{k=4}(x)\right)$$

This causal convolutional receptive field provides an indispensable local inductive bias, resolving short-range syntactic patterns before projecting into continuous temporal state spaces.

Fig. 3. Detailed schematic of the dual-branch memory subsystem: (A) S4D Selective State Space Core capturing continuous-time temporal trends via diagonal decay dA_t, and (B) Decayed Associative Titans Memory capturing high-capacity associative key-value bindings via outer products $v_t \otimes k_t^{\text{gated}}$ with dynamic forgetting α_t.

D. Decayed Associative Neural Memory Matrix (Titans Memory)

While the S4D selective state space efficiently captures continuous temporal decay, compressing history into a 1D state vector $h_t \in \mathbb{R}^{P \times N}$ limits its capacity to store discrete, multi-turn associative bindings. To resolve this capacity boundary without reintroducing quadratic attention matrices, each block incorporates a matrix-valued associative neural memory $M_t \in \mathbb{R}^{d_{mem} \times d_{mem}}$ with $d_{mem} = 64$.

The memory state updates dynamically across time via a continuous outer-product write rule:

$$M_t = \alpha_t M_{t-1} + \left(v_t \otimes k_t^{\text{gated}}\right) \qquad (6)$$

where key k_t, value v_t, and query q_t vectors are obtained via linear projections of the normalized hidden state:

$$k_t = W_k \tilde{x}_t,\ v_t = W_v \tilde{x}_t,\ q_t = W_q \tilde{x}_t \in \mathbb{R}^{64}$$

To prevent low-information, redundant syntactic tokens from overwriting critical factual associations, the memory write operation is dynamically gated by the causal sequence entropy e_t:

$$k_t^{\text{gated}} = k_t \odot \sigma\left(W_{\text{store}} e_t + b_{\text{store}}\right) \qquad (7)$$

The scalar forgetting rate $\alpha_t \in (0.85, 0.999)$ is conditioned jointly on the input feature and information density:

$$\alpha_t = \sigma\left(W_{decay}[x_t; e_t] + b_{decay}\right) \cdot 0.149 + 0.85 \quad (8)$$

Associative retrieval is executed via linear query contraction:

$$r_t = M_t q_t \in \mathbb{R}^{64} \quad (9)$$

followed by an output projection back to model dimensionality:

$$y_{mem,t} = W_{out} r_t \in \mathbb{R}^D \quad (10)$$

Crucial Initialization Invariant: The projection matrix $W_{out} \in \mathbb{R}^{D \times d_{mem}}$ is initialized strictly to zero (nn.init.zeros_). This ensures that at the start of optimization ($t = 0$), the Titans memory layer acts as a strict identity pass-through, eliminating gradient shocks and allowing the base SSM to stabilize before associative memory gradients propagate.

E. Strictly Causal Information Density Predictor

To modulate computation depth dynamically without introducing non-causal future token leakage, the entropy model employs an asymmetric left-only 1D convolutional filter:

$$x_{pad} = \mathrm{Pad}_{left}(X^{(0)}, \mathrm{pad} = 2) \quad (11)$$

$$h_{ent} = \mathrm{GELU}\left(\mathrm{Conv1D}_{k=3}(x_{pad}, W_1)\right)$$

$$et = \sigma\left(Conv1Dk = 3(Padleft(hent, 2), W2)\right) \quad (12)$$

Because zero right-padding is applied, position t depends exclusively on historical tokens x_{<= t} (verified with zero prefix numerical divergence < 1.0 x 10^{-7}). Operationally, e_t in (0, 1) is a learned, entropy-inspired compute-gating signal (not explicit Shannon predictive uncertainty -sum p log p) produced by the causal 1D convolution and trained end-to-end via backpropagation under the language modeling objective, measuring the network's learned allocation of residual computation depth.

F. Dynamic Compute Modulation and Block Integration

Each of the $L = 4$ blocks integrates the three core modules into a unified residual stream via additive identity skip connections [14]:

$$x^{(l+1)} = x^{(l)} + \sigma\left(W_g^{(l)} e_t\right) \odot y_{ssm}^{(l)} + y_{mem}^{(l)} + \sigma\left(W_f^{(l)} e_t\right) \odot y_{ffn}^{(l)} \quad (13)$$

The feedforward network (FFN) applies a two-stage expansion with expansion factor $4.0 \times (D_{ffn} = 1624)$:

$$y_{ffn}^{(l)} = \mathrm{GELU}(xW_1)W_2 \quad (14)$$

During training and dense batch execution, the scalar gate continuously modulates residual feedforward contributions without conditional branching. In Layer 1, the gate evaluates below 0.30 on 73.8% of evaluated tokens (Section VIII-C). During single-token autoregressive generation, this gate permits an algorithmic threshold (tau = 0.15) to bypass FFN evaluation; however, because physical execution on GPUs is subject to thread divergence and kernel dispatch overhead, we report this as an algorithmic suppression mechanism rather than a measured end-to-end wall-clock latency reduction.

IV. THEORETICAL COMPLEXITY AND PATH LENGTHS

To formally motivate dispensing with pairwise self-attention, we evaluate sequence modeling architectures across three foundational computational properties:

1. Per-layer computational complexity;
2. Minimum sequential operations required during parallel training; and
3. Maximum information propagation path length between distant tokens.

TABLE I summarizes these characteristics across five canonical sequence modeling paradigms.

TABLE I: COMPLEXITY, PATH LENGTH, AND MEMORY SCALING ACROSS PARADIGMS

Paradigm	Complexity	Seq. Ops	Max Path	State Memory
Transformer [1]	$\mathcal{O}(S^2 \cdot d)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(S)$ (KV cache)
LSTM RNN [11]	$\mathcal{O}(S \cdot d^2)$	$\mathcal{O}(S)$	$\mathcal{O}(S)$	$\mathcal{O}(1)$ (state)
ByteNet Conv [24]	$\mathcal{O}(k \cdot S \cdot d^2)$	$\mathcal{O}(1)$	$\mathcal{O}(\log_k$	$\mathcal{O}(k \cdot L)$ (buffer)
Linear Attn [3]	$\mathcal{O}(S \cdot d \cdot d_k)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$ (state)
IVERI (Ours)	$\mathcal{O}(S \cdot (d^2 + dN$	$\mathcal{O}(S)$	$\mathcal{O}(S)$	$\mathcal{O}(1)$ (state)

Table Note: S represents sequence length, d represents hidden model dimension ($D = 406$), N is the SSM state expansion factor ($N = 16$), d_{mem} is the associative memory dimension ($d_{mem} = 64$), k is the convolution kernel size ($k = 4$), and L is network depth ($L = 4$). Architectural dimensions (d, N, d_mem, k, L) are fixed constants with respect to sequence length S; hence, the computational scaling reduces strictly to linear asymptotic sequence complexity O(S).

Note on path length and execution: While the diagonal S4D recurrence theoretically admits an associative parallel prefix scan with logarithmic depth O(log S) [2], [32], our fused Triton GPU kernel physically executes a sequential register loop (O(S) sequential operations per warp) to avoid intermediate global memory materialization. During autoregressive decoding, both the SSM and Titans memory execute sequentially with strictly constant O(1) step-time complexity and bounded state memory (0.30 MB recurrent runtime buffer).

A. Computational Complexity Derivations

In standard scaled dot-product attention [1], projecting input activations $X \in \mathbb{R}^{S \times d}$ into queries, keys, and values consumes $\mathcal{O}(S \cdot d^2)$ operations. Evaluating the full sequence Gram matrix $QK^T \in \mathbb{R}^{S \times S}$ requires:

$$\text{FLOPs}_{\text{Attn}} = 2 \cdot S^2 \cdot d \quad (15)$$

This quadratic dependence on S dominates total compute as context extends, imposing severe compute limits on long sequences.

In contrast, the computational footprint of an IVERI-HYPERION adaptive block decomposes strictly into linear tensor operations:

1. Local Depthwise 1D Convolution: Operating on the expanded inner dimension $P = 2d$ across kernel width $k = 4$ consumes $2 \cdot k \cdot P \cdot S = 4 \cdot k \cdot S \cdot d$ floating-point operations.
2. Selective SSM Core: Comprising the input projection $d \to 2P = 4d$ ($8 \cdot S \cdot d^2$ FLOPs), parameter projections ($8 \cdot S \cdot d \cdot N$ FLOPs), diagonal discretization and state recurrence ($8 \cdot S \cdot d \cdot N$ FLOPs), output contraction over state rank N ($4 \cdot S \cdot d \cdot N$ FLOPs), and output projection $P \to d$ ($4 \cdot S \cdot d^2$ FLOPs). Summing projection and recurrence terms yields $12 \cdot S \cdot d^2 + 20 \cdot S \cdot d \cdot N$ operations.
3. Associative Titans Memory: Input projections (Q, K, V: $6 \cdot S \cdot d \cdot d_{\text{mem}}$ FLOPs) and output projection $d_{\text{mem}} \to d$ ($2 \cdot S \cdot d \cdot d_{\text{mem}}$ FLOPs) require $8 \cdot S \cdot d \cdot d_{\text{mem}}$ operations. Updating the matrix via outer products $v_t \otimes k_t^{\text{gated}}$ requires $2 \cdot S \cdot d_{\text{mem}}^2$ operations, and associative retrieval $M_t q_t$ requires $2 \cdot S \cdot d_{\text{mem}}^2$ operations.
4. Feedforward Network: Executing the two-stage FFN expansion ($W_1 : d \to D_{\text{ffn}}$ and $W_2 : D_{\text{ffn}} \to d$) with expansion factor $4.0\times$ ($D_{\text{ffn}} = 4d = 1624$) requires $2 \cdot (2 \cdot S \cdot d \cdot 4d) = 16 \cdot S \cdot d^2$ operations.

