[Video]

# Deep State Space Models

Karan Bania, Yash Bhisikar

S4 S5 S6

27th September 2024

# Hype around SSMs

**Albert Gu**
@_albertgu

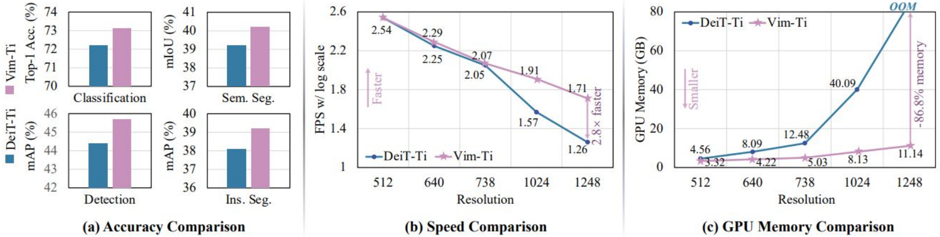
Quadratic attention has been indispensable for information-dense modalities such as language... until now.

Announcing Mamba: a new SSM arch. that has linear-time scaling, ultra long context, and most importantly--outperforms Transformers everywhere we've tried.

With @tri_dao 1/

**Aleksandar Botev**
@botev_mg

We present Griffin: A hybrid model mixing a gated line local attention. This combination is extremely effectiv the efficient benefits of linear RNNs and the expressiv transformers. Scaled up to 14B!
https://arxiv.org/abs/2402.19427

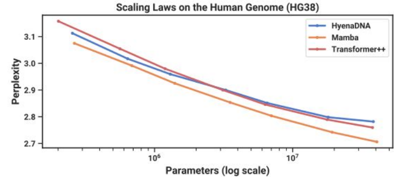


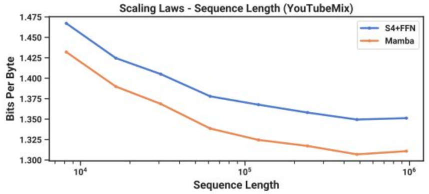

ision Mamba: https://arxiv.org/abs/2401.09417

**Tri Dao**
@tri_dao

With @_albertgu, we're collaborating with @togethercompute and @cartesia_ai and releasing a Mamba 3B model trained on 600B tokens on the SlimPajama dataset (Mamba-3B-SlimPJ). It's among the strongest 3B models, matching the performance of strong Transformers (BTLM-3B).
1/

| | Mamba-3B-SlimPJ | BTLM-3B-8K | StableLM-3B-4E1T |
|---|---|---|---|
| Number of params | 2.77B | 2.65B | 2.80B |
| Number of tokens | 604B | 627B | 4T |
| **Training FLOPs** | 1.01E22 | 1.22E22 | 8.33E22 |
| BoolQ | 71.0 | 70.0 | 75.5 |
| PIQA | 78.1 | 77.2 | 79.8 |
| HellaSwag | 71.0 | 69.8 | 73.9 |
| WinoGrande | 65.9 | 65.8 | 66.5 |
| ARC-e | 68.2 | 66.9 | 67.8 |
| ARC-c | 41.7 | 37.6 | 40.0 |
| OpenBookQA | 39.8 | 40.4 | 39.6 |
| RACE-high | 36.6 | 39.4 | 40.6 |
| TruthfulQA | 34.3 | 36.0 | 37.2 |
| MMLU | 26.2 | 28.1 | 44.2 |
| **Avg accuracy** | 53.3 | 53.1 | 56.5 |




Mamba on Language, DNA and audio data:
https://arxiv.org/abs/2312.00752

5:3

Paving the way to efficient architectures: StripedHyena-7B, open source models offering a glimpse into a world beyond Transformers

DECEMBER 8, 2023 • BY TOGETHER

And many more...

# Overview

- History: *RNNs, Transformers and Problems* (***Yash***)
- SSM Fundamentals & S4[1]: *Efficiently Modeling Long Sequences with Structured State Spaces* (***Karan***)
- S5[2]: *Simplified State Space Layers for Sequence Modeling* (***Yash, 11th Oct***)
- Mamba[3]: *Linear Time Sequence Modeling with Selective State Spaces* ***(Karan)***
- Event-SSM[4]: Scalable Event-by-event processing of Neuromorphic sensory signals with Deep State-Space Models (***Yash, 11th Oct***)

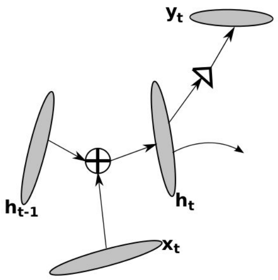
Most images have been taken from [1], [2] and [3] and this DeepMind talk in UCL.
(psst, We got to work with Mamba's CUDA kernels!)

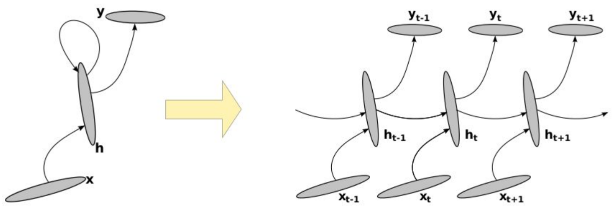
# Preface

- These models and many slides have a lot of **math**!
- They also involve a lot of CS fundamentals, and **out-of-the-box programming**.
- We think that regardless of Machine Learning these papers should also be treated as amazing **thought experiments**, it's definitely not the case that these models are wack at ML xD; it's just that they fully deserve all the hype around them!
- These are truly revolutionary architectures, let's start with "Is attention is all we need?"

# History: RNNs & LSTMs

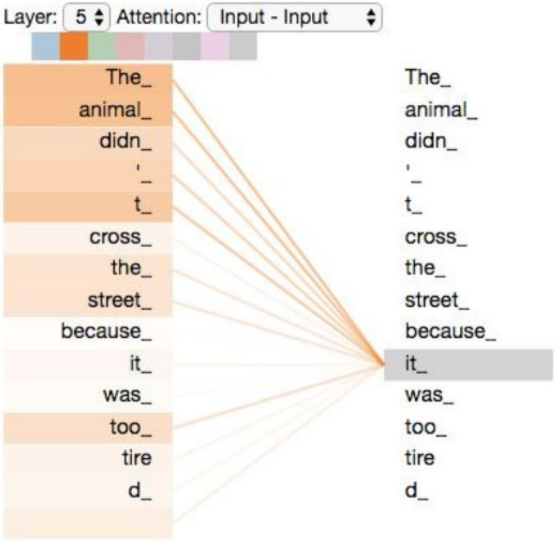
- ~1925 Ising-Models (untrained RNNs)
- ~’72–’81 Hopfield networks (trained RNNs)
- ~’80–’90 Amari, Rumelhart et al., Werbos, etc. - Back-propagation and BPTT are introduced/popularized
- ~’90–’20s Expressivity results (Hava Siegelmann & Sontag in 1991): RNNs are Turing-Complete
- ~’92–’24 Bengio et al., Hochreiter & Schmidhuber: RNNs are hard to train (vanishing/exploding gradients problems)
- ~’01–’10 Echo State Networks / LSM as an answer to the trainability problem
- 1997 Hochreiter & Schmidhuber: LSTMs
- 2014 Graves shows LSTMs to work at “scale”
  - Sutskever et al. Seq2Seq Model
  - Chung et al. 2014, GRU




