

# Inventing Modern Sequence Models as a Music 320 Project

Julius Smith  
CCRMA Open House  
Stanford University

May 17, 2024





[Basic Idea](#)

[History Samples](#)

## Abstract

Today's sequence models (such as large language models) in machine learning (AI) arose from a blend of principle-based design and empirical discovery, spanning several fields.

This talk describes how the ideas could have emerged from an elementary signal-processing approach. This viewpoint offers some features:

1. Signal processing folks can quickly learn what is happening in a motivated way
2. Machine-learning experts might benefit from signal-processing insights
3. Obvious suggestions for things to try next naturally arise

[Open House Schedule]



[Basic Idea](#)

[History Samples](#)

## Music 320 Project Idea

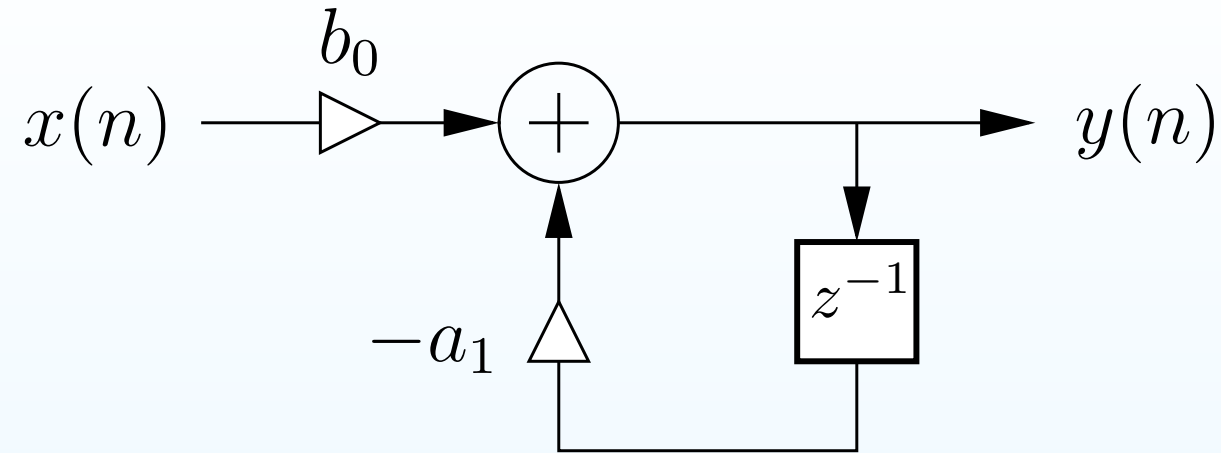


#### Basic Idea

- One Pole Filter
- Inner Product
- Vector Memory
- Gating
- Gated RNN
- Skip Connection
- Sequences
- State Expansion
- Attention
- Dot-Product Attention
- Multi-Head Attention
- Weight Tying
- Hierarchical Blocks
- Hypersphere

#### History Samples

## One Pole Recursive Digital Filter



Pole at  $z = -a_1$

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b_0}{1 + a_1 z^{-1}}$$

### Idea: Let's Make an Associative Memory!

- $x(n)$  can be a *long vector*  $\underline{x}(n) \in \mathbb{R}^N$  representing *anything we want* — any “label”
- Set  $\underline{b}_0 = 1$  and  $\underline{a}_1 = -1$  to make  $\underline{y}(n)$  a *sum of all input vectors* (“integrator”)
- Choose the dimension  $N$  so large that *vectors in the sum are mostly orthogonal*
- Retrieve similar vectors using a *matched inner product*  $\underline{w}^T \underline{x} > b$ ,  
for some suitable threshold  $b$  (Hey! That's one simulated neuron! (“Perceptron”))



#### Basic Idea

- One Pole Filter
- **Inner Product**
- Vector Memory
- Gating
- Gated RNN
- Skip Connection
- Sequences
- State Expansion
- Attention
- Dot-Product Attention
- Multi-Head Attention
- Weight Tying
- Hierarchical Blocks
- Hypersphere

#### History Samples

## Vector Retrieval by Inner Product

Given the sum of vectors

$$\underline{y}(n) = \sum_{m=0}^n \underline{x}(m)$$

and a “query vector”  $\underline{w} = \underline{x}(k)$ ,

find the query in the sum using an *inner product*:

$$\underline{w}^T \underline{y}(n) = \sum_{m=0}^n \underline{w}^T \underline{x}(m) \approx \underline{x}^T(k) \underline{x}(k) = \|\underline{x}(k)\|^2 > b(k)$$

where  $b(k)$  is the *detection threshold* for  $\underline{x}(k)$

- This works because the spatial dimension is so large that  $\underline{x}^T(j) \underline{x}(k) \approx 0$  for  $j \neq k$
- Retrieval threshold  $b(k)$  depends on  $\|\underline{x}(k)\|^2$   
 $\Rightarrow$  *reserve the radial dimension for similarity scoring*
- *I.e., only populate the **hypersphere** in  $\mathbb{R}^N$ :  $\|\underline{x}(k)\| = 1, \forall k$*
- We just invented RMSNorm, used extensively in neural networks (not LayerNorm)