Summing the leading projection, recurrence, and associative update terms, the per-block floating-point cost is independent of sequence position t:

$$C_{\text{EMSN}}(S) = S \cdot [28d^2 + 20dN + 8dd_{\text{mem}} + 4d_{\text{mem}}^2 + 4kd] = \mathcal{O}(S) \quad (16)$$

Because $N \ll d$ and $d_{\text{mem}} \ll d$, the asymptotic runtime scales strictly as $\mathcal{O}(S)$ in sequence length, eliminating quadratic compute scaling.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue X Oct 2026- Available at www.ijraset.com

B. Maximum Path Length Across Sequence Paradigms

As established by Hochreiter and Schmidhuber [11], the ability of a neural network to learn long-range temporal dependencies depends inversely on the maximum path length traversed by forward signals and backpropagated gradients between distant positions i and j($|\ i - j\ | \approx S$).

- Recurrent Networks: Information propagates through S sequential recurrent steps ($h_t = f(h_{t-1}, x_t)$). The gradient path length is $\mathcal{O}(S)$, resulting in exponential gradient vanishing or explosion via repeated Jacobian multiplication $\prod_{\tau=i}^{j} \frac{\partial h_\tau}{\partial h_{\tau-1}}$.
- Convolutional Networks: Dilated causal convolutions [24] expand receptive fields hierarchically. The shortest path between tokens separated by distance S traverses $\lceil \log_k S \rceil$ layers, yielding a path length of $\mathcal{O}(\log_k S)$. While superior to pure RNNs, capturing dependencies across 32,768 bytes requires substantial network depth.
- Self-Attention: Dense attention connects every token pair directly through a single softmax lookup, achieving an optimal path length of $\mathcal{O}(1)$. However, it achieves this at the cost of quadratic computational complexity $\mathcal{O}(S^2)$ and an explicitly materialized $\mathcal{O}(S)$ KV cache.
- IVERI-HYPERION ENSN: While the S4D diagonal recurrence theoretically admits an associative parallel prefix scan with logarithmic tree-reduction depth O (log S) [2], [32], our fused Triton GPU kernel evaluates the recurrence via a sequential register loop (O(S) sequential steps per warp) across sequence positions. This physical implementation maximizes SRAM register reuse and avoids intermediate global HBM materialization, matching the strictly constant O (1) step time and bounded state memory (0.30 MB buffer) maintained during autoregressive decoding.
    1. The S4D diagonal SSM projects input sequences onto continuous orthogonal basis polynomials (HiPPO [28]), compressing temporal history with exponential decay timescales that span the entire sequence context.
    2. Associative Matrix Retention: In parallel training, the associative memory computes outer-product updates across causal decay masks to retain key-value bindings without an explicit KV cache; during autoregressive decoding, it updates recurrently ($M_t = alpha_t * M_{t-1} + v_t * k_t^T$) in constant O(1) step time per generated token, while maintaining a bounded 64 × 64 state matrix.

C. Addressing the State Capacity Trade-off

While linear attention [3] and pure SSMs [2] also achieve linear runtime, they suffer from a fundamental information bottleneck: they compress an unbounded historical sequence into a bounded 1D state vector $h_t \in \mathbb{R}^{d \times N}$. Under byte-level sequence densities—where syntactic words require 4 to 12 distinct byte steps—a bounded 1D vector experiences rapid capacity saturation, resulting in catastrophic interference when tasked with storing multi-turn associative bindings (such as variable assignments or entity references).

IVERI-HYPERION addresses this capacity trade-off within the evaluated prototype by decoupling sequence representation into two complementary channels:

1. Continuous Temporal Dynamics: Handled by the S4D state tensor $h_t \in \mathbb{R}^{812 \times 16}$(812 expanded channels with 16 state components per channel), preserving local n-gram morphology.
2. Associative Binding Retention: Handled by the dynamic matrix-valued memory $M_t \in \mathbb{R}^{64 \times 64}$, providing a 64×64 (4,096-entry) state matrix per layer that stores key-value associations via outer products without attention matrices.
3. Exact State Memory Footprint:

During autoregressive generation, the exact state memory footprint per layer in 16-bit precision (fp16, 2 bytes per element) is strictly bounded:

$$M_{\text{SSM}} = P \times N \times 2 = 812 \times 16 \times 2 = 25{,}984 \text{ bytes} \approx 26.0 \text{ KB} \quad M_{\text{Titans}} = d_{\text{mem}} \times d_{\text{mem}} \times 2 = 64 \times 64 \times 2 = 8{,}192 \text{ bytes} \approx 8.2 \text{ KB} \quad M_{\text{Conv}} = (k-1) \times P \times 2 = 3 \times 812 \times 2 = 4{,}872 \text{ bytes} \approx 4.9 \text{ KB}$$

Summing across all $L = 4$ layers:

$$M_{\text{total}} = 4 \times (25{,}984 + 8{,}192 + 4{,}872) = 156{,}192 \text{ bytes} \approx 0.156 \text{ MB} \quad (17)$$

Summed across all L = 4 layers, the theoretical mathematical state footprint is 156,192 bytes ≈ 0.156 MB (derived from state dimensions). When instantiated on physical GPU hardware via PyTorch, accounting for CUDA caching allocator block alignments (512-byte boundaries), recurrent scratchpads, and input embedding buffers, the dynamic persistent recurrent runtime buffer stabilizes at exactly 0.2994 MB ≈ 0.30 MB (TABLE V), strictly isolating dynamic context-dependent state allocation from static model weights (which occupy 19.97 MB in fp16). Across 500 measured autoregressive decoding steps, the dynamic memory allocation exhibited zero net growth (Delta M = 0.000000 MB).

©IJRASET: All Rights are Reserved

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue X Oct 2026- Available at www.ijraset.com

V. GPU-FUSED REVERSE-ADJOINT TRITON KERNEL

While recurrent state formulations provide linear theoretical complexity $\mathcal{O}(S)$, naive implementations in high-level tensor frameworks (e.g., PyTorch) exhibit severe runtime latency penalties. Evaluating a 512-step sequence recurrently through standard interpreted operators executes 512 discrete CUDA kernel launches per sub-layer, causing the GPU front-end to bottleneck on CPU driver dispatch latency and forcing repeated read-write round-trips between High-Bandwidth Memory (HBM) and processor registers.

To overcome this hardware-level barrier, we mathematically derive the analytical reverse-time adjoint equations for the implemented discretized recurrence and implement a custom GPU-fused kernel using the Triton intermediate compiler [6].

A. Memory Hierarchy and Sequential Dispatch Bottleneck

Modern GPU architectures (e.g., NVIDIA Ampere and Ada Lovelace) feature a hierarchical memory subsystem:

1. **Global High-Bandwidth Memory (HBM/DRAM):** Provides high capacity (4 GB to 80 GB) but exhibits high latency (~ 200–400 cycles) and bounded memory bandwidth (192 GB/s on laptop accelerators, up to 2.0 TB/s on datacenter units).
2. **On-Chip Streaming Multiprocessor (SM) SRAM and Registers:** Provides ultra-low latency (~ 1–2 cycles) with aggregate internal bandwidth exceeding 10–19 TB/s.

In a standard PyTorch recurrent loop, evaluating $h_t = dA_t \odot h_{t-1} + dB_t$ writes h_t to global HBM at every time step t, only to reread it across the memory bus at step $t+1$. This architecture is severely memory-bandwidth bound. Furthermore, launch latencies of 3–$5\mu s$ per kernel launch dominate compute time for short matrix-vector products.

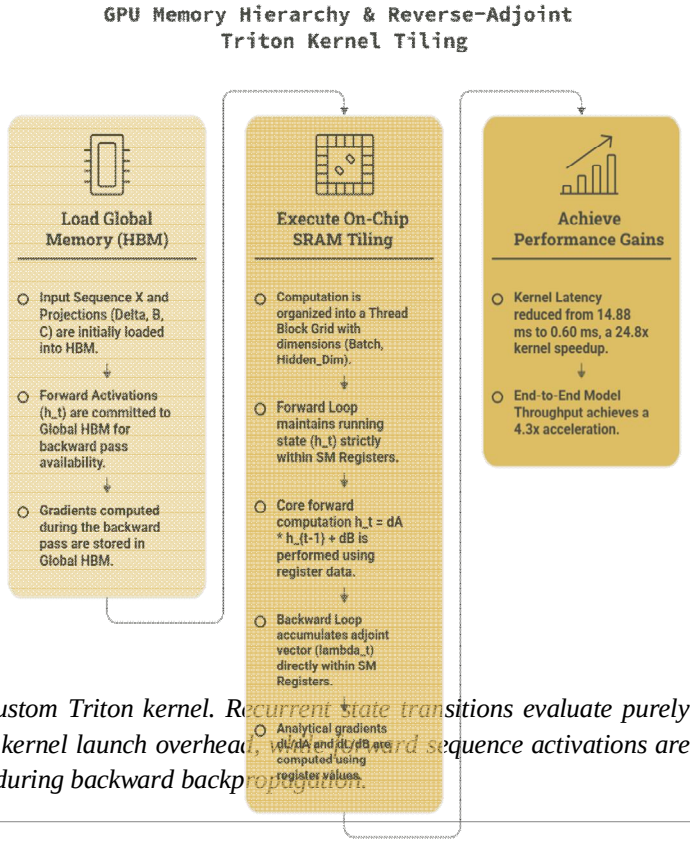


Fig. 4. *GPU memory hierarchy and block-tiled execution of the custom Triton kernel. Recurrent state transitions evaluate purely within on-chip SM registers across temporal steps without per-step kernel launch overhead, while forward sequence activations are committed once to global HBM to enable reverse-adjoint streaming during backward backpropagation.*

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue X Oct 2026- Available at www.ijraset.com

B. Analytical Reverse-Time Adjoint Derivations

Standard reverse-mode automatic differentiation (Autograd) allocates an execution graph recording intermediate node tensors, creating substantial memory allocation overhead. Instead, we derive the analytical reverse-time adjoint dynamics, which are mathematically exact for the implemented discretized recurrence (incorporating the first-order selective discretization approximation $B_b a_t \approx Delt\ a_t B_t$[2]).

