Language Modelling with Recurrent and State-Space Architectures

Satwik Bhattamishra

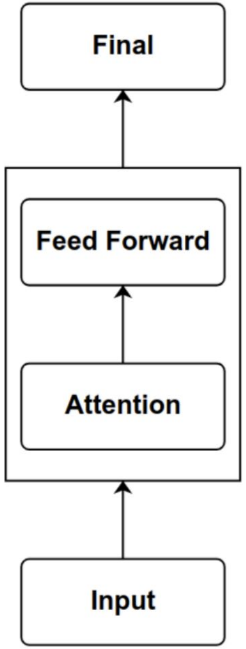
University of Oxford

Agenda

- Motivation
- Linear RNNs
  - Linear RNNs as long convolutions
  - SSMs vs Linear RNNs
- Linear Transformers
  - Recurrent formulation
  - RetNet, Mamba-2
- Strengths and Limitations
- Questions

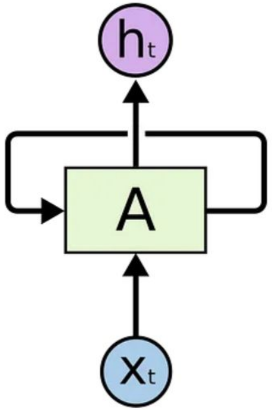
Transformers as LLMs

- Most effective architecture for building LLMs now
- Drawbacks
  - Training cost: Quadratic in length $O(n^2)$
  - Inference: Linear in length $O(n)$



Recurrent Models

- Faster Inference – O(1)
- Drawbacks
  - Traditional RNNs do not scale well
  - Could not be trained in a parallel manner
- Modifications
  - Efficiency: Linear RNNs are parallelizable for Training
  - Tricks to improve long-range dependency modelling
  - Tricks from Transformers – Layernorms, FFNs



Subquadratic Architectures

- Three main classes
  - Linear RNNs/SSMs (S4, DSS, Mamba, etc.)
  - Long convolutional models (Hyena)
  - Linear Transformer variants (Retnet, Mamba-2, Gated Linear Attention)
- They are related
  - Most linear RNNs are long convolutional models as well!
  - Linear Transformers can also be considered as linear RNNs

Linear RNNs

Traditional RNNs

$$h_t = \sigma(Ah_{t-1} + Bx_t)$$
$$y_t = Ch_t$$

Linear RNNs

$$h_t = Ah_{t-1} + Bx_t$$
$$y_t = Ch_t$$

Linear RNNs as convolutions

$$h_t = Ah_{t-1} + Bx_t$$
$$y_t = Ch_t$$

$h_t \in \mathbb{R}^N \quad x_t \in \mathbb{R}$

$A \in \mathbb{R}^{N \times N} \quad B, C^\top \in \mathbb{R}^{N \times 1}$

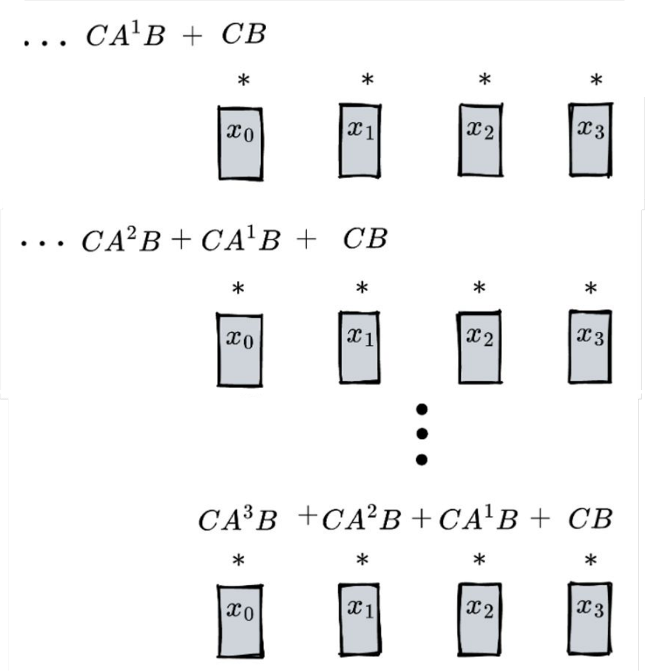
Input Length: 4

$$K = (CB, CAB, CA^2B, CA^3B)$$
$$x = (x_0, x_1, x_2, x_3)$$
$$y = K * x$$ → FFT: $O(n \log n)$

$$h_{-1} = \mathbf{0}$$

$$y_0 = CBx_0$$
$$y_1 = CABx_0 + CBx_1$$
$$y_2 = CA^2Bx_0 + CABx_1 + CBx_2$$
$$y_3 = CA^3Bx_0 + CA^2Bx_1 + CABx_2 + CBx_3$$