- BP was adopted (as BPTT) to train them
- Early expressivity results made RNN very desirable architecture
- Allowed to condition on an arbitrary length sequence
- Exhibits optimization issues (vanishing/exploding gradient) and scalability issues (required sequential computation)

# History(?): Transformers




$$\mathrm{MultiHead}(Q, K, V) = \mathrm{Concat}(\mathrm{head_1}, ..., \mathrm{head_h})W^O$$

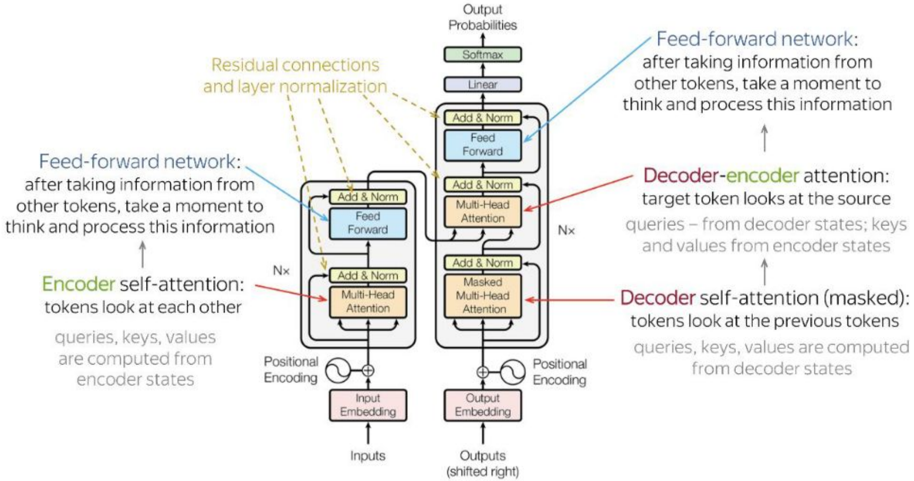
$$\text{where } \mathrm{head_i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

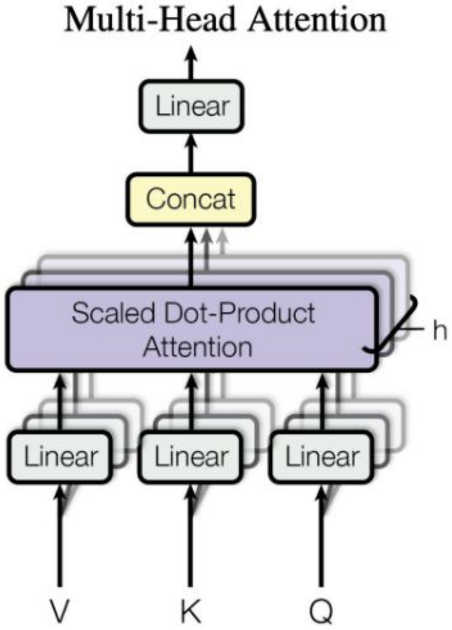
$h = 8$ parallel attention layers, or heads.

Learnable parameter matrices

$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

# History(?): Transformers



# SSM Fundamentals and S4

- Before we move forward, **two** branches with “SSMs”

Much More **DL-Like**.

**LRU** → **Griffin / Hawk**

Much More **Mathematical**.

**S4** → **S5 / Mamba**

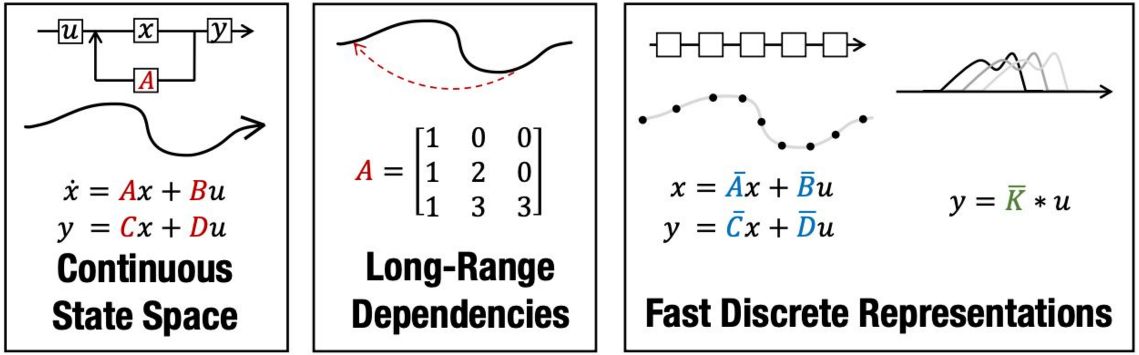
This presentation ✅

# SSM Fundamentals and S4

- 2020 HiPPO is introduced
- 2021 Linear State Space Layer (LSSL)
- 2022 Structured State Space Model (S4)
- 2022 SaShiMi
- 2022 Hyena
- 2022 Diagonal Structured Space Models (S4D)
- 2022 Liquid State Space Models
- 2022 Simplified State Space Models (S5)
- 2022 SGConv
- 2022 Hungry Hungry Hippos (H3)
- 2023 Mega
- 2023 Linear Recurrence Units (LRU)
- 2023 RWKV
- 2023 RetNet
- 2023 2-D SSMs
- 2023 Mamba
- 2023 Vision-Mamba

Basically fixed 2 **important** problems with RNNs,

- **Stable** training,
- **Scalable** training.



# SSM Fundamentals and S4

- Key ideas:
  - Continuous time interpretation,
  - Specific initialization,
  - Discretization + Diagonalization.

- Continuous time interpretation,
  - **Original** formulation of state-space models.

$$
\begin{aligned}
x'(t) &= \boldsymbol{A}x(t) + \boldsymbol{B}u(t) \\
y(t) &= \boldsymbol{C}x(t) + \boldsymbol{D}u(t)
\end{aligned}
$$

# SSM Fundamentals and S4

- Specific initialization,
  - **More theory**, this is from echo networks.