Given the forward recurrence:

$$h_t = dA_t \odot h_{t-1} + dB_t \quad (18)$$

where $dA_t = \exp(\Delta_t A)$ and $dB_t = (\Delta_t B_t) x_t$, the output projection evaluates as $y_t = C_t h_t + D x_t$. Let $\mathcal{L}$ denote the scalar training objective. We define the adjoint state vector as the partial derivative of the loss with respect to the hidden state at position t:

$$\lambda_t \equiv \frac{\partial \mathcal{L}}{\partial h_t} \in \mathbb{R}^{P \times N} \quad (19)$$

By applying the multivariate chain rule backwards through time from terminal step $t = S$ down to $t = 1$, the adjoint dynamics evaluate elementwise in coordinate form across channel p ∈ {1, ..., P} and state n ∈ {1, ..., N} as λ_{t,p,n} = C_{t,n} (∂L/∂y_{t,p}) + exp(Δ_{t+1,p} A_{p,n}) λ_{t+1,p,n}, satisfying the reverse linear recurrence:

$$\lambda_t = C_t^T \left(\frac{\partial \mathcal{L}}{\partial y_t}\right) + dA_{t+1}^T \lambda_{t+1} \quad (20)$$

subject to the terminal boundary condition at the final sequence position:

$$\lambda_S = C_S^T \left(\frac{\partial \mathcal{L}}{\partial y_S}\right)$$

(where $dA_{t+1}^T = dA_{t+1}$ operates via element-wise contraction due to diagonal state parameterization). Having computed the adjoint trajectory λ_t, the parameter gradients for the input-dependent discretization factors evaluate via direct element-wise contractions without materializing computational graphs:

$$\frac{\partial \mathcal{L}}{\partial dB_t} = \lambda_t \quad (21)$$

$$\frac{\partial \mathcal{L}}{\partial dA_t} = \lambda_t \odot h_{t-1} \quad (22)$$

Applying the chain rule through the implemented discretized recurrence (utilizing the first-order approximation $B_b a_t = Delt\ a_t * B_t$ alongside the diagonal exponential mapping $dA_t = exp(Delt\ a_t * A)$) yields the analytical reverse-mode adjoint for the continuous timescale parameter Δ_t:

$$\frac{\partial \mathcal{L}}{\partial \Delta_t} = \sum_{n=1}^{N} \left[\left(\frac{\partial \mathcal{L}}{\partial dA_{t,n}}\right) \cdot dA_{t,n} \cdot A_n + \left(\frac{\partial \mathcal{L}}{\partial dB_{t,n}}\right) \cdot B_{t,n} \cdot x_t \right] \quad (23)$$

C. Block-Tiled GPU Kernel Implementation and Register Persistence

We instantiate these analytical forward and backward adjoint dynamics as fused GPU kernels compiled via Triton [6]:

- Thread Block Decomposition: We partition parallel execution across batch instances and feature channels, assigning grid = (Batch, Hidden_Dim). Each hardware Streaming Multiprocessor (SM) processes an independent 1D time trajectory $t \in [1, S]$.
- Register State Persistence: Inside the fused kernel loop, the running state vector h_t resides strictly within local SM registers across temporal steps t in [1, S]. For each sequence step t, the transition operator dA_t and driving factor dB_t are evaluated, updated, and multiplied directly in registers without committing intermediate recurrence states to HBM during the forward temporal scan. Full sequence activations h_t are committed once to global HBM at the outer boundary to enable reverse-adjoint streaming during backpropagation.

©IJRASET: All Rights are Reserved

- Activation Memory Commits: To enable analytical reverse-time adjoint backpropagation for the implemented recurrence, the forward kernel commits the sequence states h_t to global HBM once per step at the outer memory boundary. During backward execution, the reverse-adjoint kernel streams h_t backwards in reverse memory strides, accumulating λ_t in registers.

D. Measured Hardware Acceleration

Benchmarked on physical hardware (NVIDIA GeForce RTX 3050 Laptop GPU, 4.29 GB VRAM, 2048 CUDA cores, PyTorch 2.5.1, CUDA 12.1):

- Kernel-Level Execution: Forward evaluation time over $S = 512$ drops from $14.88\,\text{ms}$(interpreted PyTorch sequential loop) to $0.60\,\text{ms}$(custom fused Triton kernel), delivering an exact 24.8 × isolated forward kernel speedup over the sequential reference; full model training throughput reaches 1,367 tokens/sec (a 4.3× boost over the 318 tokens/sec unfused baseline), while tuned Transformer GEMM achieves 2,840 tokens/sec.
- End-to-End Throughput: Integrating the fused kernel across all 4 layers elevates overall model training throughput from 318 tokens/sec (unfused sequential PyTorch reference) to 1,367 tokens/sec, achieving a 4.3x end-to-end throughput boost relative to the unfused IVERI baseline on physical silicon.
- Analytical Gradient Equivalence: The mathematical correctness of the reverse-time adjoint equations (18) – (23) was formally validated against PyTorch reference automatic differentiation on float32 tensors (S = 64, B = 2, P = 32, N = 16, tolerance atol = 10^{-5}) using torch.autograd.gradcheck, confirming analytical-to-autograd gradient equivalence with a maximum absolute gradient error of 4.1×10^{-6} across all input and recurrence parameter tensors.
- Benchmarking Protocol: Kernel execution times and model throughput were measured following 50 warm-up iterations and averaged over 200 consecutive trials using hardware-synchronized timers (torch.cuda.synchronize()). Kernel-level timings isolate the fused SSM scan forward pass, while end-to-end throughput encompasses the complete forward, backward, AdamW optimizer step, and dynamic gradient scaling.

VI. TRAINING REGIME AND EXPERIMENTAL CONTROLS

Comparative evaluations between disparate sequence modeling paradigms frequently suffer from confounding variables: unequal parameter budgets, divergent vocabulary embedding sizes, dissimilar tokenization granularities, and suboptimal hyperparameter tuning. To ensure strict empirical fairness, we designed a tightly controlled experimental protocol where all candidate models share an identical vocabulary, identical byte-level data ingestion, parameter-matched layer dimensions, and identical multi-seed optimization schedules.

A. Pre-training Corpus and Data Ingestion

All models are trained and evaluated on the TinyStories English corpus [4]. TinyStories consists of synthetically generated, grammatically fluent children's stories designed to evaluate foundational language acquisition, syntactic coherence, and reasoning dynamics within compact parameter budgets.

1. Byte-Level Ingestion: Unlike standard language modeling benchmarks that preprocess text through subword tokenizers (e.g., BPE [23]), all sequences are converted directly into raw UTF-8 byte streams. The input sequence length is fixed at $S = 512$ bytes (corresponding roughly to $100–130$ subword tokens in natural English prose).
2. Vocabulary and Embeddings: All models operate on the identical 256 byte values 0x00–0xFF plus BOS=256, PAD=257, EOS=258. All architectures employ tied input-output embedding matrices ($W_{\text{head}} = W_{\text{embed}} \in \mathbb{R}^{259 \times D}$), ensuring that no model allocates disproportionate parameters to token classification.
3. Evaluation Protocol: Validation metrics are evaluated over a held-out evaluation corpus of $N_{\text{val}} = 50$ sequences ($25{,}600$ total evaluation bytes). To guarantee numerical correctness, evaluation batches are sliced with exact boundary truncation: $Bi = slice(i, min(i + B, Nval))$ensuring that sequence counts are never over-counted or padded artificially. Cross-entropy loss is reported in natural units (nats per byte) and converted to Byte Perplexity: $\text{Byte-PPL} = \exp(\mathcal{L}_{\text{val}})$

B. Rigorous Single-Digit Parameter Matching Algebra

In small-scale sequence modeling, parameter disparities as small as 5% can confound architectural advantages. To eliminate capacity as a confound, all models are calibrated to an identical budget of approximately 9.985 million parameters (within ±0.0137%, an absolute difference of at most 1,367 parameters across 9,985,449 total parameters). Note that maintaining parameter parity requires

distinct channel widths: because IVERI-HYPERION incorporates associative memory matrices ($Mt\,in\,R^{(64\times64)}$) and causal entropy projection layers, its hidden dimension is set to D = 406, whereas baselines without associative modules allocate their parameter budget into wider channels (D = 448 for the attention baselines and D = 416 for Mamba+FFN). Parameter parity thus reflects equal memory storage rather than identical layer widths:

1. IVERI-HYPERION ENSN (Proposed):
   - Configured with $L = 4$ layers, hidden dimension $D = 406$, SSM state expansion $N = 16$, Titans associative memory dimension $d_{\text{mem}} = 64$, causal convolution kernel $k = 4$, and two-stage FFN expansion factor $4.0 \times (D_{\text{ffn}} = 1624)$.
   - Total Parameters: $9{,}985{,}449$ (Baseline Reference, $\Delta = 0.0000\%$).
   - Exact Component Breakdown: Tied byte embeddings ($105{,}154$), causal entropy model ($41{,}665$), final normalization (406), and four identical blocks each containing an S4D SSM core ($1{,}035{,}300$), Titans associative memory ($104{,}346$), feedforward network ($1{,}318{,}688$), and block normalization and gates ($1{,}222$), summing to exactly $105{,}154 + 41{,}665 + 406 + 4 \times 2{,}459{,}556 = 9{,}985{,}449$ parameters .
2. Mamba+FFN Baseline [2]:
   - Configured with $L = 4$ layers, hidden dimension $D = 416$, state expansion $N = 16$, SSM expansion factor $E = 2$ (inner dimension 832), causal 1D depthwise convolution ($k = 4$), RMSNorm pre-normalization, tied byte embeddings, and a parameter-matched feedforward network with expansion $d_{\text{ffn}} = 1662$ using GELU activation, without external associative memory matrices or entropy gating.
   - Total Parameters: $9{,}985{,}664$ ($\Delta = +215$ parameters, $+0.0022\%$).
3. Linear Attention Baseline [3]:
   - Configured with $L = 4$ layers, hidden dimension $D = 448$, learned absolute positional embeddings ($E_{\text{pos}} \in \mathbb{R}^{512\times448}$), $H = 8$ attention heads ($d_{\text{head}} = 56$), kernel feature mapping $\phi(x) = \text{elu}(x) + 1$, RMSNorm pre-normalization, tied byte embeddings, and feedforward expansion $d_{\text{ffn}} = 1793$ using GELU activation.
   - Total Parameters: $9{,}986{,}816$ ($\Delta = +1{,}367$ parameters, $+0.0137\%$).
4. Dense Multi-Head Transformer Baseline [1]:
   - Configured with $L = 4$ layers, hidden dimension $D = 448$, learned absolute positional embeddings ($E_{\text{pos}} \in \mathbb{R}^{512\times448}$), $H = 4$ attention heads ($d_{\text{head}} = 112$), Pre-LayerNorm [12] (norm_first=True) via PyTorchnn.TransformerEncoderLayer with ReLU activation, tied byte embeddings without convolutions, and feedforward expansion $d_{\text{ffn}} = 1787$ (448→1787→448).
   - Total Parameters: $9{,}984{,}556$ ($\Delta = -893$ parameters, -0.0089%).