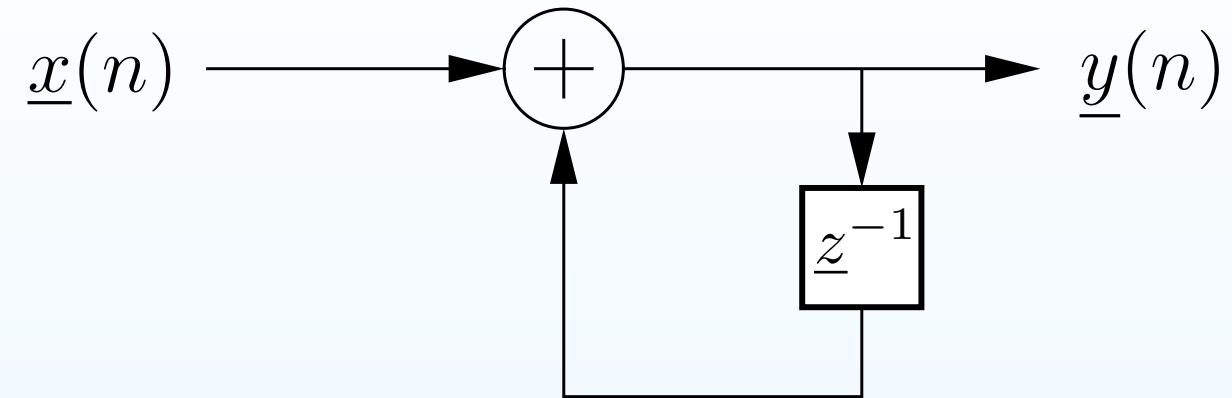


#### Basic Idea

- One Pole Filter
- Inner Product
- **Vector Memory**
- Gating
- Gated RNN
- Skip Connection
- Sequences
- State Expansion
- Attention
- Dot-Product Attention
- Multi-Head Attention
- Weight Tying
- Hierarchical Blocks
- Hypersphere

#### History Samples

## Cumulative Vector Memory



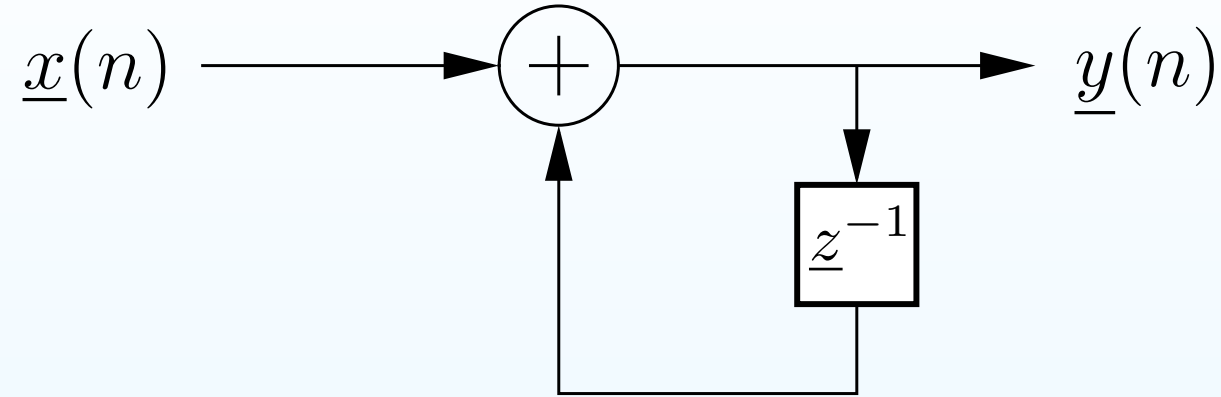


#### Basic Idea

- One Pole Filter
- Inner Product
- Vector Memory
- **Gating**
- Gated RNN
- Skip Connection
- Sequences
- State Expansion
- Attention
- Dot-Product Attention
- Multi-Head Attention
- Weight Tying
- Hierarchical Blocks
- Hypersphere

#### History Samples

## Gated Vector Memory



Input Vector Summer

- **Problem:** Need a *memory reset*
- **Solution:** Set *feedback gain to zero* for one step to clear the memory
- **Problem:** Need an *input gate* to suppress unimportant inputs
- **Solution:** Set *input gain to zero* for unimportant inputs
- We just invented **gating**, used extensively in neural sequence models





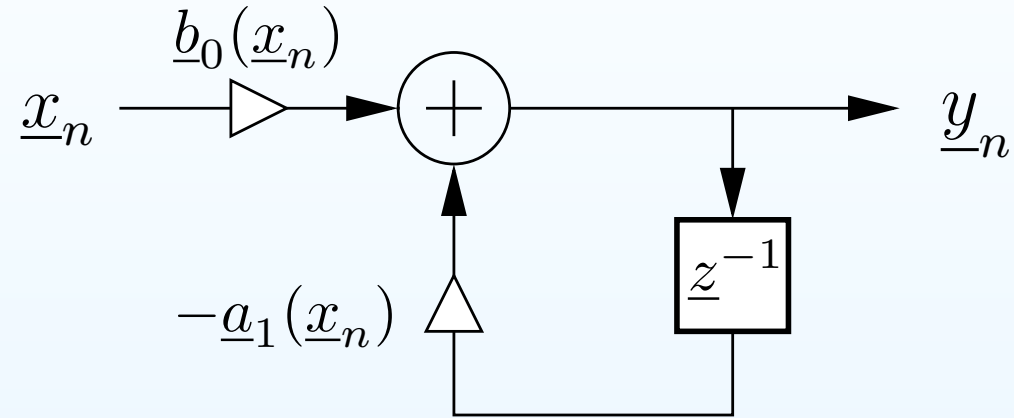
#### Basic Idea

- One Pole Filter
- Inner Product
- Vector Memory
- Gating
- **Gated RNN**
- Skip Connection
- Sequences
- State Expansion
- Attention
- Dot-Product Attention
- Multi-Head Attention
- Weight Tying
- Hierarchical Blocks
- Hypersphere