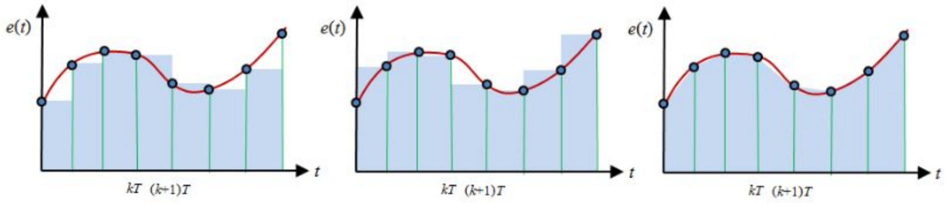
$$\textbf{(HiPPO Matrix)} \qquad \boldsymbol{A}_{nk} = -\begin{cases} (2n+1)^{1/2}(2k+1)^{1/2} & \text{if } n > k \\ n+1 & \text{if } n = k \\ 0 & \text{if } n < k \end{cases}$$

  - This initialization helped get from **60% to 98%** on MNIST!

# SSM Fundamentals and S4

- **Discretization** and Diagonalization.
  Clearly, our input is **not** continuous time (speech, image-pixels, etc.), so we need to discretize the system, this lends us

$$
\begin{aligned}
x_k &= \overline{\boldsymbol{A}} x_{k-1} + \overline{\boldsymbol{B}} u_k \\
y_k &= \overline{\boldsymbol{C}} x_k
\end{aligned}
\qquad
\begin{aligned}
\overline{\boldsymbol{A}} &= (\boldsymbol{I} - \Delta/2 \cdot \boldsymbol{A})^{-1}(\boldsymbol{I} + \Delta/2 \cdot \boldsymbol{A}) \\
\overline{\boldsymbol{B}} &= (\boldsymbol{I} - \Delta/2 \cdot \boldsymbol{A})^{-1} \Delta \boldsymbol{B} \qquad \overline{\boldsymbol{C}} = \boldsymbol{C}.
\end{aligned}
$$




Forward Difference Backward Difference Bilinear Transform

# <u>SSM Fundamentals</u> and S4

- Discretization and **Diagonalization**.
  Why diagonalize in the first place? <u>**Very important question**</u>!
  We will answer this but first let's see the how-to (briefly).

$$x_k = \bar{A}x_{k-1} + \bar{B}u_k$$

$$\bar{A} = PDP^{-1}$$

$$x_k = PDP^{-1}x_{k-1} + \bar{B}u_k$$

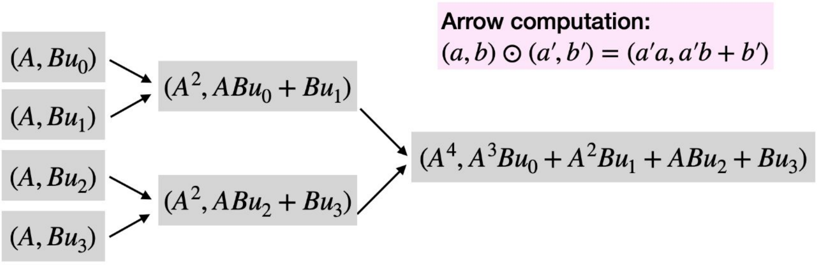
$$P^{-1}x_k = DP^{-1}x_{k-1} + P^{-1}\bar{B}u_k$$

$$\tilde{x}_k = D\tilde{x}_{k-1} + \tilde{B}u_k$$

- For arbitrary Ā, D will be **complex**.

# SSM Fundamentals and S4

- Now, how is this **stable**?
  - Because Ā is **diagonal**, we can directly access it's spectrum, and thus have **control** on recurrence blow-up. (Just parametrize such that λ <= 1, used commonly - $\bar{A} = -e^{\phi}$ where $\Phi$ is **learnable**)
  - Also, recurrence is **linear**.
- How / why is it **scalable**?
  - **Associative Scans**[2] (or interpret as convolution).



# SSM Fundamentals and S4

- How / why is it **scalable**?
  - Associative Scans[2] (**or interpret as convolution**).

$$
\begin{aligned}
x_0 &= \overline{B}u_0 & x_1 &= \overline{AB}u_0 + \overline{B}u_1 & x_2 &= \overline{A}^2\overline{B}u_0 + \overline{AB}u_1 + \overline{B}u_2 & &\dots \\
y_0 &= \overline{CB}u_0 & y_1 &= \overline{CAB}u_0 + \overline{CB}u_1 & y_2 &= \overline{CA}^2\overline{B}u_0 + \overline{CAB}u_1 + \overline{CB}u_2 & &\dots
\end{aligned}
$$

$$
\begin{aligned}
y_k &= \overline{CA}^k\overline{B}u_0 + \overline{CA}^{k-1}\overline{B}u_1 + \cdots + \overline{CAB}u_{k-1} + \overline{CB}u_k \\
y &= \overline{K} * u.
\end{aligned}
$$

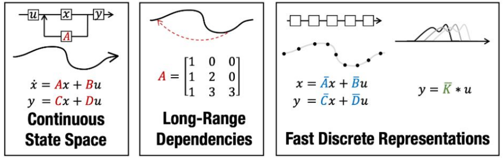
$$
\overline{K} \in \mathbb{R}^L := \mathcal{K}_L(\overline{A}, \overline{B}, \overline{C}) := \left(\overline{CA}^i\overline{B}\right)_{i\in[L]} = (\overline{CB}, \overline{CAB}, \dots, \overline{CA}^{L-1}\overline{B}).
$$

  - +1 to diagonalization, reduces **FLOPs**, $O(H^2)$ to $O(H)$!

# SSM Fundamentals and S4

- Then why LRU?

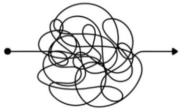
## State Space Models




**But…**

- Many variants of SSMs have been proposed (S4, S4D, H3, Mamba ..)
- Multiple choices for discretization (e.g. bilinear, ZOH)
- Unclear how changes to the architecture interacts with parametrization of SSM
- Is the continuous time interpretation necessary

How can we disentangle what is important?



# Efficiently Modeling Long Sequences with Structured State Spaces

Albert Gu, Karan Goel, and Christopher Ré

Department of Computer Science, Stanford University

{albertgu,krng}@stanford.edu, chrismre@cs.stanford.edu

[2111.00396]
[The Annotated S4]
[GitHub]

# SSM Fundamentals and S4

- Now the S4 paper is just connecting all the dots and making it a learnable system.
    - Continuous time interpretation ✅,
    - HiPPO initialization ❌ (almost),
    - Discretization ✅,
    - Training = Convolutional interpretation ✅,
    - Diagonalization (Motivated through computational efficiency) ✅,
    - + Extra stuff for an actual **fast** implementation. (Really fast!)