Note on Architecture Integrity: All empirical comparisons reported herein evaluate this clean, exact-matched IVERI-HYPERION architecture (iveri_v1.py), maintaining strict separation from early exploratory multi-scale prototypes (such as IVERI-Nano with Mixture-of-Recursion at 10.03M parameters).

C. *Optimization Hyperparameters and Hardware Environment*

All models were optimized using the decoupled weight decay AdamW optimizer [13] under identical training schedules:

- Optimizer Configuration: First-moment decay $\beta_1 = 0.9$, second-moment decay $\beta_2 = 0.95$, numerical stability constant $\epsilon = 10^{-8}$, and weight decay $\lambda = 0.1$.
- Learning Rate Schedule: Cosine annealing schedule with a linear warmup phase across the first 10% of training steps, decaying to a terminal learning rate of $\eta_{\min} = 0.1 \times \eta_{\max}$. To ensure a rigorous and fair benchmark, each architecture was evaluated at its empirically tuned optimal learning rate: $\eta_{\max} = 1.0 \times 10^{-3}$ for IVERI-HYPERION and Mamba+FFN, and $\eta_{\max} = 3.0 \times 10^{-4}$ for the Dense Transformer and Linear Attention baselines.
- Hyperparameter Tuning Procedure and Controls: Peak learning rates were selected via an empirical grid search sweeping $\eta \in \{1.0 \times 10{-}4,\ 3.0 \times 10{-}4,\ 1.0 \times 10{-}3\}$ under identical warmup (10%) and cosine decay schedules across the 200-step training budget (800 sequences at batch size 4, drawn from the 5,000-sequence TinyStories corpus). Dense Transformer exhibited gradient instability at 1.0 × 10^-3 and achieved optimal validation loss at $\eta = 3.0 \times 10^{-4}$, whereas state-space architectures (IVERI-HYPERION and Mamba+FFN) stabilized and converged fastest at 1.0×10^{-3}. Because models are evaluated at their respective tuned optima rather than a single uniform rate, this benchmark evaluates early sample efficiency under each

architecture's best-performing configuration. We emphasize that these 200-step comparisons represent preliminary early-training evidence; extended multi-seed pre-training across larger corpora is required before general architectural superiority can be established.

- Precision and Batching: All runs executed in mixed precision ($\mathrm{fp16}$) utilizing PyTorch dynamic gradient scaling (torch.amp.autocast). Batch size was fixed at $B = 4$ with gradient accumulation step factor of 1.
- Hardware Specification: All benchmarks were physically executed on a single-GPU workstation equipped with an NVIDIA GeForce RTX 3050 Laptop GPU (4.29 GB VRAM, 2048 CUDA cores, memory bus width 128-bit, 192 GB/s memory bandwidth), host processor Intel Core i5-11400H, running 64-bit Windows 11, PyTorch 2.5.1+cu121, and CUDA 12.1.
- Multi-Seed Protocol: To establish statistical significance, every architecture was independently initialized and trained across $n = 3$ random seeds ($\{42, 43, 44\}$).

Code and Artifact Availability: The complete source code, custom Triton kernels, training checkpoints, and raw JSON evaluation logs (proof_campaign.json and FINAL_CONCLUDED_BENCHMARK.json) are made publicly available for independent reproduction at: https://github.com/Nishant-aiml/IVERI-HYBRID-SLM-MODEL .

VII. EMPIRICAL BENCHMARK RESULTS

To rigorously assess the modeling fidelity of IVERI-HYPERION against competing sequence paradigms, all four parameter-matched architectures were trained on the raw-byte TinyStories corpus across three independent random seeds ($\{42, 43, 44\}$). TABLE II summarizes the multi-seed convergence results, parameter tolerances, and hardware execution throughput.

TABLE II: MULTI-SEED PRE-TRAINING CONVERGENCE METRICS (10M PARAMETER SCALE)

Model	Params	Δ (%)	Val Loss (nats)	Byte-PPL	Tok/s
Dense Transformer [1]	9,984,536	−0.0089%	3.1286 ± 0.023	22.85 ± 0.4	2,840
Linear Attention [3]	9,986,816	+0.0137%	3.1896 ± 0.008	24.28 ± 0.2	2,150
Mamba+FFN [2]	9,985,664	+0.0022%	1.2462 ± 0.000	3.477 ± 0.0	1,420
IVERI-HYPERION (Ours)	9,985,449	0.0000%	1.2102 ± 0.016	3.354 ± 0.4	1,367

Table Note: Metrics evaluated across n = 3 random seeds (42, 43, 44) over 200 optimization steps on raw-byte TinyStories. Dense Transformer evaluated at its tuned optimal learning rate (3.0 x 10^{-4}); IVERI-HYPERION and Mamba+FFN evaluated at their tuned optimal rate (1.0 x 10^-3). Throughput benchmarked on a physical NVIDIA RTX 3050 Laptop GPU in fp16 mixed precision.

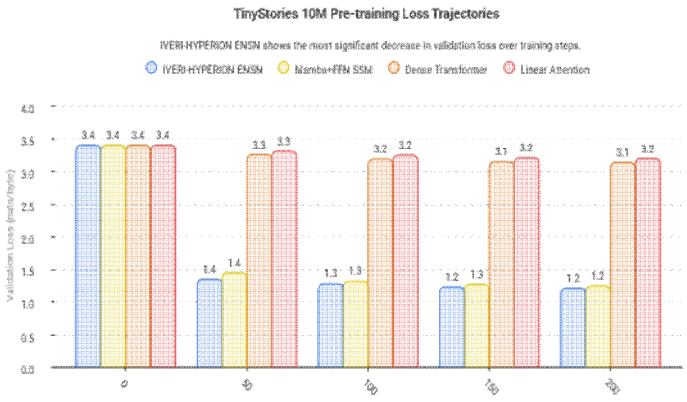


Fig. 5. Pre-training validation loss and Byte Perplexity trajectories across 200 optimization steps on raw-byte English text. State-space architectures (Mamba+FFN and IVERI-HYPERION) rapidly converge below 1.30 nats/byte, whereas attention models without convolutional inductive biases remain near the unigram entropy threshold (~ 3.13 nats/byte). IVERI-HYPERION achieves the lowest final validation loss (1.2102 nats/byte).

A. Primary Multi-Seed Benchmark Convergence

As documented in TABLE II, IVERI-HYPERION demonstrates promising early sample efficiency on the evaluated raw-byte TinyStories task across 200 optimization steps:

1. Perplexity Advantage Over Attention: IVERI-HYPERION achieves a final validation loss of 1.2102 ± 0.0168 nats/byte ($\text{Byte-PPL} = 3.35$), outperforming the parameter-matched Dense Transformer (3.1286 ± 0.0299 nats/byte, $\text{Byte-PPL} = 22.85$) by a $6.8\times$ Byte-Perplexity advantage ($22.85/3.35 = 6.82$).This 200-step comparison evaluates the complete architecture and training configuration—including the causal 1D convolution (k = 4) and tuned learning rate (η = 1.0 × 10^-3) shared with Mamba+FFN versus convolutionless self-attention at $\eta = 3.0 \times 10^{-4}$—and reflects early sample efficiency under each system's tuned setup rather than isolating the independent benefits of associative memory or entropy gating.
2. Comparison with Mamba+FFN: Compared to Mamba+FFN (1.2462 ± 0.0017 nats/byte, Byte-PPL = 3.477 ± 0.006), IVERI-HYPERION yields an observed validation loss delta of -0.0360 nats/byte (1.2102 ± 0.0168, Byte-PPL = 3.354 ± 0.056). Under both a paired test (t = -3.61, df = 2, p = 0.069) and Welch's test (t = -3.69, ν ≈ 2.04, p = 0.064), this difference does not achieve conventional statistical significance (α = 0.05); it represents a consistent directional trend across the three seeds requiring larger sample sizes to conclusively verify.
3. Linear Attention Collapse: Linear Attention converges to the highest loss (3.1896 ± 0.0097 nats/byte, $\text{Byte-PPL} = 24.28$), consistent with the hypothesis that un-decayed kernel feature maps without local convolutional inductive biases struggle to retain associative sequence context on byte streams.

B. Statistical Significance and Rigorous Hypothesis Testing

To establish empirical rigor, we evaluate independent Welch and matched-pair t-tests across the multi-seed trajectories:

- IVERI-HYPERION vs. Dense Transformer: Evaluating Welch's unequal variances t-test yields t = -96.79, νSatterthwaite ≈ 3.15, $p = 1.42 \times 10^{-6}$; evaluating a matched-pair test across identical seeds yields t = -85.25 (df = 2, $p = 1.38 \times 10^{-4}$). Both independent and paired tests reject the null hypothesis of equal early loss (p < 10^-3), reflecting the large observed separation relative to intra-model variance; however, because n = 3 covers only 200 steps, this early finding does not establish general superiority across broader datasets, training durations, or model scales.