#### History Samples

## Gated Recurrent Network

**Idea:** *Learn* the input and feedback gates as functions of  $\underline{x}_n$  based on many input-output examples  $(\underline{x}_n, \underline{y}_n)$  (“training data”):



Vector Memory with Learned Input and Feedback Gates

### “Obvious” Training Considerations:

- Initialize  $-\underline{a}_1$  for desired initial memory duration (exponentially fading)
- Learn  $-\underline{a}_1(\underline{x}_n)$  as  $\mathbf{I} \cdot e^{-\Delta} \approx \mathbf{I} - \mathbf{I}\Delta$ , and  $\underline{\beta}_0(\underline{x}_n)$  as  $\text{Linear}(\underline{x}_n, \underline{y}_n) \cdot \Delta$ , where  $\Delta = \text{softPlus}(\text{parameter}(\underline{x}_n, \underline{Y}_n))$  (approximately gain-normalized)





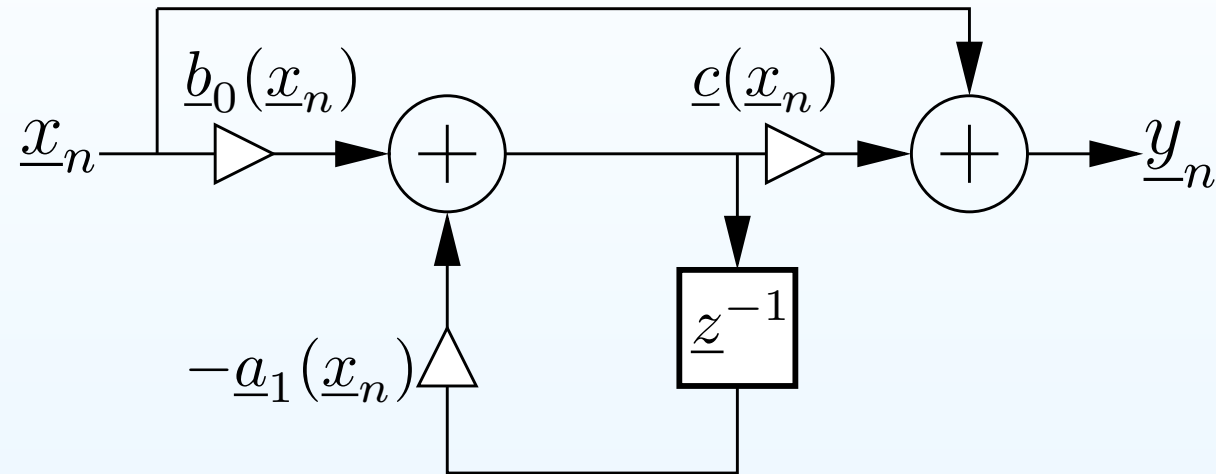
#### Basic Idea

- One Pole Filter
- Inner Product
- Vector Memory
- Gating
- Gated RNN
- **Skip Connection**
- Sequences
- State Expansion
- Attention
- Dot-Product Attention
- Multi-Head Attention
- Weight Tying
- Hierarchical Blocks
- Hypersphere

#### History Samples

## Output Gating

**Idea:** Since we have input and feedback gates, why not an **output gate**?



Gated RNN with **Skip Connection**

Output gating allows network to be “bypassed” when not helpful

**Idea:** For detecting vectors in  $\underline{y}$  using *learned* inner-product weights and thresholds, use an array of “*Perceptrons*” (P)

- Each P detects one or more memory vectors similar to its weight vectors
- The P outputs indicate *which weight-vectors are present* in the vector sum



#### Basic Idea

- One Pole Filter
- Inner Product
- Vector Memory
- Gating
- Gated RNN
- Skip Connection
- Sequences
- State Expansion
- Attention
- Dot-Product Attention
- Multi-Head Attention
- Weight Tying
- Hierarchical Blocks
- Hypersphere

#### History Samples

## Sequence Modeling

- If each vector represents a *word*, a vector sum is simply a *bag of words*
- To model a *sequence* of words, we have options:
  1. Use a *positional encoding* scheme, such as
    - (a) *Amplitude Decay* - Multiply the sum by a *forgetting factor* each sequence step (RNNs) - *poor choice* (conflates with angular distance on the hypersphere)
    - (b) *Sinusoidal Amplitude Modulation* - Add a sinusoid with *increasing frequency* to each vector summing into the history (used in the original Transformer)
    - (c) *Phase Shift* - Multiply the sum by  $e^{j\Delta}$  each sample (“RoPE”) - *apparently most used today*
  2. Use many vector-sum memories in parallel, positionally encoded (“State Expansion” in SSMs)