# SSM Fundamentals and S4 [the complicated stuff]

- Actual computation and NPLR / DPLR matrices.
  - First, why Normal? Because we perform **conjugation**.
  - Second, why Low Rank? To **approximate** HiPPO matrices.
- However, this would **not** be enough, powering up a sum is still as **problematic** as powering up any other matrix => **slow** implementation.
- Naïvely, this needs $O(N^2L)$ computations to just compute the kernel, however they describe an algorithm which does the following!

**Theorem 3** (S4 Convolution). *Given any step size $\Delta$, computing the SSM convolution filter $\overline{\boldsymbol{K}}$ can be reduced to 4 Cauchy multiplies, requiring only $\widetilde{O}(N+L)$ operations and $O(N+L)$ space.*

# SSM Fundamentals and S4 [the **really** complicated stuff]

**Algorithm 1** S4 CONVOLUTION KERNEL (SKETCH)

**Input:** S4 parameters $\boldsymbol{\Lambda}, \boldsymbol{P}, \boldsymbol{Q}, \boldsymbol{B}, \boldsymbol{C} \in \mathbb{C}^N$ and step size $\Delta$

**Output:** SSM convolution kernel $\overline{\boldsymbol{K}} = \mathcal{K}_L(\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}}, \overline{\boldsymbol{C}})$ for $\boldsymbol{A} = \boldsymbol{\Lambda} - \boldsymbol{P}\boldsymbol{Q}^*$ (equation (5))

1: $\widetilde{\boldsymbol{C}} \leftarrow \left(\boldsymbol{I} - \overline{\boldsymbol{A}}^L\right)^* \overline{\boldsymbol{C}}$ ▷ Truncate SSM generating function (SSMGF) to length $L$

2: $\begin{bmatrix} k_{00}(\omega) & k_{01}(\omega) \\ k_{10}(\omega) & k_{11}(\omega) \end{bmatrix} \leftarrow \left[\widetilde{\boldsymbol{C}}\ \boldsymbol{Q}\right]^* \left(\frac{2}{\Delta}\frac{1-\omega}{1+\omega} - \boldsymbol{\Lambda}\right)^{-1} [\boldsymbol{B}\ \boldsymbol{P}]$ ▷ Black-box Cauchy kernel

3: $\hat{\boldsymbol{K}}(\omega) \leftarrow \frac{2}{1+\omega}\left[k_{00}(\omega) - k_{01}(\omega)(1 + k_{11}(\omega))^{-1}k_{10}(\omega)\right]$ ▷ Woodbury Identity

4: $\hat{\boldsymbol{K}} = \{\hat{\boldsymbol{K}}(\omega) : \omega = \exp(2\pi i \frac{k}{L})\}$ ▷ Evaluate SSMGF at all roots of unity $\omega \in \Omega_L$

5: $\overline{\boldsymbol{K}} \leftarrow \mathsf{iFFT}(\hat{\boldsymbol{K}})$ ▷ Inverse Fourier Transform

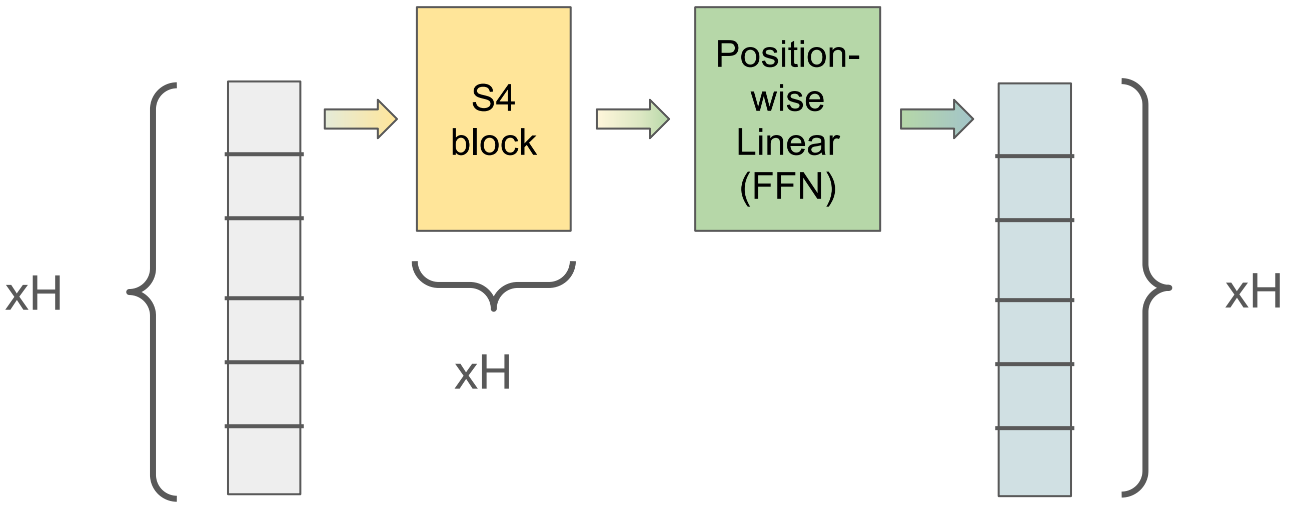
# SSM Fundamentals and S4

- Key takeaways -

| | Convolution[3] | Recurrence | Attention | S4 |
|---|---|---|---|---|
| Parameters | $LH$ | $\boldsymbol{H^2}$ | $\boldsymbol{H^2}$ | $\boldsymbol{H^2}$ |
| Training | $\boldsymbol{\tilde{L}H(B+H)}$ | $BLH^2$ | $B(L^2H+LH^2)$ | $\boldsymbol{BH(\tilde{H}+\tilde{L})+B\tilde{L}H}$ |
| Space | $\boldsymbol{BLH}$ | $\boldsymbol{BLH}$ | $B(L^2+HL)$ | $\boldsymbol{BLH}$ |
| Parallel | **Yes** | No | **Yes** | **Yes** |
| Inference | $LH^2$ | $\boldsymbol{H^2}$ | $L^2H+H^2L$ | $\boldsymbol{H^2}$ |

