

State Space Models, S4, S6, and Mamba

Beyond the Transformer

Your Name

Your Affiliation

December 1, 2025

Source: Mamba (arXiv:2312.00752v2) and S4/SSM literature

Motivation: The Limits of Transformers

Core limitations of the Attention mechanism:

- **Training Cost:** Quadratic complexity $O(L^2D)$ with sequence length L
- **Inference Cost:** Linear complexity $O(LD^2)$ and $O(LD)$ memory for KV cache
- **Efficiency at Scale:** Severe memory and latency bottlenecks for very long sequences ($L > 50K$)

Goal: Achieve linear-time sequence modeling with:

- **Long-Range Reasoning:** Capture global dependencies effectively
- **Efficient Inference:** Constant-time per token generation, minimal state
- **Parallel Training:** Maintain GPU-friendly parallelism

The Foundation: Continuous State Space Models

SSMs are dynamical systems relating input $x(t)$ to output $y(t)$ via hidden state $h(t)$.

Continuous-Time SSM (LTI - Linear Time-Invariant):

$$\frac{dh(t)}{dt} = Ah(t) + Bx(t) \quad (1)$$

$$y(t) = Ch(t) + Dx(t) \quad (\text{usually } D = 0) \quad (2)$$

- $x(t) \in \mathbb{R}^D$: Input Signal
- $h(t) \in \mathbb{R}^N$: State Vector (typically $N = 16$ in Mamba)
- $y(t) \in \mathbb{R}^D$: Output
- $A \in \mathbb{R}^{N \times N}$: State transition matrix
- $B \in \mathbb{R}^{N \times D}$: Input-to-state projection
- $C \in \mathbb{R}^{D \times N}$: State-to-output projection

Discretization required to map continuous dynamics to discrete tokens (x_k, h_k, y_k)

Discretization and the Convolution Kernel

Zero-Order Hold Discretization over timestep Δ :

$$h_k = \bar{A}h_{k-1} + \bar{B}x_k \quad (3)$$

where \bar{A} and \bar{B} are functions of (A, B, Δ)

Training Parallelism via Convolution:

Unrolling the recurrence yields:

$$y_k = \sum_{j=1}^k K_{k-j} x_j \quad \text{where} \quad K_j = C \bar{A}^j \bar{B} \quad (4)$$

- $K \in \mathbb{R}^{D \times L}$: Structured convolution kernel
- Training: $y = K * x$ computable via FFT in $O(L \log L)$
- Convolution enables **parallel training**, recurrence enables **fast inference**

S4: Structured State Space Sequence Model

S4 overcomes challenges in learning stable SSM parameters and computing K efficiently.

Key Innovations:

- **HiPPO Matrix A :** Initialization based on High-order Polynomial Projection Operator ensures optimal memory compression and stability
- **Structured A :** Low-Rank + Diagonal or purely Diagonal structure allows fast kernel computation without $O(LN^2)$ complexity
- **Parallel Training:** Uses $O(L \log L)$ convolution via FFT for GPU efficiency

Key Limitation: S4 is **Linear Time-Invariant (LTI)**

The filter K is *fixed* regardless of input content — same processing for all sequences

The Need for Selectivity (S6 Motivation)

LTI limitation is critical for discrete modalities like language:

- **Lack of Content-Awareness:** Attention selectively attends based on content; S4 applies fixed filter K
- **Memory Rigidity:** S4 cannot dynamically forget irrelevant information or focus on critical tokens

Solution: Make SSM parameters **input-dependent**
(selective)

Transform from **LTI** \rightarrow **LTV** (Linear Time-Varying)

This is the **crucial innovation** enabling language modeling performance!

S6: The Selective State Space Model

Core Idea: Make discretization step-size Δ and matrices B, C **functions of input** x_k

Input x_k generates selective parameters:

$$(\Delta_k, B_k, C_k) = \text{Project}(x_k) \quad (5)$$

Yielding time-varying discrete recurrence:

$$h_k = \hat{A}_k h_{k-1} + \hat{B}_k x_k \quad \text{where} \quad \hat{A}_k = \exp(\Delta_k A) \quad (6)$$

Selective Mechanism:

- Small $\Delta_k \implies \hat{A}_k \approx I + \Delta_k A \rightarrow$ **forget quickly**
- Large $\Delta_k \implies \hat{A}_k \approx \exp(\Delta_k A) \rightarrow$ **remember long-term**

The model learns to *compress* or *expand* time based on content!

S6 Discretization (Zero-Order Hold Formulas)

ZOH transformation from (A, B, Δ_k) to (\hat{A}_k, \hat{B}_k) :

$$\hat{A}_k = \exp(\Delta_k A) \quad (7)$$

$$\hat{B}_k = (\Delta_k A)^{-1}(\exp(\Delta_k A) - I)B \quad (8)$$

Simplification for diagonal A (Mamba's choice):

- Matrix inverse $(\Delta_k A)^{-1}$ becomes element-wise division
- Matrix exponential $\exp(\Delta_k A)$ becomes element-wise exponential
- Enables fast, numerically stable computation

Key Point: Δ_k, B_k, C_k are **selective** (input-dependent)
 A remains **fixed/learned** (not input-dependent)

Parallel Scan: Making LTV Training Possible

LTV recurrence $h_k = \hat{A}_k h_{k-1} + \hat{B}_k x_k$ cannot use FFT convolution (no fixed K)

Solution: Parallel Scan (Prefix Sum) Algorithm

- ① **Affine Operator:** Define $T_k(h) = \hat{A}_k h + \hat{B}_k x_k$ for each step
- ② **Composition:** State $h_L = T_L \circ T_{L-1} \circ \cdots \circ T_1(h_0)$
- ③ **Associativity:** $(T_3 \circ T_2) \circ T_1 = T_3 \circ (T_2 \circ T_1)$