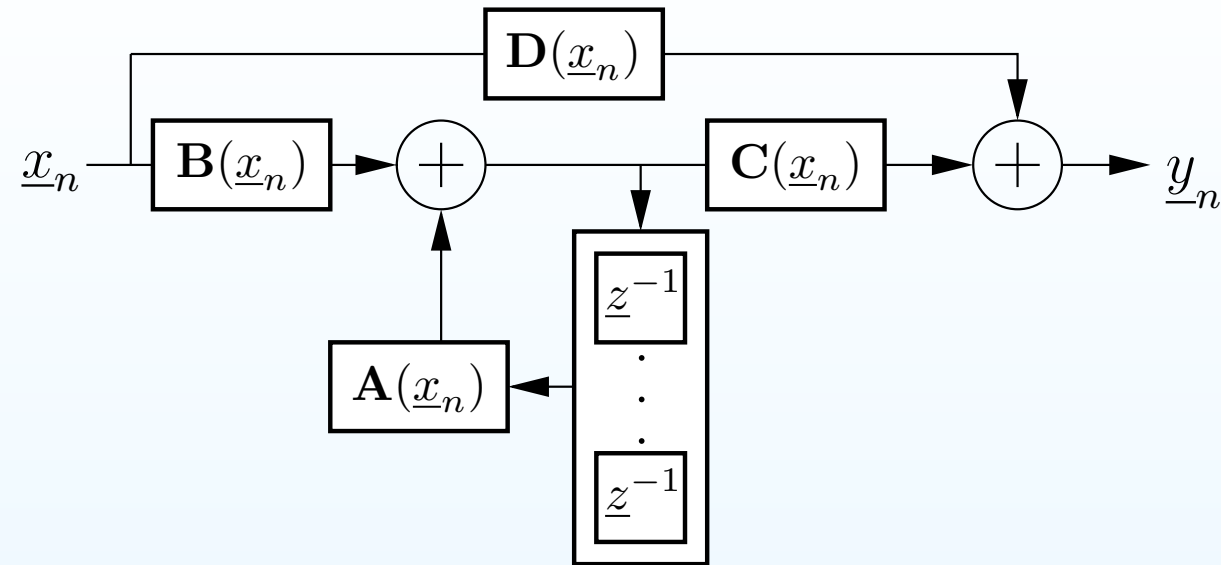


#### Basic Idea

- One Pole Filter
- Inner Product
- Vector Memory
- Gating
- Gated RNN
- Skip Connection
- Sequences
- State Expansion
- Attention
- Dot-Product Attention
- Multi-Head Attention
- Weight Tying
- Hierarchical Blocks
- Hypersphere

#### History Samples

## State Expansion



- Called a “*Structured State-space Model*” (SSM) in machine learning
- Feedback matrix  $\mathbf{A}$  has been *diagonal* since “S4D” (2022)  
 $\Rightarrow$  Parallel bank of vector one-poles (*gated, state-expanded RNNs*)
- Processed sequence (“context buffer”) is *indefinitely long*
- Gating matrices are typically simple linear input projections, *e.g.*,

$$[\mathbf{B}(\underline{x}_n), \mathbf{C}(\underline{x}_n)] = \mathbf{L} \underline{x}_n$$

(See Mamba, *e.g.*)



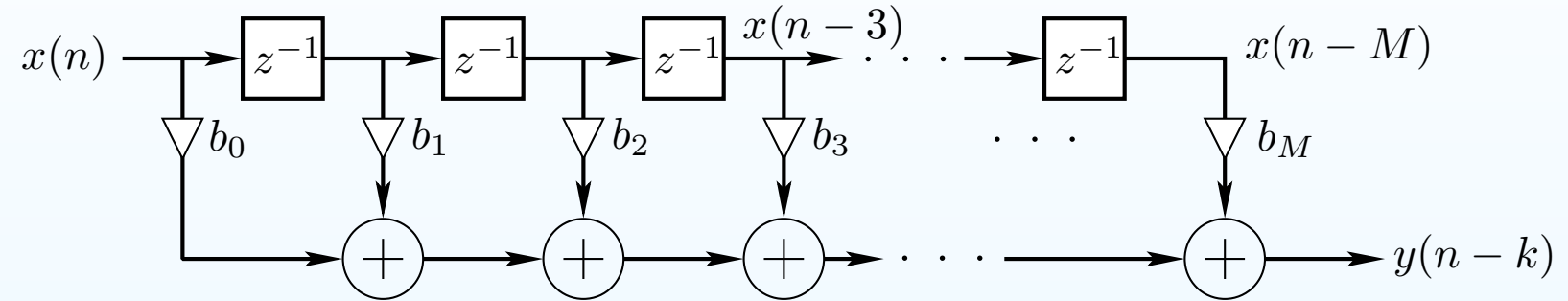
#### Basic Idea

- One Pole Filter
- Inner Product
- Vector Memory
- Gating
- Gated RNN
- Skip Connection
- Sequences
- State Expansion
- **Attention**
- Dot-Product Attention
- Multi-Head Attention
- Weight Tying
- Hierarchical Blocks
- Hypersphere

#### History Samples

## Attention Layer

**Idea:** Also use *FIR Filtering*



We can have *separately learned FIR coefficient matrices*  $b_j[\underline{x}(n-j), k]$ , which depend on the

1. input *position*  $j$  in the input sequence (“context buffer”)
2. input *vector*  $\underline{x}(n-j)$ ,  $j = 0, 1, 2, \dots, M$ , (nonlinear)
3. *output-position*  $k$  being computed,  $k = 0, 1, 2, \dots, M$  ( $M + 1$  outputs)