# SSM Fundamentals and S4

- Architecture (one layer) -



# SSM Fundamentals and S4

## 😮‍💨 Results,

Table 4: (**Long Range Arena**) (*Top*) Original Transformer variants in LRA. Full results in Appendix D.2. (*Bottom*) Other models reported in the literature. *Please read Appendix D.5 before citing this table.*

| MODEL | LISTOPS | TEXT | RETRIEVAL | IMAGE | PATHFINDER | PATH-X | AVG |
|---|---|---|---|---|---|---|---|
| Transformer | 36.37 | 64.27 | 57.46 | 42.44 | 71.40 | ✗ | 53.66 |
| Reformer | <u>37.27</u> | 56.10 | 53.40 | 38.07 | 68.50 | ✗ | 50.56 |
| BigBird | 36.05 | 64.02 | 59.29 | 40.83 | 74.87 | ✗ | 54.17 |
| Linear Trans. | 16.13 | <u>65.90</u> | 53.09 | 42.34 | 75.30 | ✗ | 50.46 |
| Performer | 18.01 | 65.40 | 53.82 | 42.77 | 77.05 | ✗ | 51.18 |
| FNet | 35.33 | 65.11 | 59.61 | 38.67 | <u>77.80</u> | ✗ | 54.42 |
| Nyströmformer | 37.15 | 65.52 | <u>79.56</u> | 41.58 | 70.94 | ✗ | 57.46 |
| Luna-256 | 37.25 | 64.57 | 79.29 | <u>47.38</u> | 77.72 | ✗ | <u>59.37</u> |
| **S4** | **59.60** | **86.82** | **90.90** | **88.65** | **94.20** | **96.35** | **86.09** |

# SSM Fundamentals and S4

😮‍💨 Results,

Table 2: Deep SSMs: The S4 parameterization with Algorithm 1 is asymptotically more efficient than the LSSL.

| | TRAINING STEP (MS) | | | MEMORY ALLOC. (MB) | | |
|---|---|---|---|---|---|---|
| Dim. | 128 | 256 | 512 | 128 | 256 | 512 |
| LSSL | 9.32 | 20.6 | 140.7 | 222.1 | 1685 | 13140 |
| **S4** | 4.77 | 3.07 | 4.75 | 5.3 | 12.6 | 33.5 |
| Ratio | 1.9× | 6.7× | **29.6**× | 42.0× | 133× | **392**× |

Table 3: Benchmarks vs. efficient Transformers

| | LENGTH 1024 | | LENGTH 4096 | |
|---|---|---|---|---|
| | Speed | Mem. | Speed | Mem. |
| Transformer | 1× | 1× | 1× | 1× |
| Performer | 1.23× | 0.43× | 3.79× | 0.086× |
| Linear Trans. | **1.58**× | **0.37**× | **5.35**× | **0.067**× |
| **S4** | **1.58**× | 0.43× | 5.19× | 0.091× |

# SIMPLIFIED STATE SPACE LAYERS FOR SEQUENCE MODELING

**!11th Oct!**

**Jimmy T.H. Smith**[*, 1, 2], **Andrew Warrington**[*, 2, 3], **Scott W. Linderman**[2, 3]
[*]Equal contribution.
[1]Institute for Computational and Mathematical Engineering, Stanford University.
[2]Wu Tsai Neurosciences Institute, Stanford University.
[3]Department of Statistics, Stanford University.
`{jsmith14,awarring,scott.linderman}@stanford.edu.`

[2208.04933]
[GitHub]

# Mamba: Linear-Time Sequence Modeling with Selective State Spaces

Albert Gu*[1] and Tri Dao*[2]

[1]Machine Learning Department, Carnegie Mellon University

[2]Department of Computer Science, Princeton University

agu@cs.cmu.edu, tri@tridao.me

[2312.00752]

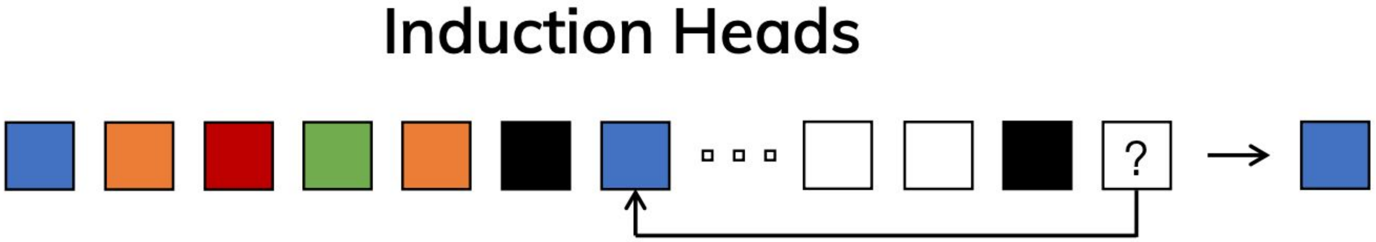
[GitHub]

# Mamba: Linear Time Sequence Modeling with Selective State Spaces

- S6 borrows a lot of parts from S4 + an important **linear-time-variant** extension.
  - Drops Complex analysis ❌
  - Drops HiPPO completely ❌
  - Linear Time Invariance ❌
- The paper has 3 major contributions:
  - The LTI-drop (**selection** mechanism),
  - Hardware-aware algorithm,
  - Scaling.

# Mamba: Linear Time Sequence Modeling with Selective State Spaces

- **Selection**:
  - *Why?* Very closely related LSTM-gating, select data in an **input-dependent** manner.
  - Selectivity as the **goal** of ~~language~~ **sequence** modelling, effectiveness - efficiency tradeoff.



# Mamba: Linear Time Sequence Modeling with Selective State Spaces

- **Selection**:
  - They make a very strong claim. It means

$$
\begin{aligned}
x_k &= \overline{\boldsymbol{A}} x_{k-1} + \overline{\boldsymbol{B}} u_k \\
y_k &= \overline{\boldsymbol{C}} x_k
\end{aligned}
\quad\Bigg|\quad
\begin{aligned}
\overline{\boldsymbol{A}} &= (\boldsymbol{I} - \Delta/2 \cdot \boldsymbol{A})^{-1}(\boldsymbol{I} + \Delta/2 \cdot \boldsymbol{A}) \\
\overline{\boldsymbol{B}} &= (\boldsymbol{I} - \Delta/2 \cdot \boldsymbol{A})^{-1} \Delta \boldsymbol{B} \qquad\qquad \overline{\boldsymbol{C}} = \boldsymbol{C}.
\end{aligned}
$$

    **<u>cannot learn</u>** the induction heads task for **<u>any</u>** A, B, C, Δ. Though they have not proved this.
  - Their formulation was inspired from hypernetworks, gating, data-dependent transforms research BUT is not an GLU activation!