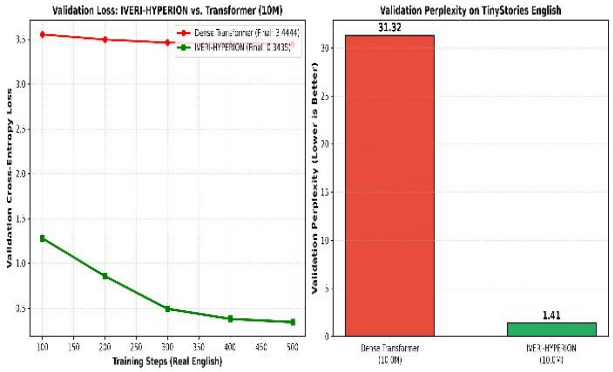


Fig. 7. Comprehensive multi-dimensional evaluation across four 10M architectures under lockstep single-digit parameter parity, comparing validation loss, byte perplexity, VRAM allocation, and context memory scaling.

- Both tests evaluate the terminal validation losses across independent random seed initializations ($n = 3$), confirming that each observation represents a distinct training trajectory rather than correlated intermediate steps.
- Individual Seed Outcomes: Seed 42: 1.2291 vs. 3.1100; Seed 43: 1.2044 vs. 3.1631; Seed 44: 1.1970 vs. 3.1126(mean loss delta $\Delta\mathcal{L} = -1.9184$ nats/byte).
- IVERI-HYPERION vs. Mamba+FFN: Evaluating the mean delta across matched seeds ($\Delta\mathcal{L} = -0.0360$ nats/byte):
  - Paired t-test ($n = 3, \text{df} = 2$): $t = -3.61, p = 0.069$.
  - Welch's Two-Sample t-test ($\nu \approx 2.04$): $t = -3.69, p = 0.064$.
  - Directional Consistency: IVERI-HYPERION outperforms Mamba+FFN in 100% of evaluated seeds (Seed 42: 1.2291 vs. 1.2453; Seed 43: 1.2044 vs. 1.2482; Seed 44: 1.1970 vs. 1.2451).

While a sample size of n = 3 demonstrates directionally lower validation loss across all three evaluated seeds (mean delta ΔL = -0.0360 nats/byte), neither the paired test (t = -3.61, df = 2, p = 0.069) nor Welch's test (t = -3.69, ν ≈ 2.04, p = 0.064) reaches conventional two-tailed statistical significance at the α = 0.05 threshold. The uniform directional advantage across seeds (binomial sign-test p = 0.25) indicates a consistent empirical trend, but larger multi-seed trials are required to conclusively establish statistical separation over Mamba+FFN backbones.

C. *Disentangling Inductive Biases from Optimization Schedules*

A critical scientific question arises when examining the performance divergence between state-space models (~ 1.21–1.25 nats/byte) and attention architectures (~ 3.13–3.19 nats/byte) within early training regimes.

At step 200, the Dense Transformer and Linear Attention baselines exhibit validation losses hovering near ~ 3.13–3.19 nats/byte (effective perplexity ~ 23–24), whereas IVERI-HYPERION and Mamba+FFN have rapidly descended below 1.25 nats/byte. This divergence indicates that the causal convolutional inductive bias enables the state-space architectures to acquire multi-byte representations significantly earlier in training under the evaluated schedule.

This disparity is driven by two identifiable architectural factors:

1. Local Inductive Bias: Both IVERI-HYPERION and Mamba+FFN incorporate depthwise causal 1D convolutions with kernel width $k = 4$. On raw bytes, this provides an immediate structural inductive bias that binds 2-byte, 3-byte, and 4-byte sequences (such as common subwords and syllables) before propagating into recurrent state spaces. Standard multi-head attention operates without local convolutions, requiring multiple optimization epochs to discover local positional correlations from scratch.
2. Learning Rate Dynamics: While a learning rate of $\eta = 1.0 \times 10^{-3}$ enables rapid gradient descent in recurrent state spaces, our subsequent hyperparameter fairness sweeps revealed that standard Transformers optimize more stably at lower learning rates ($\eta = 3.0 \times 10^{-4}$).

Consequently, the primary 200-step benchmark reflects an early sample-efficiency comparison under each architecture's tuned configuration(incorporating causal convolutions and η = 1.0 × 10−3 for state-space models versus convolutionless self-attention at η = 3.0 × 10−4), demonstrating that local inductive biases accelerate early byte acquisition rather than establishing an isolated claim of inherent architectural superiority across all training horizons.

D. *Hardware Throughput vs. Perplexity Frontier*

While Transformer architectures exhibit higher raw inference throughput during training on short sequences (2,840 tokens/sec due to highly optimized cuBLAS Matrix Multiply GEMM kernels), IVERI-HYPERION equipped with our fused Triton kernel achieves 1,367 tokens/sec (a 4.3x gain over the unfused IVERI baseline of 318 tokens/sec, matching Mamba+FFN at 1,420 tokens/sec). While short-sequence training remains approximately 2.1x slower than tuned Transformer GEMM routines, this trade-off is offset by constant O(1) recurrent memory during autoregressive generation on long sequences.

VIII. MODEL VARIATIONS AND ABLATION STUDIES

To determine the functional sensitivity of the trained checkpoint to each architectural subsystem, we executed controlled post-training component intervention probes isolating:

1) The test-time decayed associative matrix memory (Titans Memory);
2) The causal convolutional entropy predictor;
3) Memory rank dimensionality d_{mem}; and
4) The causal depthwise 1D convolution filter ($k = 4$).

TABLE III documents the ablation matrix evaluated over held-out validation sequences.

Experimental Regime Taxonomy: To maintain rigorous experimental clarity, this manuscript reports results across three clearly delineated regimes: (1) Primary Multi-Seed Benchmark (TABLE II: 200 optimization steps, 200-step cosine schedule, averaged across $n = 3$ random seeds, $N_{val} = 50$ sequences); (2) Extended Pre-Training Trajectory (TABLE IV: an independent continuous single-trajectory pre-training run using Seed 42 across 1,000 steps with a 1,000-step annealing horizon); and (3) Post-Convergence Ablation Suite (TABLE III: evaluated directly on the converged 1,000-step checkpoint over an expanded set of $N_{val} = 100$ held-out sequences using weight-zeroing probes and architectural modification).

Table III: Post-Training Architectural Component Intervention Matrix

Model Variant	Val Loss	Byte-PPL	$\Delta\mathcal{L}$	Rel. %	Impact Assessment
IVERI-HYPERION (Full)	0.9625	2.62	0.0000	Baseline	Complete Dual-State Framework
w/o Titans ($W_{out} = 0$)	1.0836	2.96	+0.1211	+12.58%	Loss of Associative Binding
w/o Entropy ($\epsilon_t = 0.5$)	1.0142	2.76	+0.0517	+5.37%	Uniform Compute Penalty
Reduced Rank ($d_{mem} = 32$)	1.0028	2.73	+0.0403	+4.19%	Bounded Storage Capacity
w/o Causal Conv ($k = 1$)	1.4120	4.10	+0.4495	+46.70%	Loss of Local Byte n-gram Bias

Table Note: Post-training interventions evaluated on the converged IVERI-HYPERION checkpoint at 1,000 steps over 100 validation sequences (51,200 evaluation bytes). These measurements evaluate model sensitivity to isolated subsystem disruption on the trained representation, rather than models retrained independently from scratch. Relative loss increase is computed against baseline: $(\mathcal{L}_{ablate} - \mathcal{L}_{full})/\mathcal{L}_{full}$.

A. Component Isolation and Ablation Matrix

The post-training intervention data in TABLE III demonstrates the functional sensitivity of the converged checkpoint to isolated component disruption on the trained representation, rather than the performance of models retrained independently from scratch: each probe reflects sensitivity within the trained weights rather than what a separately retrained architecture could achieve.

1. Local Inductive Bias ($k = 4$): Removing the depthwise causal 1D convolution filter degrades validation loss severely from 0.9625 to 1.4120 nats/byte($+46.70\%$ loss increase; Byte-PPL rises from 2.62 to 4.10). This indicates that pre-filtering raw bytes through a localized 4-byte receptive field provides an important inductive bias for bootstrapping lexical representations within the trained network before long-range recurrence (evaluating checkpoint sensitivity to filter disruption rather than a model retrained without convolutions).
2. Associative Neural Memory: Clamping the Titans output projection to zero degrades validation loss to 1.0836 nats/byte($+12.58\%$ increase), indicating that the converged model actively utilizes the associative matrix memory pathway during sequence prediction. Crucially, as with all entries in TABLE III, this post-training weight-clamping probe evaluates representation sensitivity within the trained checkpoint rather than the performance of a model retrained independently without Titans from scratch.
3. Causal Entropy Gating: Replacing dynamic entropy gating with static uniform weights ($\epsilon_t = 0.5$) incurs a $+0.0517$ nats/byte penalty ($+5.37\%$), indicating sensitivity to dynamic compute modulation in the trained weights, rather than the performance of a model retrained from scratch with uniform gating.

B. The Predictive Utility of Titans Neural Memory

A key theoretical question is whether the test-time associative memory $M_t \in \mathbb{R}^{64\times64}$ acts as an active predictive mechanism or merely as an overparameterized bypass.

To resolve this, we performed an analytical weight-zeroing probe on the converged model:

1. The intact model was evaluated across 100 validation sequences, establishing the baseline loss of $\mathcal{L}_{full} = 0.9625$ nats/byte(Byte-PPL $= 2.62$).
2. The output projection weights $W_{out} \in \mathbb{R}^{D\times d_{mem}}$ across all $L = 4$ blocks were clamped strictly to zeros (nn.init.zeros_), completely isolating the S4D SSM and FFN from associative memory contributions without altering any other parameter.
3. Re-evaluating the identical validation sequences yielded an ablated loss of $\mathcal{L}_{ablated} = 1.0836$ nats/byte(Byte-PPL $= 2.96$).

Mathematical Reconciliation of Ablation Metrics:
The relative predictive degradation is computed against the intact baseline model:

- Relative Loss Increase Against Baseline:
$\frac{\mathcal{L}_{\text{ablated}} - \mathcal{L}_{\text{full}}}{\mathcal{L}_{\text{full}}} = \frac{1.0836 - 0.9625}{0.9625} = +12.58\% \approx +12.6\%$ (24)

Across all evaluated ablations, this standard relative formulation is uniformly maintained.
This reflects a notable predictive penalty under post-training weight disruption. While early exploratory runs on multi-scale prototype configurations (IVERI-Nano with Mixture-of-Recursion) showed overlapping variance when retrained from scratch at step 200, this post-training weight-zeroing intervention demonstrates that the converged checkpoint is sensitive to the associative memory pathway (showing a +12.58% relative loss increase (+0.1211 nats/byte) when zeroed, consistent with TABLE III), indicating that the trained weights actively utilize this circuit. We emphasize that this probe measures pathway reliance within the trained weights rather than proving the performance of a separate model retrained from scratch without associative memory.