**Idea:** Add *relevance gating* suppressing unimportant inputs to each output (“attention”)

**Idea:** Measure relevance using an *inner product* between the output and input positions (“dot-product attention”)

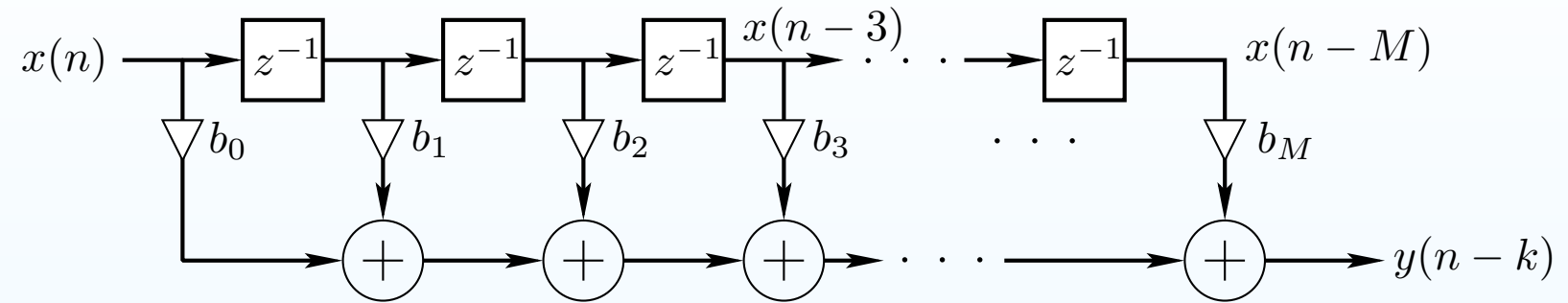


#### Basic Idea

- One Pole Filter
- Inner Product
- Vector Memory
- Gating
- Gated RNN
- Skip Connection
- Sequences
- State Expansion
- Attention
- **Dot-Product Attention**
- Multi-Head Attention
- Weight Tying
- Hierarchical Blocks
- Hypersphere

#### History Samples

## Dot-Product Attention



### Relevance Gating

Contribution from input  $\underline{x}(n-j)$  to the nonlinear FIR sum for output  $\underline{y}(n-k)$  is  $(Q_k^T K_j) \underline{x}(n-j)$ , where

- $Q_k[\underline{x}(n-k), k]$  is called the *query* vector for position  $k$  in the input sequence
- $K_j[\underline{x}(n-j), j]$  is called the *key* vector for position  $j$  in the input sequence

**Idea:** To provide more flexibility for the attention sum, replace  $\underline{x}(n-j)$  in the attention sum with a learned *value vector*  $V_j[\underline{x}(n-j)]$

$\Rightarrow$

Relevance of input  $j$  to output  $k \propto (Q_k^T K_j) V_j$ , where all three vectors are learned projections of  $\underline{x}(n-j)$  for each  $(j, k)$  (“Transformer”)



#### Basic Idea

- One Pole Filter
- Inner Product
- Vector Memory
- Gating
- Gated RNN
- Skip Connection
- Sequences
- State Expansion
- Attention
- Dot-Product Attention
- Multi-Head Attention
- Weight Tying
- Hierarchical Blocks
- Hypersphere

#### History Samples

## Multi-Head Attention

**Idea:** To support multiple meaning possibilities, *partition the model space* into parallel independent *attention calculations* (“multi-head attention”)

- Each *attention head* can form an independent input interpretation
- Useful for *ambiguous* sequences, especially in the lower layers
- Also introduced in the Transformer paper (2017)



#### Basic Idea

- One Pole Filter
- Inner Product
- Vector Memory
- Gating
- Gated RNN
- Skip Connection
- Sequences
- State Expansion
- Attention
- Dot-Product Attention
- Multi-Head Attention
- **Weight Tying**
- Hierarchical Blocks
- Hypersphere

#### History Samples

## Weight Tying

When the sequence model maps input to output in the same “language” (e.g., English to English), it makes sense to use the *same embedding vectors* at the input and output layers, instead of separately learning a set of weights for mapping to the final output. This is called “weight tying” (many fewer parameters, better results).





#### Basic Idea

- One Pole Filter
- Inner Product
- Vector Memory
- Gating
- Gated RNN
- Skip Connection
- Sequences
- State Expansion
- Attention
- Dot-Product Attention
- Multi-Head Attention
- Weight Tying
- Hierarchical Blocks
- Hypersphere

#### History Samples

## Hierarchical Blocks

- Cascade blocks of attention + MLP and/or gated recurrence + MLP to model *hierarchical relationships* like image features or grammatical constructs
- Attention and gated RNNs are called “mixing layers” (successive inputs are combined)
- MLPs are called “point transformations” (general mapping of any vector from one place to another)
- RMSNorm typical at the input to put it on the hypersphere — also used internally (see Hawk/Griffin e.g.)
- Thus, these point MLPs are effectively data-dependent *rotation matrices*.  
In model-dimension  $N$ , we need only learn  $N - 1$  *rotation angles* for each input class.  
The direct mapping of the coordinates used now seems efficient enough, but projection back to the sphere during training might help. Then perhaps there is no need for RMSNorm at the output after summing the skip connection.