Linear RNNs as convolutions



Input Length: 4

$$K = (CB, CAB, CA^2B, CA^3B)$$

$$x = (x_0, x_1, x_2, x_3)$$

$$y = K * x \longrightarrow \text{FFT:} O(n \log n)$$

$$y_0 = CBx_0$$

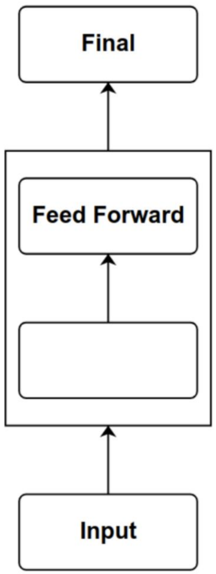
$$y_1 = CABx_0 + CBx_1$$

$$y_2 = CA^2Bx_0 + CABx_1 + CBx_2$$

$$y_3 = CA^3Bx_0 + CA^2Bx_1 + CABx_2 + CBx_3$$

Scaling RNNs

- Transformer recipe works very well for scaling
  - Having layernorm+residual after sequence mixer
  - Having a FFN after sequence mixer
- Training LSTMs with such a recipe works very well for deep networks [*]
- Training pretty much any sequence mixer with this recipe works
  (in terms of scaling)



[*] Resurrecting Recurrent Neural Networks for Long Sequences. 2023

State-space vs RNNs

- Key difference lies in parameterisation of weights

RNN

$$(A, B, C)$$

$$h_t = Ah_{t-1} + Bx_t$$
$$y_t = Ch_t$$

SSM

$$(\Delta, \bar{A}, \bar{B}, C)$$

$$A = f_A(\Delta, \bar{A})$$
$$B = f_B(\Delta, \bar{B})$$

e.g. $A = \exp(\Delta \bar{A})$

SSMs: S4, DSS, etc

Mamba

- Time variant recurrence

$$h_t = Ah_{t-1} + Bx_t$$
$$y_t = Ch_t$$

$$h_t = Ah_{t-1} + B_t x_t$$
$$y_t = C_t h_t$$

$$B_t = s_B(x_t)$$

e.g. $B_t = Wx_t$

Why are they not adopted

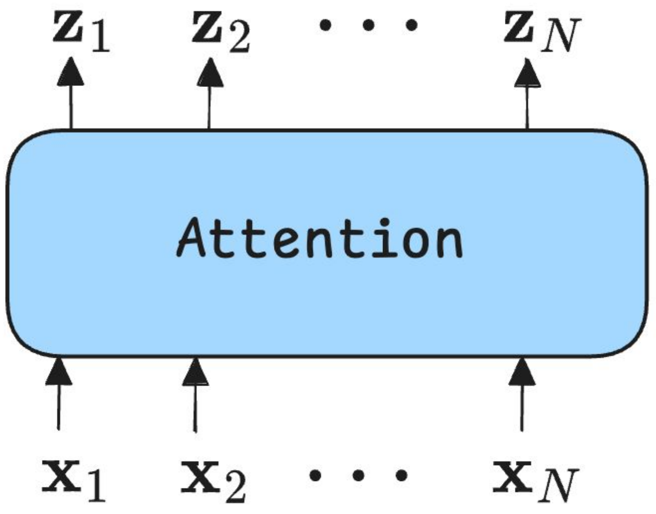
- Performance does not match Transformers*
  - Perplexity is not as good as Transformers
  - Performance is worse on downstream tasks
- Not as efficient on current hardware as they are on paper
  - FFT is quite slow on TPUs
  - Current hardware is well-suited for Transformers

Linear Transformers $\rightarrow$ RetNet $\rightarrow$ Mamba-2

Attention - Transformer

$$\mathbf{x}_i \mapsto \mathbf{q}_i, \mathbf{k}_i, \mathbf{v}_i \in \mathbb{R}^d$$

$$\mathbf{A}_{ij} = \langle \mathbf{q}_i, \mathbf{k}_j \rangle$$



Attention - Transformer

$$\mathbf{x}_i \mapsto \mathbf{q}_i, \mathbf{k}_i, \mathbf{v}_i \in \mathbb{R}^d$$

$$\mathbf{A}_{ij} = \langle \mathbf{q}_i, \mathbf{k}_j \rangle$$

$$\mathbf{Z} = \mathrm{softmax}(L \circ \mathbf{A})\mathbf{V}^\top$$

$$\begin{bmatrix} 1 & -\infty & -\infty & -\infty & -\infty \\ 1 & 1 & -\infty & -\infty & -\infty \\ 1 & 1 & 1 & -\infty & -\infty \\ 1 & 1 & 1 & 1 & -\infty \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \circ \begin{bmatrix} \mathbf{A}_{11} & \cdot & \cdot & \cdot & \mathbf{A}_{15} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \mathbf{A}_{ij} & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \mathbf{A}_{51} & \cdot & \cdot & \cdot & \mathbf{A}_{55} \end{bmatrix}$$