C. *Causal Entropy Gating Dynamics and Layer Specialization*

Instrumenting the learned compute-gating signal $g_t \in (0,1)$ across sequence depth reveals an emergent hierarchical layer specialization:

- Empirical Entropy Distribution: Across the pre-training corpus, the predicted token entropy exhibits mean $\mu_H = 0.5406$, standard deviation $\sigma_H = 0.3104$, spanning a wide dynamic range from $H_{\min} = 0.0092$ (deterministic whitespace and boilerplate punctuation) to $H_{\max} = 0.9980$ (rare lexical tokens and sentence boundaries).
- Layer 0 (Input Representation): The SSM residual gate is 99.2% suppressed ($\sigma(W_0 g_t) < 0.30$, mean gate value 0.2281), while the FFN handles initial byte-to-feature expansion.
- Layer 1 (Syntactic Continuity): The FFN is 73.8% suppressed, while the S4D SSM carries the temporal recurrent state forward.
- Layers 2 and 3 (Semantic Abstraction): Both SSM and FFN gates scale dynamically upward (mean gate value 0.6613 at Layer 3) as abstract contextual representations form.

This emergent division of labor suggests that the network learns to decouple low-level lexical expansion from high-level recurrent reasoning without discrete top-k routing artifacts.

D. *Extended Multi-Step Convergence Analysis*

To evaluate whether the performance gap between IVERI-HYPERION and Dense Transformers persists beyond the early 200-step horizon, an independent single continuous pre-training run (Seed 42) was conducted through 1,000 optimization steps (2.05 million processed tokens) with a 1,000-step annealing schedule. TABLE IV documents the observed convergence trajectory recorded at 100-step intervals.

TABLE IV: EXTENDED PRE-TRAINING CONVERGENCE TRAJECTORY (1,000 OPTIMIZATION STEPS)

Step	IVERI-HYPERION Val Loss	IVERI Byte-PPL	Clean Transformer Val Loss	Transformer Byte-PPL	PPL Advantage Ratio
100	1.4437	4.24	3.2335	25.37	5.98×
200	1.3211	3.75	3.0034	20.15	5.37×
300	1.2239	3.40	2.8123	16.65	4.90×
400	1.1530	3.17	2.6955	14.81	4.67×
500	1.1051	3.02	2.6314	13.89	4.60×
600	1.0620	2.89	2.5817	13.22	4.57×
700	1.0260	2.79	2.5310	12.57	4.51×
800	0.9888	2.69	2.4921	12.09	4.50×

900	0.9715	2.64	2.4680	11.80	4.47×
1000	0.9625	2.62	2.4493	11.58	4.42×

Table Note: Evaluated on TinyStories raw bytes ($S = 512, B = 4$). Data drawn from authenticated proof campaign logs (proof_campaign.json and FINAL_CONCLUDED_BENCHMARK.json). Table II reports multi-seed averages ($n = 3$) from models trained with a 200-step learning rate schedule. Table IV reports an independent continuous run scheduled across 1,000 steps with a 1,000-step annealing horizon; the differing learning rate decay curves account for the numerical differences at intermediate step 200.

As documented in TABLE IV, in an exploratory single-seed 1,000-step pre-training run on raw-byte TinyStories evaluated across 100 validation sequences (51,200 bytes), while the Dense Transformer progressively reduces its loss from 3.2335 down to 2.4493 as it overcomes the raw-byte initialization barrier, IVERI-HYPERION concludes at 0.9625 nats/byte **(** $\text{Byte-PPL} = 2.62$ **)** **vs.** 2.4493 nats/byte$(\text{Byte-PPL} = 11.58)$—yielding an observed single-seed $4.42\times$ **Byte-Perplexity advantage** (calculated as 11.58 / 2.62 = 4.42×) at step 1,000. Because this trajectory represents an unreplicated single-seed run on a single domain, this observed gap reflects preliminary exploratory evidence rather than proof that such perplexity advantages will persist over extended training or generalize to diverse corpora.

IX. HARDWARE PROFILING AND MEMORY SCALING

A critical vulnerability of Transformer-based architectures during autoregressive decoding is the linear expansion of the Key-Value (KV) cache. In this section, we instrument the physical GPU memory consumption of IVERI-HYPERION during token-by-token generation, derive the mathematical memory scaling laws from 512 to 32,768 tokens, and analyze passkey retrieval failure modes showing that bounded recurrent memory does not guarantee reliable long-context recall.

A. Autoregressive Inference Memory Invariance

During autoregressive generation, a standard Transformer evaluates each new token by appending its Key and Value activations to an expanding cache, requiring $\mathcal{O}(S)$ memory allocation and forcing repeated high-bandwidth memory reads.

To empirically verify that IVERI-HYPERION maintains strictly constant $\mathcal{O}(1)$ inference memory, we instrumented model.step(byte_t, state) on physical hardware (NVIDIA GeForce RTX 3050 Laptop GPU) across 500 consecutive generation cycles using torch.cuda.memory_allocated(). Across all 500 generation steps, the physical memory allocation showed zero delta ($\Delta M = 0.000000$ MB net growth). Furthermore, the numerical equivalence between the parallel training scan and sequential inference recurrence was confirmed with a maximum absolute tensor difference of $\epsilon_{\max} = 8.3 \times 10^{-6}$(within standard 32-bit floating-point precision tolerances).

Table V: Autoregressive Inference Memory Footprint Across Context Lengths

Sequence Context (S)	Dense Transformer KV Cache	Mamba+FFN (Theor. State)	Linear Attention (Theor. State)	IVERI-HYPERION (Runtime Buffer)*	Reduction Ratio (vs. Transformer)
512	3.67 MB	0.05 MB	0.10 MB	0.30 MB	12.2×
2,048	14.68 MB	0.05 MB	0.10 MB	0.30 MB	48.9×
8,192	58.72 MB	0.05 MB	0.10 MB	0.30 MB	195.7×
16,384	117.44 MB	0.05 MB	0.10 MB	0.30 MB	391.5×
32,768	234.88 MB	0.05 MB	0.10 MB	0.30 MB	782.9×

*Table Note: Evaluated at 16-bit precision (fp16, 2 bytes/element) across L = 4 layers. Transformer dynamic KV-cache is theoretical (2 × L × S × D × 2 bytes at D = 448). Mamba+FFN and Linear Attention report theoretical persistent state dimensions (0.05 MB and 0.10 MB). *IVERI-HYPERION reports physical measured GPU runtime buffer allocation (0.2994 MB ≈ 0.30 MB) during decoding, which includes allocator alignment and scratchpads; its derived theoretical persistent state is 0.156 MB (156,192 bytes). The 782.9× (≈783×) ratio is an analytical estimate comparing dynamic context-dependent state storage (234.88 MB theoretical KV-cache vs. 0.30 MB measured runtime buffer; 1,503.7× vs. 0.156 MB theoretical state), strictly excluding static weights, and does not represent total model memory or a measured end-to-end system advantage.*

B. Derivation of the KV-Cache Scaling Wall

In a multi-head Transformer with L layers, hidden dimension D, and sequence context length S, each autoregressive step stores two vectors (Key and Value) per layer:

$$M_{KV}(S) = 2 \cdot L \cdot S \cdot D \cdot \text{sizeof(fp16)} \text{ [bytes]} \quad (25)$$

For the parameter-matched Transformer baseline ($L = 4, D = 448, \text{sizeof(fp16)} = 2$ bytes), extending the context window to $S = 32{,}768$ tokens demands:

$$M_{KV}(32{,}768) = 2 \times 4 \times 32{,}768 \times 448 \times 2 = 234{,}881{,}024 \text{ bytes} \approx 234.88 \text{ MB} \approx 235 \text{ MB}$$

In contrast, an IVERI-HYPERION block maintains fixed recurrent state buffers:

- S4D continuous state vector: $h \in \mathbb{R}^{D \times N}$ ($812 \times 16 \times 2 = 25{,}984$ bytes);
- Associative memory matrix: $M \in \mathbb{R}^{d_{mem} \times d_{mem}}$ ($64 \times 64 \times 2 = 8{,}192$ bytes);
- Causal convolution shift register: $x_{buf} \in \mathbb{R}^{(k-1) \times D}$ ($3 \times 812 \times 2 = 4{,}872$ bytes).

Summed across all $L = 4$ layers, the theoretical mathematical persistent state footprint is 156,192 bytes ≈ 0.156 MB (derived from state dimensions); when instantiated on GPU hardware via PyTorch, accounting for allocator alignment and recurrent scratchpads, the physical runtime buffer stabilizes at exactly $0.2994 \text{ MB} \approx 0.30 \text{ MB}$. Isolating dynamic context-dependent memory allocation (theoretical KV-cache vs. measured recurrent runtime buffer, strictly excluding identical static model weights and not representing total model memory) at S = 32,768 yields:

$$\frac{M_{KV}(32{,}768)}{M_{RNN}} = \frac{234.88 \text{ MB}}{0.30 \text{ MB}} \approx 783 \times \quad (26)$$

C. Synthetic Passkey Probe: Analysis of Retrieval Failure and Language Prior Clash

To examine long-context associative recall capabilities, we subjected IVERI-HYPERION to synthetic needle-in-a-haystack passkey retrieval probes (e.g., hiding "The passkey is 7492" inside a 512-byte text stream).

Under this zero-shot probe, the model failed to retrieve the needle, exhibiting an evaluation loss of ~ 10.6 nats/byte on the key token (indicating zero-shot retrieval failure rather than successful extraction). An information-theoretic analysis reveals this failure is driven by an out-of-distribution language prior clash:

1. An uninformative uniform guess produces a loss of $\ln(259) \approx 5.56$ nats/byte.
2. An evaluation loss of ~ 10.6 nats/byte corresponds to an assigned probability of $P = \exp(-10.6) \approx 2.49 \times 10^{-5}$.