## Where Meaning Lies

### Basic Idea

- One Pole Filter
- Inner Product
- Vector Memory
- Gating
- Gated RNN
- Skip Connection
- Sequences
- State Expansion
- Attention
- Dot-Product Attention
- Multi-Head Attention
- Weight Tying
- Hierarchical Blocks
- **Hypersphere**

### History Samples



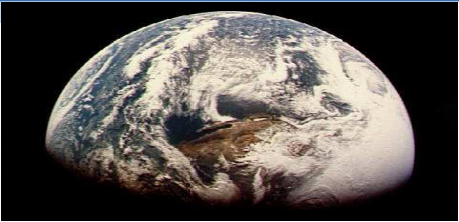


[Basic Idea](#)

[History Samples](#)

## Sequence Modeling Snapshots





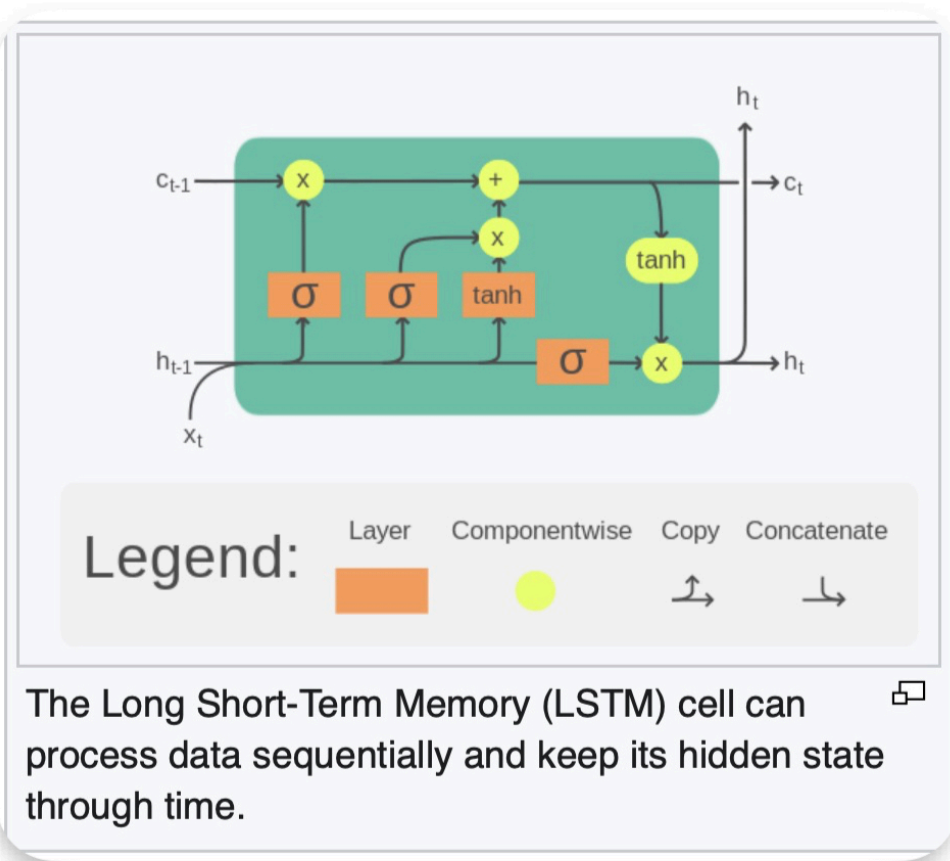
# LSTM and GRU

## Basic Idea

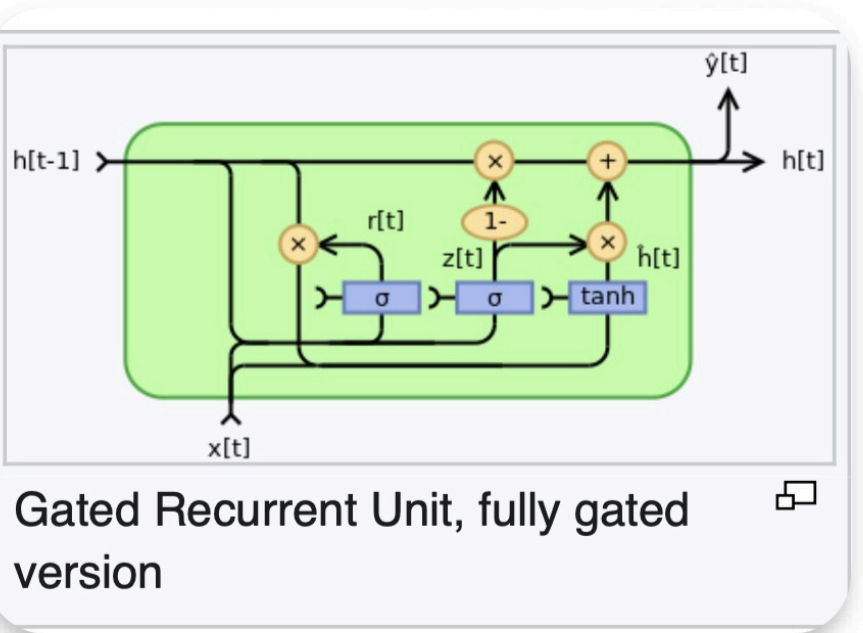
## History Samples

- LSTM & GRU
- SSM & Mamba
- Hawk & Griffin
- HGRN2
- RWKV+

## 1997: LSTM



## 2014: GRU





## Basic Idea

## History Samples

- LSTM & GRU