# Mamba: Linear Time Sequence Modeling with Selective State Spaces

- **Selection**:

Table 11: (**Induction heads**.) Models are trained on sequence length $2^8 = 256$, and tested on various sequence lengths of $2^6 = 64$ up to $2^{20} = 1048576$. ✓ denotes perfect generalization accuracy, while ✗ denotes out of memory.

| MODEL | PARAMS | TEST ACCURACY (%) AT SEQUENCE LENGTH | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $2^6$ | $2^7$ | $\mathbf{2^8}$ | $2^9$ | $2^{10}$ | $2^{11}$ | $2^{12}$ | $2^{13}$ | $2^{14}$ | $2^{15}$ | $2^{16}$ | $2^{17}$ | $2^{18}$ | $2^{19}$ | $2^{20}$ |
| MHA-Abs | 137K | ✓ | 99.6 | 100.0 | 58.6 | 26.6 | 18.8 | 9.8 | 10.9 | 7.8 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| MHA-RoPE | 137K | ✓ | ✓ | 100.0 | 83.6 | 31.3 | 18.4 | 8.6 | 9.0 | 5.5 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| MHA-xPos | 137K | ✓ | ✓ | 100.0 | 99.6 | 67.6 | 25.4 | 7.0 | 9.0 | 7.8 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| H3 | 153K | ✓ | ✓ | 100.0 | 80.9 | 39.5 | 23.8 | 14.8 | 8.2 | 5.9 | 6.6 | 8.2 | 4.7 | 8.2 | 6.3 | 7.4 |
| Hyena | 69M* | 97.7 | ✓ | 100.0 | ✓ | 44.1 | 12.5 | 6.6 | 5.1 | 7.0 | 5.9 | 6.6 | 6.6 | 5.9 | 6.3 | 9.8 |
| Mamba | 74K | ✓ | ✓ | 100.0 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

* Most of the parameters are in learnable positional encodings.

# Mamba: Linear Time Sequence Modeling with Selective State Spaces

**Algorithm 1** SSM (S4)

**Input:** $x : (\mathsf{B}, \mathsf{L}, \mathsf{D})$

**Output:** $y : (\mathsf{B}, \mathsf{L}, \mathsf{D})$

$\boldsymbol{A} : (\mathsf{D}, \mathsf{N}) \leftarrow \mathsf{Parameter}$
▷ Represents structured $N \times N$ matrix

$\boldsymbol{B} : (\mathsf{D}, \mathsf{N}) \leftarrow \mathsf{Parameter}$

$\boldsymbol{C} : (\mathsf{D}, \mathsf{N}) \leftarrow \mathsf{Parameter}$

$\Delta : (\mathsf{D}) \leftarrow \tau_\Delta(\mathsf{Parameter})$

$\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}} : (\mathsf{D}, \mathsf{N}) \leftarrow \mathsf{discretize}(\Delta, \boldsymbol{A}, \boldsymbol{B})$

$y \leftarrow \mathsf{SSM}(\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}}, \boldsymbol{C})(x)$
▷ Time-invariant: recurrence or convolution

**return** $y$

**Algorithm 2** SSM + Selection (S6)

**Input:** $x : (\mathsf{B}, \mathsf{L}, \mathsf{D})$

**Output:** $y : (\mathsf{B}, \mathsf{L}, \mathsf{D})$

$\boldsymbol{A} : (\mathsf{D}, \mathsf{N}) \leftarrow \mathsf{Parameter}$
▷ Represents structured $N \times N$ matrix

$\boldsymbol{B} : (\mathsf{B}, \mathsf{L}, \mathsf{N}) \leftarrow s_B(x)$

$\boldsymbol{C} : (\mathsf{B}, \mathsf{L}, \mathsf{N}) \leftarrow s_C(x)$

$\Delta : (\mathsf{B}, \mathsf{L}, \mathsf{D}) \leftarrow \tau_\Delta(\mathsf{Parameter} + s_\Delta(x))$

$\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}} : (\mathsf{B}, \mathsf{L}, \mathsf{D}, \mathsf{N}) \leftarrow \mathsf{discretize}(\Delta, \boldsymbol{A}, \boldsymbol{B})$

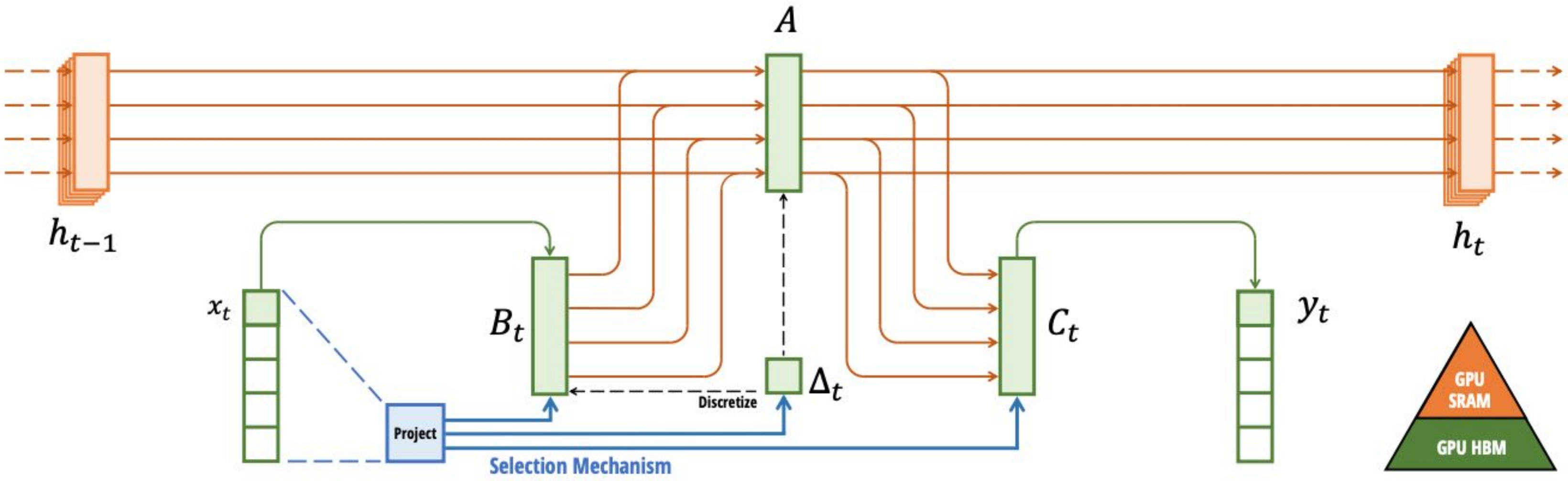
$y \leftarrow \mathsf{SSM}(\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}}, \boldsymbol{C})(x)$
▷ Time-varying: recurrence (*scan*) only

**return** $y$

# Mamba: Linear Time Sequence Modeling with Selective State Spaces [the somewhat complicated part]

- The algorithm is theoretically faster than S4, for small state dimensions, but has a major problem.
    - Why faster than S4? Convolution is ***O*(B*L*D*log(L))** (L*log(L) because FFT), and Mamba conv is ***O*(B*L*D*N)**.
    - However, a naïve implementation would materialize the hidden state of dimensions **B*L*D*N** in GPU HBM (High Bandwidth Memory).
    - Basically, CUDA kernels from Nvidia are not sufficient, so they write their own CUDA kernels to have **kernel fusion**.
    - This seems like fancy terms but are really easy concepts in reality. (easy to think of, not easy to implement!)