Linear Attention

$$\mathbf{x}_i \mapsto \mathbf{q}_i, \mathbf{k}_i, \mathbf{v}_i \in \mathbb{R}^d$$

$$\mathbf{A}_{ij} = \langle \mathbf{q}_i, \mathbf{k}_j \rangle$$

$$\mathbf{Z} = (L \circ \mathbf{A})\mathbf{V}^\top$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \circ \begin{bmatrix} \mathbf{A}_{11} & \cdot & \cdot & \cdot & \mathbf{A}_{15} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \mathbf{A}_{ij} & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \mathbf{A}_{51} & \cdot & \cdot & \cdot & \mathbf{A}_{55} \end{bmatrix}$$

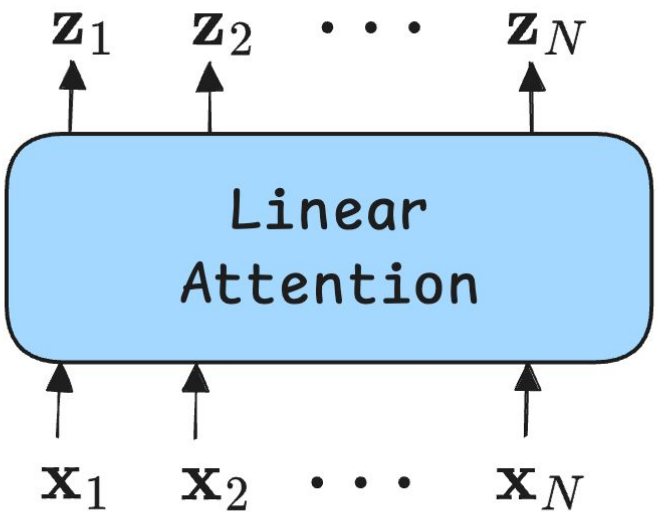
Linear Attention

$$\mathbf{z}_i = (\mathbf{q}_i^\top \mathbf{K}_{:i})\mathbf{V}_{:i}^\top$$

$$\mathbf{z}_1 = \mathbf{q}_1^\top \mathbf{k}_1 \mathbf{v}_1^\top$$

$$\mathbf{z}_2 = \mathbf{q}_2^\top \mathbf{k}_1 \mathbf{v}_1^\top + \mathbf{q}_2^\top \mathbf{k}_2 \mathbf{v}_2^\top$$

$$\mathbf{z}_3 = \mathbf{q}_3^\top \mathbf{k}_1 \mathbf{v}_1^\top + \mathbf{q}_3^\top \mathbf{k}_2 \mathbf{v}_2^\top + \mathbf{q}_3^\top \mathbf{k}_3 \mathbf{v}_3^\top$$



Linear Attention

$$\mathbf{z}_i = (\mathbf{q}_i^\top \mathbf{K}_{:i})\mathbf{V}_{:i}^\top$$

$$\mathbf{z}_1 = \mathbf{q}_1^\top \mathbf{k}_1 \mathbf{v}_1^\top$$

$$\mathbf{z}_2 = \mathbf{q}_2^\top \mathbf{k}_1 \mathbf{v}_1^\top + \mathbf{q}_2^\top \mathbf{k}_2 \mathbf{v}_2^\top$$

$$\mathbf{z}_3 = \mathbf{q}_3^\top \mathbf{k}_1 \mathbf{v}_1^\top + \mathbf{q}_3^\top \mathbf{k}_2 \mathbf{v}_2^\top + \mathbf{q}_3^\top \mathbf{k}_3 \mathbf{v}_3^\top$$

$$\mathbf{z}_1 = \mathbf{q}_1^\top (\mathbf{k}_1 \mathbf{v}_1^\top)$$

$$\mathbf{z}_2 = \mathbf{q}_2^\top (\mathbf{k}_1 \mathbf{v}_1^\top + \mathbf{k}_2 \mathbf{v}_2^\top)$$

$$\mathbf{z}_3 = \mathbf{q}_3^\top (\mathbf{k}_1 \mathbf{v}_1^\top + \mathbf{k}_2 \mathbf{v}_2^\top + \mathbf{k}_3 \mathbf{v}_3^\top)$$