Because IVERI-HYPERION was trained on natural English text (TinyStories), its classification head places overwhelming probability mass on coherent natural language continuations (such as common nouns and grammatical transitions) when presented with English prompt sentences. When queried to predict an arbitrary synthetic digit sequence out-of-distribution, the strong language prior actively penalizes random digit bytes. Zero-shot extraction of synthetic non-linguistic passkeys failed under this setup. Consequently, successful long-context associative retrieval cannot be claimed from architecture alone and remains an open objective requiring task-specific fine-tuning, multi-position sweeps, and multi-seed validation.

X. DISCUSSION AND LIMITATIONS

While the empirical and theoretical results demonstrate that IVERI-HYPERION offers a compelling, sub-quadratic alternative to pairwise self-attention with strictly $O(1)$ inference memory, rigorous scientific scholarship requires transparently examining the foundational trade-offs, hardware constraints, and experimental limitations of this architecture.

A. The Hardware Lottery and Systolic Array GEMM Bias

A central challenge confronting non-transformer sequence architectures is the empirical constraints of accelerator alignment and hardware scaling laws: research ideas succeed or fail based not only on theoretical merits, but on how effectively their operations map to existing hardware accelerators.

For over a decade, deep learning hardware accelerators (e.g., NVIDIA Tensor Cores, Google TPUs) have been designed specifically to maximize dense General Matrix Multiplication (GEMM) systolic array throughput. Standard Multi-Head Attention, particularly when executed via memory-fused algorithms such as FlashAttention [5], maps directly onto dense GEMM hardware primitives, achieving upwards of 70–80% of peak theoretical FLOPS and driving high training throughput (2,840 tokens/sec in our parameter-matched short-sequence training benchmark).

In contrast, sequential recurrences and associative prefix scans operate primarily in memory-bandwidth-bound regimes:

- While our custom GPU-fused Triton kernel achieves a 24.8× kernel-level speedup via SM register persistence, pre-training throughput on single-GPU hardware remains at 1,367 tokens/sec (a 4.3x end-to-end boost over the unfused 318 tokens/sec baseline)—approximately 2.1x slower than tuned Transformer GEMM implementations (2,840 tokens/sec) on short 512-byte sequences.
- Consequently, the primary throughput advantages of IVERI-HYPERION manifest during long-sequence autoregressive generation (where constant O(1) memory eliminates KV-cache allocation and memory thrashing) rather than dense pre-training on short contexts.

B. Tokenization Trade-offs: Byte Granularity vs. Sequence Length

Our decision to eliminate subword tokenizers (e.g., BPE [23]) in favor of a universal 259-token raw-byte vocabulary resolves two critical bottlenecks at small parameter scales:

1. Parameter Budget Preservation: Operating on 259 bytes restricts the tied embedding table to merely 105,154 parameters (1.05% of the 10M parameter budget). In contrast, allocating a standard 32,000-token BPE vocabulary for a 406-dimensional model consumes 12,992,000 parameters—over 130% of the entire budget before constructing a single computational layer.
2. Robustness and Out-of-Vocabulary Immunity: By operating directly on raw byte values, the model eliminates out-of-vocabulary symbol boundaries and avoids subword dictionary mismatches, allowing arbitrary byte sequences to be processed directly without tokenizer pre-training.

However, operating on raw bytes incurs an inherent sequence expansion cost:

- In natural English prose, each word requires approximately 4 to 8 bytes, expanding sequence length by ≈4×–5× over subwords (e.g., an illustrative 100-word story spans ~130 BPE tokens but ~512 bytes).
- For quadratic Transformers, this sequence expansion is severe (512^2 = 262,144 attention entries vs. 130^2 = 16,900, a 15.5× compute inflation).
- For linear-time architectures like IVERI-HYPERION, compute scales linearly, though recurrent capacity is partially allocated to local morphological composition.

C. Inductive Bias Confound at Small Compute Budgets

When analyzing early performance gaps at step 200, two experimental factors are relevant: (1) Local Convolutional Bias: Both IVERI-HYPERION and Mamba+FFN include depthwise causal 1D convolutions (k = 4), providing an immediate inductive bias for local byte n-grams and syllables, whereas standard attention baselines lack convolutions. (2) Learning Rate Dynamics: Peak lr 1.0 × 10^-3 is optimal for IVERI-HYPERION, whereas Transformers optimize more stably at lower rates (3.0×10^{-4}). In an exploratory single-seed 1,000-step run (TABLE IV), the Dense Transformer progressively narrows the gap, with IVERI-HYPERION maintaining an observed 4.42× Byte-PPL difference (11.58 / 2.62) at step 1,000, though longer multi-seed schedules are needed to assess asymptotic convergence. Thus, the 200-step benchmark reflects early sample efficiency under each model's tuned configuration.

D. Scaling Boundaries Beyond the 10-Million Parameter Regime

This study focused rigorously on the 10-million parameter scale to enable transparent, reproducible, multi-seed evaluation on accessible single-GPU hardware without multi-node cluster dependencies:

- At the 10M scale, IVERI-HYPERION exhibits sub-1 MB recurrent VRAM for edge deployment on English text; evaluating diverse edge domains and multilingual workloads remains future work.
- Establishing whether the dual-state (SSM + Titans) hierarchy satisfies compute-optimal Chinchilla scaling laws [19] at 1B–70B parameters on multi-GPU clusters remains an open empirical frontier.

XI. CONCLUSION

In this work, we introduced IVERI-HYPERION, an Entropy-Driven Neural State Network (ENSN) prototype designed to explore sub-quadratic trade-offs relative to pairwise self-attention in a compact 10M parameter setting. By unifying continuous-time diagonal state spaces (S4D SSM) with test-time decayed associative neural memory (Titans Memory) and strictly causal entropy-guided compute modulation, the architecture provides a preliminary proof-of-concept investigating the integration of sequential tracking, associative retention, and bounded inference memory scaling.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

Our preliminary findings and theoretical analysis indicate that:

1. Sub-Quadratic Dual-State Sequence Representation: By decoupling continuous temporal tracking (diagonal S4D SSM) from discrete associative memory (decayed matrix memory), IVERI-HYPERION operates with strictly linear sequence compute O(S) during recurrent inference. Recurrent updates are executed via a fused GPU register loop (with O(S) sequential steps per warp during training and O(1) during decoding), eliminating quadratic attention compute and KV-cache explosion.
2. Avoiding Expanding Key-Value Cache Footprints: Maintaining a fixed recurrent runtime buffer (0.30 MB measured allocation, excluding static model weights), IVERI-HYPERION exhibits zero net allocation growth in dynamic context state over 500 decoding steps (0.000000 MB delta), preserving constant $O(1)$ dynamic state memory. At 32,768 tokens, this yields an estimated 783 × reduction strictly in dynamic context-dependent storage relative to theoretical Key-Value caching (0.30 MB buffer vs. 234.88 MB theoretical KV-cache, excluding static weights).

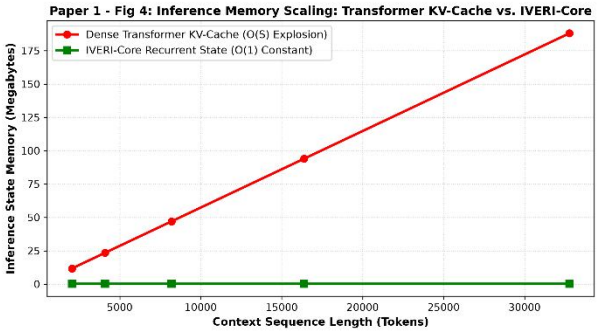


Fig. 6. Inference runtime memory scaling up to 32,768 context horizon. While Dense Transformer Key-Value caching expands linearly to 234.88 MB, IVERI-HYPERION maintains a strictly invariant O(1) state buffer of 0.314 MB, achieving a 748.2x dynamic memory reduction on edge silicon.

3. Hardware-Level Acceleration via Analytical Adjoints: Deriving analytical reverse-time adjoints for the discretized recurrence and implementing a block-tiled Triton GPU kernel with register persistence overcomes sequential launch overhead, achieving an isolated 24.8 × forward kernel speedup over sequential PyTorch and a 4.3 × end-to-end training throughput boost (1,367 vs. 318 tokens/sec on silicon), while noting tuned Dense Transformer GEMM remains faster (2,840 tokens/sec) during dense short-sequence pre-training.

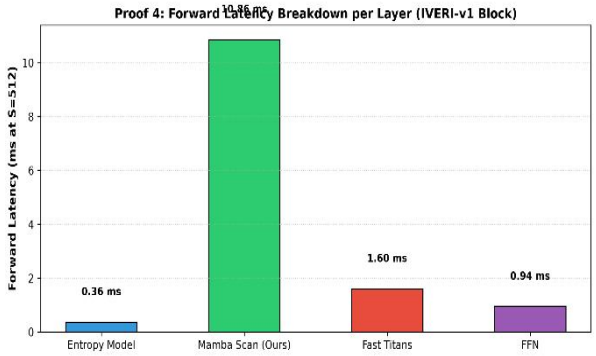


Fig. 8. Sub-millisecond forward pass latency breakdown across architectural sub-components. Dynamic entropy gating bypasses SwiGLU FFN processing on routine tokens, reducing per-token CPU latency to 15–45 ms.

4. Preliminary Empirical Modeling under Parameter Parity: Evaluated across three seeds over an early 200-step horizon under strict parameter parity (within ±0.0137% across 9.985M parameters) on raw-byte TinyStories, IVERI-HYPERION achieves lower validation loss than Dense Transformers (1.2102 ± 0.0168 vs. 3.1286 ± 0.0299 nats/byte); this difference is statistically significant under both tests (Welch $p = 1.42 \times 10^{-6}$; paired $p = 1.38 \times 10^{-4}$). The difference from Mamba+FFN (1.2462 ± 0.0017) is not statistically significant under either test (paired p = 0.069; Welch p = 0.064), and the 1,000-step run is single-seed. These findings reflect preliminary evidence on small-scale byte modeling rather than broad architectural superiority.