Binary operator:

$$(A_2, b_2) \circ (A_1, b_1) = (A_2 A_1, A_2 b_1 + b_2) \quad (9)$$

Associativity \rightarrow compute all prefixes in $O(\log L)$ depth on parallel hardware
Hardware-aware kernel fuses discretization + scan in one optimized step

Mamba Block Architecture

M-shaped gated block replaces Attention + MLP:

- ① **Expand:** $\tilde{\mathbf{x}} = \text{Linear}(\mathbf{x}) \in \mathbb{R}^{2ED}$
- ② **Split:** $\mathbf{u}, \mathbf{v} \in \mathbb{R}^{ED}$
- ③ **S6 Layer:** $\mathbf{s} = \text{S6}(\mathbf{v})$
 \mathbf{v} computes (Δ, B, C)
- ④ **Gate:** $\mathbf{a} = \text{SiLU}(\mathbf{u})$
- ⑤ **Combine:** $\mathbf{z} = \mathbf{a} \odot \mathbf{s}$
- ⑥ **Project:** $\mathbf{y} = \text{Linear}(\mathbf{z})$

Key Features:

- **Content-aware** via $\mathbf{v} \rightarrow (\Delta, B, C)$
- **Non-linearity** via $\text{SiLU}(\mathbf{u})$
- **Per-channel** S6 (D independent SSMs)
- Similar to GLU structure

Mamba Inference (Recurrent Mode)

Constant-time per token update (given previous state h_{t-1}):

- 1 **Project:** Compute $\mathbf{u}_t, \mathbf{v}_t$ from \mathbf{x}_t
- 2 **Selective Params:** $\Delta_t, B_t, C_t = f(\mathbf{v}_t)$ via linear + softplus
- 3 **Discretize:** Compute \hat{A}_t, \hat{B}_t using ZOH (element-wise for diagonal A)
- 4 **State Update:** $\mathbf{h}_t = \hat{A}_t \odot \mathbf{h}_{t-1} + \hat{B}_t \odot \mathbf{v}_t \quad O(DN)$ ops
- 5 **Output:** $\mathbf{s}_t = C_t \mathbf{h}_t$
- 6 **Final:** $\mathbf{y}_t = \text{Linear}(\text{SiLU}(\mathbf{u}_t) \odot \mathbf{s}_t)$

Memory & time per step: $O(DN)$
Constant w.r.t. sequence length L !
($N \approx 16 \implies$ very fast)

Why Mamba Achieves Performance and Efficiency

- **Unconstrained Speed:** $O(1)$ per token inference vs. $O(L)$ for Transformer KV cache
- **Training Parallelism:** $O(\log L)$ depth Parallel Scan for fast GPU training
- **Content-Awareness:** Selectivity handles discrete data (language) unlike LTI SSMs
- **Hardware Optimization:** Fused kernel (Discretization + Scan) maximizes SRAM usage
 - 20-40 \times faster inner kernels vs. standard implementations

Empirical Result:

Up to **5 \times higher inference throughput** than optimized Transformers
(e.g., 2.8B Mamba vs. 2.7B Transformer)

Comparison: Attention vs. SSMs vs. Mamba

Feature	Transformer	S4	Mamba
Training Complexity	$O(L^2)$	$O(L \log L)$	$O(L \log L)$
Inference Cost/Token	$O(L)$ (KV)	$O(1)$	$O(1)$
Content-Aware?	Yes	No (LTI)	Yes (Selective)
Memory (Inference)	$O(LD)$	$O(DN)$	$O(DN)$
Language Performance	Excellent	Poor/Mixed	Excellent

Key Advantage:

Mamba achieves **content-awareness** + $O(1)$ **inference** simultaneously

Practical Applications of Mamba

Mamba excels where **long context** + **fast inference** are critical:

- **Language Modeling:** Competitive with Transformers at $3\text{-}5\times$ inference speed
- **Genomics/DNA:** Handles million-token sequences infeasible for Attention
- **Time-Series Forecasting:** Complex long-range dependencies in financial/climate data
- **Audio/Waveform:** Traditional SSM strength + added selectivity

Scale Advantage: Processes contexts that are **impossible** for standard Attention

Current Limitations and Future Research

- **Engineering Complexity:** Requires custom CUDA/C++ kernels for advertised speedups
 - Not a simple drop-in replacement
- **Training Stability:** SSM training needs careful parameterization (though Mamba helps)
- **Fixed A Matrix:** Only B, C, Δ are selective
 - Can making A selective improve further? *Open question*
- **Vision/Multimodality:** Extending 1D Selective SSM to 2D/3D data is active research

Implementation barrier is high, but community adoption is growing

Summary: The Mamba Breakthrough

Evolution of State Space Models:

- **SSM**: Foundational continuous dynamics model for sequences
- **S4**: Structured A enables parallel training via $O(L \log L)$ convolution
- **S6**: Added **selectivity** (Δ, B, C input-dependent) for content-awareness
- **Mamba**: First practical model combining:
 - Efficiency of **recurrent state** ($O(1)$ inference)
 - Expressivity of **content-aware dynamics** (selective SSM)

Primary Takeaway:

Transformer-level performance with $O(1)$ inference cost

References & Further Reading

- Gu, A., & Dao, T. (2023). *Mamba: Linear-Time Sequence Modeling with Selective State Spaces*. arXiv:2312.00752v2
- Gu, A., Goel, K., & Ré, C. (2021). *Efficiently Modeling Long Sequences with Structured State Spaces (S4)*
- Related work: LSSL, S3, S5, H3
- Blelloch, G. E. (1990). *Prefix Sums and Their Applications*

Questions?