Linear Attention as Recurrence

$$\mathbf{S}_t \in \mathbb{R}^{d \times d}$$

$$\mathbf{S}_0 = \mathbf{0}$$

$$\mathbf{S}_t = \mathbf{S}_{t-1} + \mathbf{k}_t \mathbf{v}_t^\top$$

← State update rule

$$\mathbf{z}_t = \mathbf{q}_t^\top \mathbf{S}_{t-1}$$

Linear Attention as Recurrence

$$\mathbf{S}_t \in \mathbb{R}^{d\times d}$$

$$\mathbf{S}_0 = \mathbf{0}$$

$$\mathbf{S}_t = \mathbf{S}_{t-1} + \mathbf{k}_t \mathbf{v}_t^\top$$

$$\mathbf{z}_t = \mathbf{q}_t^\top \mathbf{S}_{t-1}$$

$$\mathbf{S}_1 = \mathbf{k}_1 \mathbf{v}_1^\top$$

$$\mathbf{S}_2 = \mathbf{k}_1 \mathbf{v}_1^\top + \mathbf{k}_2 \mathbf{v}_2^\top$$

$$\mathbf{z}_1 = \mathbf{q}_1^\top (\mathbf{k}_1 \mathbf{v}_1^\top)$$

$$\mathbf{z}_2 = \mathbf{q}_2^\top (\mathbf{k}_1 \mathbf{v}_1^\top + \mathbf{k}_2 \mathbf{v}_2^\top)$$

RetNet

$$\mathbf{S}_0 = \mathbf{0}$$

$$\mathbf{S}_t = \gamma \mathbf{S}_{t-1} + \mathbf{k}_t \mathbf{v}_t^\top$$

Exponential Decay factor

$$\mathbf{z}_t = \mathbf{q}_t^\top \mathbf{S}_{t-1}$$

Three Changes

- Exponential Decay factor
- RoPE
- Changes to activation/norm

RetNet

$$\mathbf{x}_i \mapsto \mathbf{q}_i, \mathbf{k}_i, \mathbf{v}_i \in \mathbb{R}^d$$

$$\mathbf{A}_{ij} = \langle \mathbf{q}_i, \mathbf{k}_j \rangle$$

$$\mathbf{Z} = (L \circ \mathbf{A})\mathbf{V}^\top$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ \gamma & 1 & 0 & 0 & 0 \\ \gamma^2 & \gamma & 1 & 0 & 0 \\ \gamma^3 & \gamma^2 & \gamma & 1 & 0 \\ \gamma^4 & \gamma^3 & \gamma^2 & \gamma & 1 \end{bmatrix} \circ \begin{bmatrix} \mathbf{A}_{11} & \cdot & \cdot & \cdot & \mathbf{A}_{15} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \mathbf{A}_{ij} & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \mathbf{A}_{51} & \cdot & \cdot & \cdot & \mathbf{A}_{55} \end{bmatrix}$$

Mamba-2

$$\mathbf{S}_0 = \mathbf{0} \qquad \mathbf{S}_t \in \mathbb{R}^{d\times d}$$

$$\mathbf{S}_t = a_t\mathbf{S}_{t-1} + \mathbf{B}_t\mathbf{x}_t^\top \qquad \mathbf{z}_t = \mathbf{C}_t^\top\mathbf{S}_{t-1}$$

RetNet

$$\mathbf{S}_0 = \mathbf{0} \qquad \mathbf{S}_t \in \mathbb{R}^{d\times d}$$

$$\mathbf{S}_t = \gamma\mathbf{S}_{t-1} + \mathbf{k}_t\mathbf{v}_t^\top \qquad \mathbf{z}_t = \mathbf{q}_t^\top\mathbf{S}_{t-1}$$

Mamba-2

$$\mathbf{S}_0 = \mathbf{0} \qquad \mathbf{S}_t \in \mathbb{R}^{d\times d}$$

$$\mathbf{S}_t = a_t\mathbf{S}_{t-1} + \mathbf{k}_t\mathbf{v}_t^\top \qquad \mathbf{z}_t = \mathbf{q}_t^\top\mathbf{S}_{t-1}$$