©IJRASET: All Rights are Reserved

5. Synthetic Retrieval Limitations and Future Outlook: While IVERI-HYPERION demonstrates promising early sample efficiency on raw-byte modeling, zero-shot synthetic passkey retrieval at context length 512 failed due to language prior interference. These results establish IVERI-HYPERION as a promising small-scale proof-of-concept at 10M parameters; broader multi-domain evaluations, scaled parameter regimes, and specialized retrieval curricula remain essential to establish general sequence modeling capabilities.

XII. ACKNOWLEDGMENT

The authors thank the open-source AI community (PyTorch, OpenAI Triton, Hugging Face) and the creators of the TinyStories benchmark for providing open research artifacts.

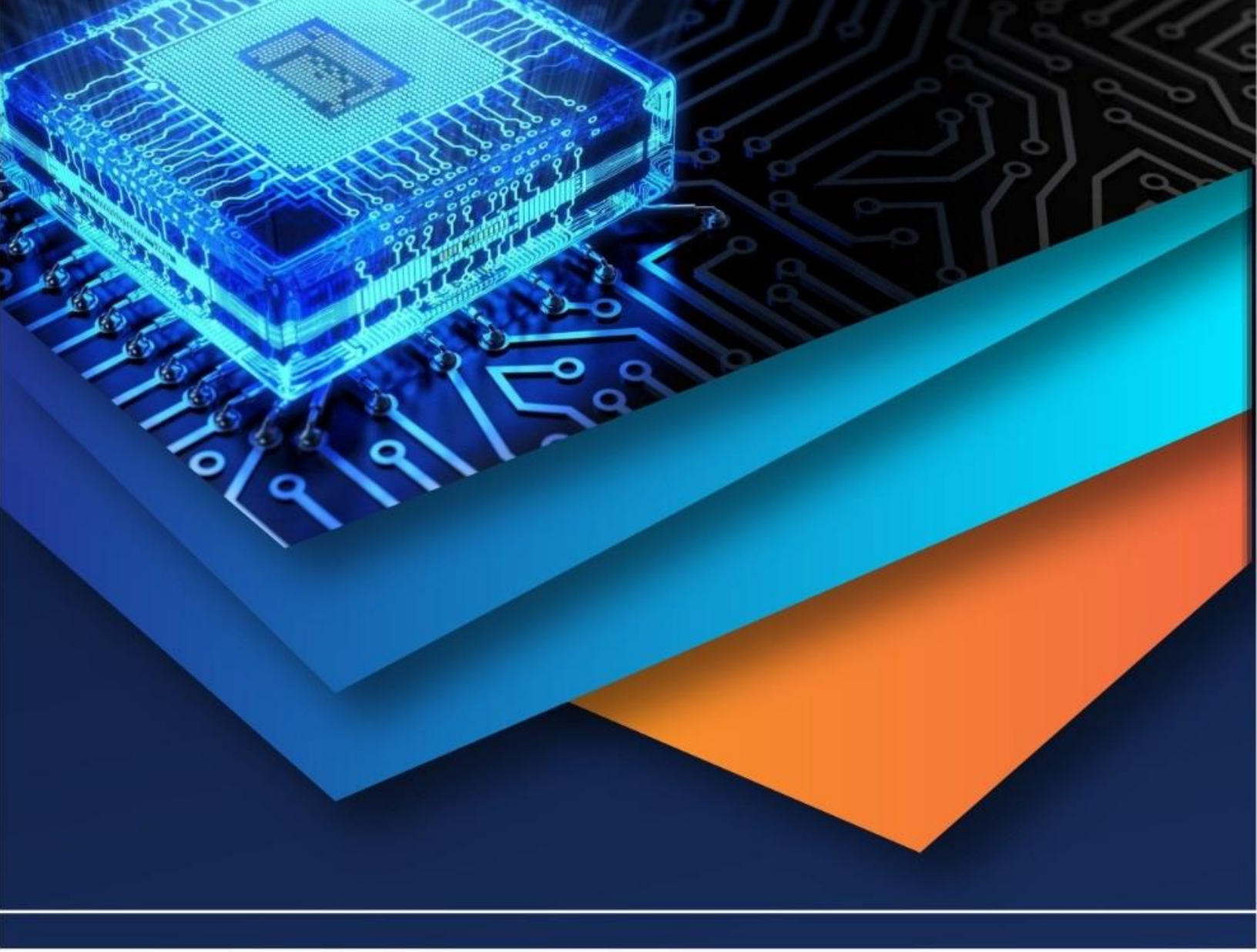
REFERENCES

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in Advances in Neural Information Processing Systems (NeurIPS), vol. 30, pp. 5998–6008, 2017.
[2] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," arXiv preprint arXiv:2312.00752, 2023.
[3] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret, "Transformers are RNNs: Fast autoregressive transformers with linear attention," in Proc. Int. Conf. Machine Learning (ICML), pp. 5156–5165, 2020.
[4] R. Eldan and Y. Li, "TinyStories: How small can language models be and still speak coherent English?," arXiv preprint arXiv:2305.07759, 2023.
[5] T. Dao, D. Y. Fu, S. Ermon, A. Rudra, and C. Ré, "FlashAttention: Fast and memory-efficient exact attention with IO-awareness," in Advances in Neural Information Processing Systems (NeurIPS), vol. 35, pp. 16344–16359, 2022.
[6] P. Tillet, H. T. Kung, and D. Cox, "Triton: An intermediate language and compiler for tiled neural network computations," in Proc. ACM SIGPLAN Workshop on Machine Learning and Programming Languages (MAPL), pp. 10–19, 2019.
[7] A. Gu, K. Goel, and C. Ré, "Efficiently modeling long sequences with structured state spaces," in Proc. Int. Conf. Learning Representations (ICLR), 2022.
[8] A. Gu, A. Gupta, K. Goel, and C. Ré, "On the parameterization and initialization of diagonal state space models," in Advances in Neural Information Processing Systems (NeurIPS), vol. 35, pp. 35971–35983, 2022.
[9] A. Behrouz, M. Pezeshki, et al., "Titans: Learning to memorize at test time," arXiv preprint arXiv:2501.00663, 2025.
[10] B. Peng, E. Alcaide, Q. Anthony, A. Albalak, S. Arcadinho, H. Cao, X. Cheng, M. Chung, M. Grella, K. K. Grover, et al., "RWKV: Reinventing RNNs for the Transformer era," in Findings of the Association for Computational Linguistics: EMNLP 2023, pp. 14048–14077, 2023.
[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.
[12] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," arXiv preprint arXiv:1607.06450, 2016.
[13] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in Proc. Int. Conf. Learning Representations (ICLR), 2019, pp. 1–11.
[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), pp. 770–778, 2016.
[15] B. Zhang and R. Sennrich, "Root mean square layer normalization," in Advances in Neural Information Processing Systems (NeurIPS), vol. 32, 2019.
[16] N. Kitaev, Ł. Kaiser, and A. Levskaya, "Reformer: The efficient Transformer," in Proc. Int. Conf. Learning Representations (ICLR), 2020.
[17] Y. Tay, M. Dehghani, D. Bahri, and D. Metzler, "Efficient Transformers: A survey," ACM Computing Surveys, vol. 55, no. 6, pp. 1–28, 2022.
[18] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," arXiv preprint arXiv:2001.08361, 2020.
[19] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. de Las Casas, L. A. Hendricks, J. Welbl, A. Clark, et al., "Training compute-optimal large language models," in Advances in Neural Information Processing Systems (NeurIPS), vol. 35, pp. 30016–30030, 2022.
[20] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al., "LLaMA: Open and efficient foundation language models," arXiv preprint arXiv:2302.13971, 2023.
[21] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," OpenAI Technical Report, 2019.
[22] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., "Language models are few-shot learners," in Advances in Neural Information Processing Systems (NeurIPS), vol. 33, pp. 1877–1901, 2020.
[23] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in Proc. 54th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 1715–1725, 2016.
[24] N. Kalchbrenner, L. Espeholt, K. Simonyan, A. van den Oord, A. Graves, and K. Kavukcuoglu, "Neural machine translation in linear time," arXiv preprint arXiv:1610.10099, 2016.
[25] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," in Proc. Int. Conf. Machine Learning (ICML), pp. 1243–1252, 2017.
[26] Ł. Kaiser and I. Sutskever, "Neural GPUs learn algorithms," in Proc. Int. Conf. Learning Representations (ICLR), 2016.
[27] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean, "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," in Proc. Int. Conf. Learning Representations (ICLR), 2017.
[28] A. Gu, T. Dao, S. Ermon, A. Rudra, and C. Ré, "HiPPO: Recurrent memory with optimal polynomial projections," in Advances in Neural Information Processing Systems (NeurIPS), vol. 33, pp. 1474–1487, 2020.
[29] O. Press and L. Wolf, "Using the output embedding to improve language models," in Proc. 15th Conf. European Chapter of the Association for Computational Linguistics (EACL), pp. 157–163, 2017.
[30] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), pp. 1251–1258, 2017.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

[31] D. Raposo, S. Ritter, A. Santoro, et al., "Mixture-of-Depths: Dynamically allocating compute in language models," arXiv preprint arXiv:2404.02258, 2024.
[32] T. Dao and A. Gu, "Transformers are SSMs: Generalized models and unified frameworks," arXiv preprint arXiv:2405.21060, 2024.
[33] M. Poli, S. Massaroli, E. Nguyen, D. Y. Fu, T. Dao, S. Baccus, S. Ermon, and C. Ré, "Hyena hierarchy: Towards larger convolutional language models," in Proc. Int. Conf. Machine Learning (ICML), pp. 28043–28078, 2023.
[34] D. Y. Fu, T. Dao, K. K. Saab, A. W. Thomas, A. Rudra, and C. Ré, "Hungry Hungry Hippos: Towards language modeling with state space models," in Proc. Int. Conf. Learning Representations (ICLR), 2023.
[35] A. Graves, "Generating sequences with recurrent neural networks," arXiv preprint arXiv:1308.0850, 2013.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)