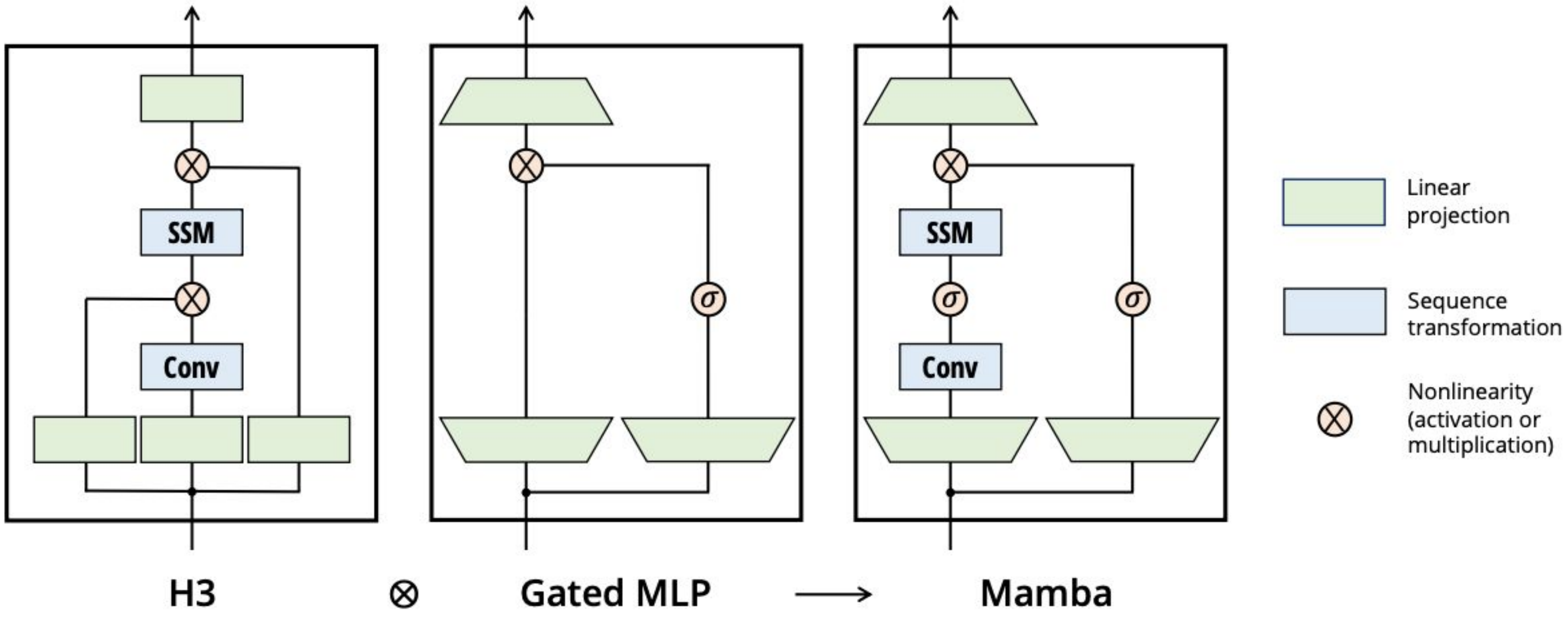
# Mamba: Linear Time Sequence Modeling with Selective State Spaces



# Mamba: Linear Time Sequence Modeling with Selective State Spaces

- The paper has 3 major contributions:
  - The LTI-drop (**selection** mechanism), ✅
  - Hardware-aware algorithm, ✅
  - Scaling ❓
- Finally, this is the first paper on **this** branch of SSMs that scales their model to a few billion parameters to test on Language Modelling, and other real world tasks.
- Interestingly, they got **<u>rejected</u>** from ICLR’24 because at the time of submission they did not include LRA tasks.

# Mamba: Linear Time Sequence Modeling with Selective State Spaces



# Mamba: Linear Time Sequence Modeling with Selective State Spaces

- An **important** theorem.

**Theorem 1.** *When* $N = 1$, $\boldsymbol{A} = -1$, $\boldsymbol{B} = 1$, $s_\Delta = \mathsf{Linear}(x)$, *and* $\tau_\Delta = \mathsf{softplus}$, *then the selective SSM recurrence (Algorithm 2) takes the form*

$$\begin{aligned} g_t &= \sigma(\mathsf{Linear}(x_t)) \\ h_t &= (1 - g_t)h_{t-1} + g_t x_t. \end{aligned} \tag{5}$$

# Mamba: Linear Time Sequence Modeling with Selective State Spaces

- Some more discussion on the arch.
  - Variable Spacing / Filtering Context,
  - Boundary Resetting,
  - Interpretation of A, B, C and Δ.
- Major takeaway, Δ dictates a lot!

$\Delta \to \infty$ Resets **context**, massive focus on input.

$\Delta \to 0$ Ignores **input**, massive focus on context.

# Mamba: Linear Time Sequence Modeling with Selective State Spaces

## 😮‍💨 Results (synthetic),

| Model | Arch. | Layer | Acc. |
|---|---|---|---|
| S4 | No gate | S4 | 18.3 |
| - | No gate | S6 | **97.0** |
| H3 | H3 | S4 | 57.0 |
| Hyena | H3 | Hyena | 30.1 |
| - | H3 | S6 | **99.7** |
| - | Mamba | S4 | 56.4 |
| - | Mamba | Hyena | 28.4 |
| Mamba | Mamba | S6 | **99.8** |

Table 1: (**Selective Copying**.) Accuracy for combinations of architectures and inner sequence layers.