- SSM & Mamba
- Hawk & Griffin
- HGRN2
- RWKV+

# Structured State Space and Mamba

## 2023: Mamba (S6)

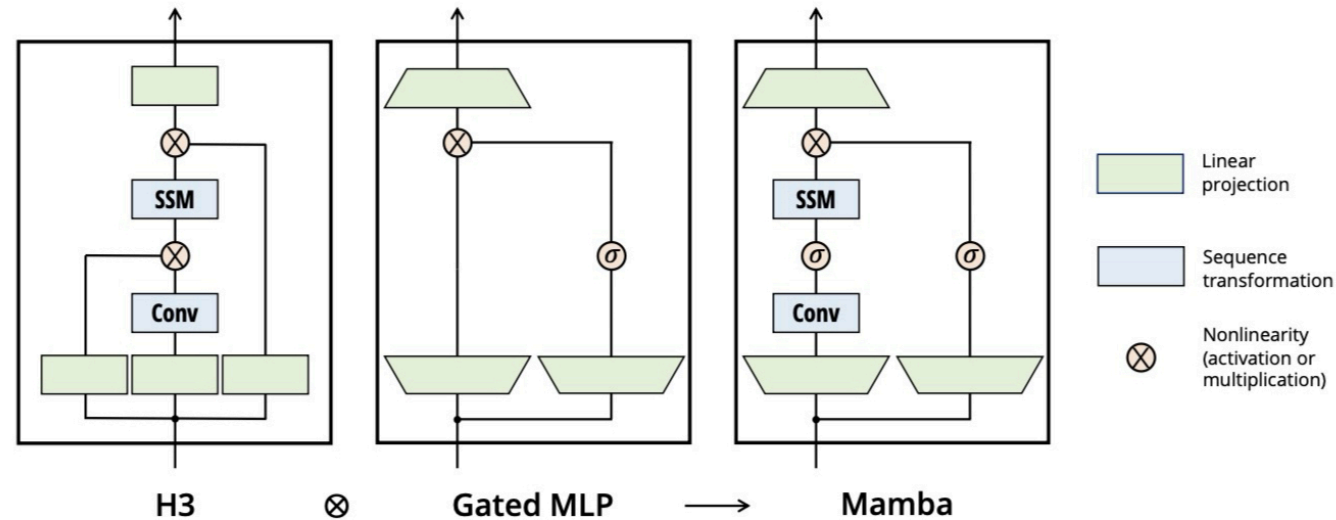


Figure 2: (**Architecture.**) Our simplified block design combines the H3 block, which is the basis of most SSM architectures, with the ubiquitous MLP block of modern neural networks. Instead of interleaving these two blocks, we simply repeat the Mamba block homogenously. Compared to the H3 block, Mamba replaces the first multiplicative gate with an activation function. Compared to the MLP block, Mamba adds an SSM to the main branch. For  $\sigma$  we use the SiLU / Swish activation ([Hendrycks & Gimpel, 2016](#); [Ramachandran et al., 2017](#)).