Mamba-2

$$\mathbf{x}_i \mapsto \mathbf{q}_i, \mathbf{k}_i, \mathbf{v}_i \in \mathbb{R}^d$$

$$\mathbf{Z} = (L \circ \mathbf{A})\mathbf{V}^\top$$

$$\mathbf{A}_{ij} = \langle \mathbf{q}_i, \mathbf{k}_j \rangle$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ a_1 & 1 & 0 & 0 & 0 \\ a_2 a_1 & a_2 & 1 & 0 & 0 \\ a_3 a_2 a_1 & a_3 a_2 & a_3 & 1 & 0 \\ a_4 \ldots a_1 & \cdot & \cdot & a_4 & 1 \end{bmatrix} \circ \begin{bmatrix} \mathbf{A}_{11} & \cdot & \cdot & \cdot & \mathbf{A}_{15} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \mathbf{A}_{ij} & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \mathbf{A}_{51} & \cdot & \cdot & \cdot & \mathbf{A}_{55} \end{bmatrix}$$

RetNet

$$\mathbf{x}_i \mapsto \mathbf{q}_i, \mathbf{k}_i, \mathbf{v}_i \in \mathbb{R}^d$$

$$\mathbf{A}_{ij} = \langle \mathbf{q}_i, \mathbf{k}_j \rangle$$

$$\mathbf{Z} = (L \circ \mathbf{A})\mathbf{V}^\top$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ \gamma & 1 & 0 & 0 & 0 \\ \gamma^2 & \gamma & 1 & 0 & 0 \\ \gamma^3 & \gamma^2 & \gamma & 1 & 0 \\ \gamma^4 & \gamma^3 & \gamma^2 & \gamma & 1 \end{bmatrix} \circ \begin{bmatrix} \mathbf{A}_{11} & \cdot & \cdot & \cdot & \mathbf{A}_{15} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \mathbf{A}_{ij} & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \mathbf{A}_{51} & \cdot & \cdot & \cdot & \mathbf{A}_{55} \end{bmatrix}$$

Mamba-2

- The mask matrix is not fully materialised
- Output can computed more efficiently based on structure of the mask matrix
- Inference can be done in a recurrent fashion

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ a_1 & 1 & 0 & 0 & 0 \\ a_2 a_1 & a_2 & 1 & 0 & 0 \\ a_3 a_2 a_1 & a_3 a_2 & a_3 & 1 & 0 \\ a_4 \ldots a_1 & \cdot & \cdot & a_4 & 1 \end{bmatrix} \circ \begin{bmatrix} \mathbf{A}_{11} & \cdot & \cdot & \cdot & \mathbf{A}_{15} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \mathbf{A}_{ij} & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \mathbf{A}_{51} & \cdot & \cdot & \cdot & \mathbf{A}_{55} \end{bmatrix}$$

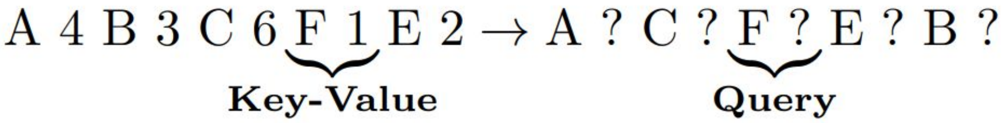
Transformer vs RNNs/SSMs

- Current state
  - There is a performance gap compared to Transformers at the moment [*]
  - Hybrid architectures seem promising [**]
- Both of them process inputs very differently
  - Transformers maintain N vectors and process inputs in parallel
  - RNNs have a memory vector that they can update at every step

[*] Don’t quote me on this!
[*] Zoology: Measuring and Improving Recall in Efficient Language Models. 2023
[**] Samba: Simple Hybrid State Space Models for Efficient Unlimited Context Language Modeling. 2024
[**] Simple linear attention language models balance the recall-throughput tradeoff. 2024
[**] An Empirical Study of Mamba-based Language Models. 2024

Transformer vs RNNs

- RNNs must compress all the required information into a fixed-size vector
    - Makes it difficult for them to perform various associative recall-related tasks
    - Multiple evidence based on theory [1], synthetic tasks [2], and LLM performance [3]



[1] Separations in the Representational Capabilities of Transformers and Recurrent Architectures. 2024
[2] Zoology: Measuring and Improving Recall in Efficient Language Models. 2023
[3] Griffin: Mixing Gated Linear Recurrences with Local Attention for Efficient Language Models. 2024

Transformer vs RNNs

- Transformers perform worse on various algorithmic tasks related to modular counting
  - Multiple evidence based on theory [4, 5] and synthetic tasks [5, 6]
  - Impact on LLM performance is unclear

Parity

x 0 1 1 0 1 0

$$y = \sum_{i=1}^{N} x_i \mod 2$$

[4] Theoretical Limitations of Self-Attention in Neural Sequence Models. 2020
[5] Exposing Attention Glitches with Flip-Flop Language Modeling. 2023
[6] On the Ability and Limitations of Transformers to Recognize Formal Languages. 2020

finis.