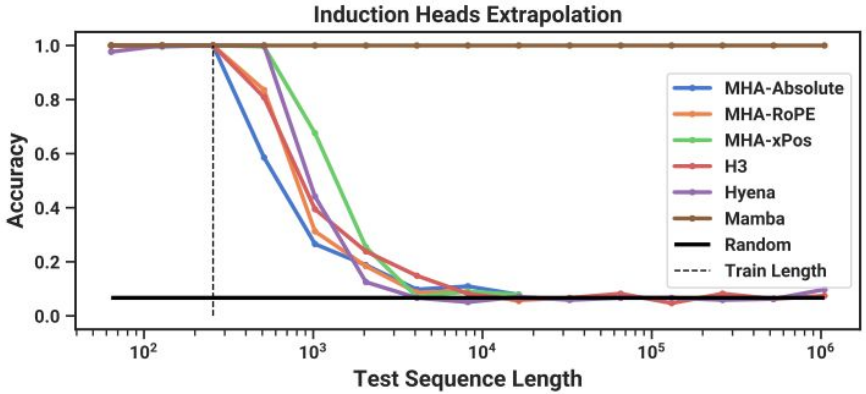
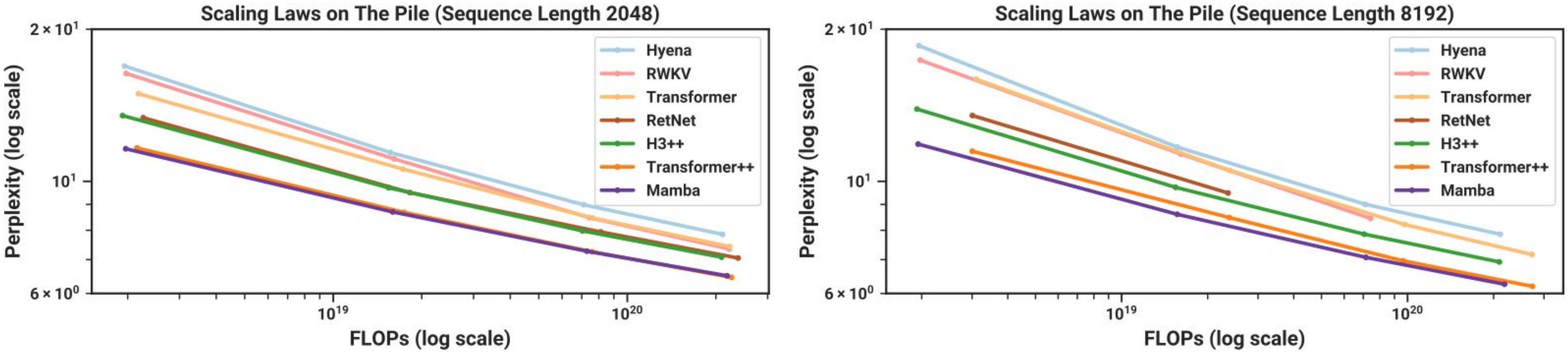



Table 2: (**Induction Heads**.) Models are trained on sequence length $2^8 = 256$, and tested on increasing sequence lengths of $2^6 = 64$ up to $2^{20} = 1048576$. Full numbers in Table 11.

# Mamba: Linear Time Sequence Modeling with Selective State Spaces

😮‍💨 Results (scaling laws),



# Mamba: Linear Time Sequence Modeling with Selective State Spaces

😮‍💨 Results (language modelling),

| Model | Token. | Pile ppl ↓ | LAMBADA ppl ↓ | LAMBADA acc ↑ | HellaSwag acc ↑ | PIQA acc ↑ | Arc-E acc ↑ | Arc-C acc ↑ | WinoGrande acc ↑ | Average acc ↑ |
|---|---|---|---|---|---|---|---|---|---|---|
| Hybrid H3-130M | GPT2 | — | 89.48 | 25.77 | 31.7 | 64.2 | 44.4 | 24.2 | 50.6 | 40.1 |
| Pythia-160M | NeoX | 29.64 | 38.10 | 33.0 | 30.2 | 61.4 | 43.2 | 24.1 | **51.9** | 40.6 |
| **Mamba-130M** | NeoX | **10.56** | **16.07** | **44.3** | **35.3** | **64.5** | **48.0** | **24.3** | **51.9** | **44.7** |
| Hybrid H3-360M | GPT2 | — | 12.58 | 48.0 | 41.5 | 68.1 | 51.4 | 24.7 | 54.1 | 48.0 |
| Pythia-410M | NeoX | 9.95 | 10.84 | 51.4 | 40.6 | 66.9 | 52.1 | 24.6 | 53.8 | 48.2 |
| **Mamba-370M** | NeoX | **8.28** | **8.14** | **55.6** | **46.5** | **69.5** | **55.1** | **28.0** | **55.3** | **50.0** |
| Pythia-1B | NeoX | 7.82 | 7.92 | 56.1 | 47.2 | 70.7 | 57.0 | 27.1 | 53.5 | 51.9 |
| **Mamba-790M** | NeoX | **7.33** | **6.02** | **62.7** | **55.1** | **72.1** | **61.2** | **29.5** | **56.1** | **57.1** |
| GPT-Neo 1.3B | GPT2 | — | 7.50 | 57.2 | 48.9 | 71.1 | 56.2 | 25.9 | 54.9 | 52.4 |
| Hybrid H3-1.3B | GPT2 | — | 11.25 | 49.6 | 52.6 | 71.3 | 59.2 | 28.1 | 56.9 | 53.0 |
| OPT-1.3B | OPT | — | 6.64 | 58.0 | 53.7 | 72.4 | 56.7 | 29.6 | 59.5 | 55.0 |
| Pythia-1.4B | NeoX | 7.51 | 6.08 | 61.7 | 52.1 | 71.0 | 60.5 | 28.5 | 57.2 | 55.2 |
| RWKV-1.5B | NeoX | 7.70 | 7.04 | 56.4 | 52.5 | 72.4 | 60.5 | 29.4 | 54.6 | 54.3 |
| **Mamba-1.4B** | NeoX | **6.80** | **5.04** | **64.9** | **59.1** | **74.2** | **65.5** | **32.8** | **61.5** | **59.7** |
| GPT-Neo 2.7B | GPT2 | — | 5.63 | 62.2 | 55.8 | 72.1 | 61.1 | 30.2 | 57.6 | 56.5 |
| Hybrid H3-2.7B | GPT2 | — | 7.92 | 55.7 | 59.7 | 73.3 | 65.6 | 32.3 | 61.4 | 58.0 |
| OPT-2.7B | OPT | — | 5.12 | 63.6 | 60.6 | 74.8 | 60.8 | 31.3 | 61.0 | 58.7 |
| Pythia-2.8B | NeoX | 6.73 | 5.04 | 64.7 | 59.3 | 74.0 | 64.1 | 32.9 | 59.7 | 59.1 |
| RWKV-3B | NeoX | 7.00 | 5.24 | 63.9 | 59.6 | 73.7 | 67.8 | 33.1 | 59.6 | 59.6 |
| **Mamba-2.8B** | NeoX | **6.22** | **4.23** | **69.2** | **66.1** | **75.2** | **69.7** | **36.3** | **63.5** | **63.3** |
| GPT-J-6B | GPT2 | – | 4.10 | 68.3 | 66.3 | 75.4 | 67.0 | 36.6 | 64.1 | 63.0 |
| OPT-6.7B | OPT | – | 4.25 | 67.7 | 67.2 | 76.3 | 65.6 | 34.9 | 65.5 | 62.9 |
| Pythia-6.9B | NeoX | 6.51 | 4.45 | 67.1 | 64.0 | 75.2 | 67.3 | 35.5 | 61.3 | 61.7 |
| RWKV-7.4B | NeoX | 6.31 | 4.38 | 67.2 | 65.5 | 76.1 | 67.8 | 37.5 | 61.0 | 62.5 |

Thank you! Questions?