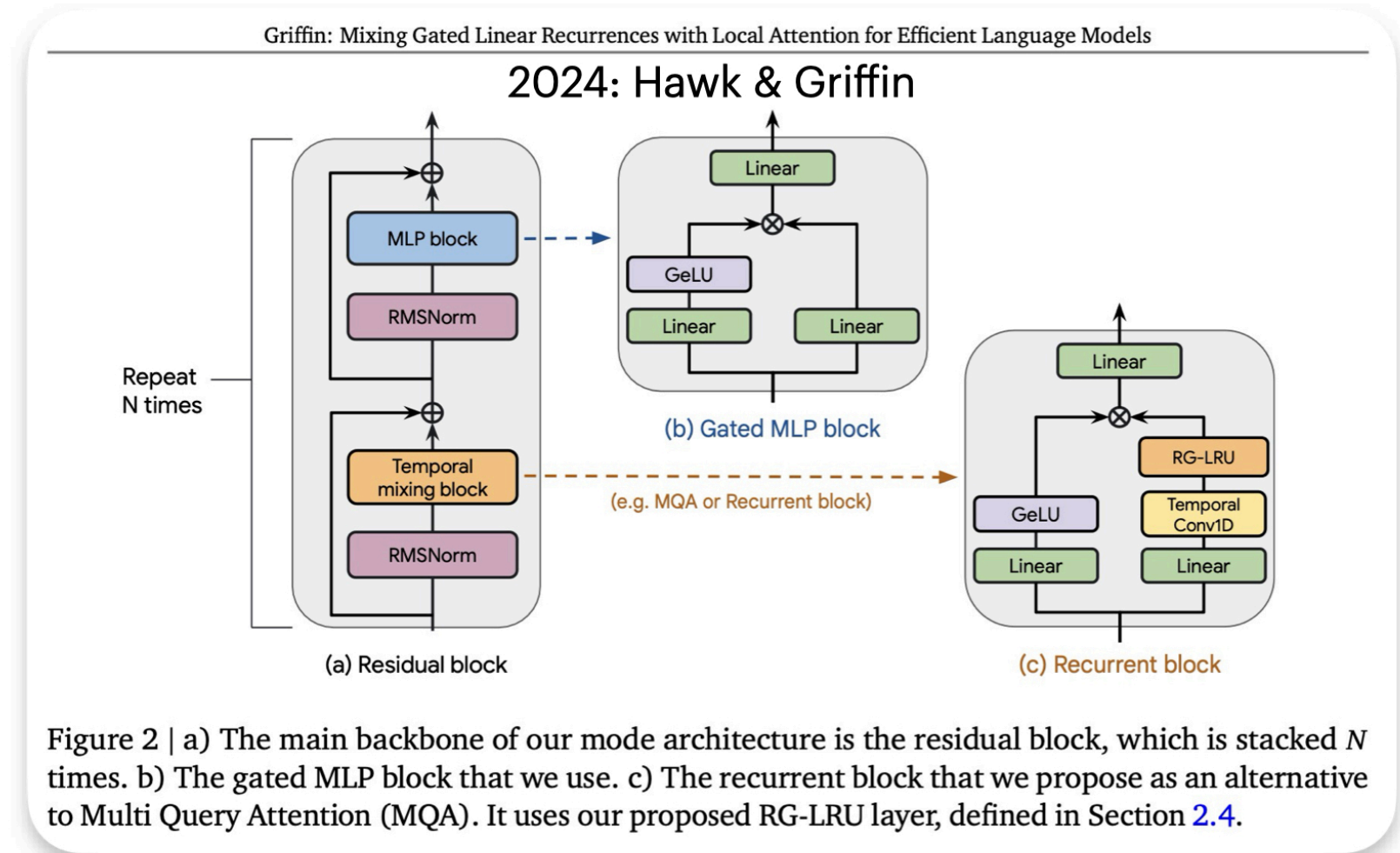


# Hawk and Griffin

## Basic Idea

## History Samples

- LSTM & GRU
- SSM & Mamba
- Hawk & Griffin
- HGRN2
- RWKV+



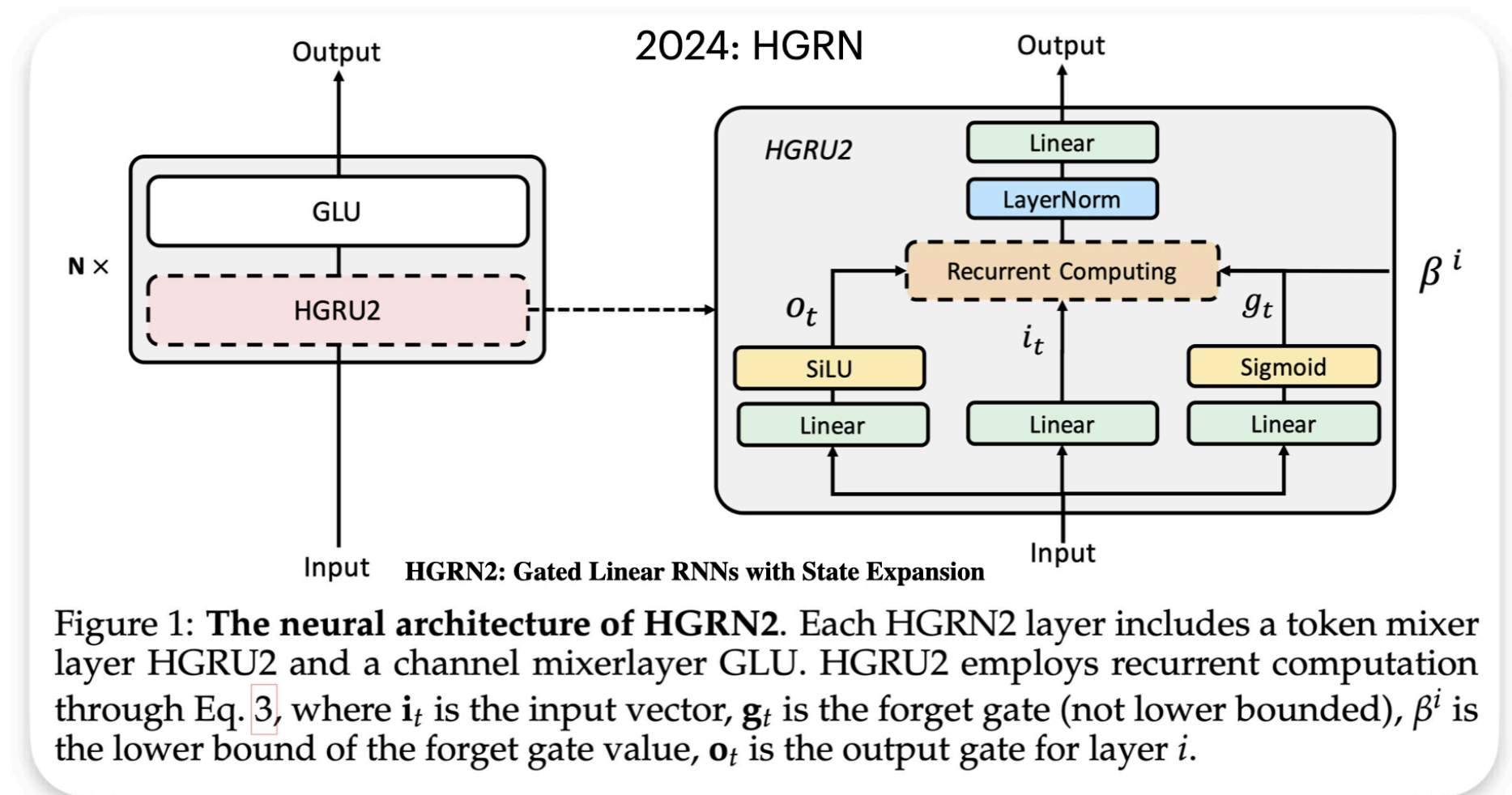


## Basic Idea

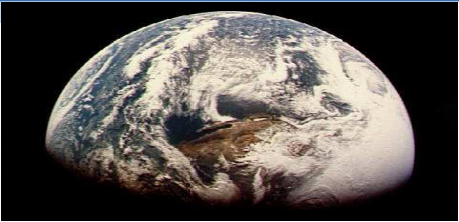
## History Samples

- LSTM & GRU
- SSM & Mamba
- Hawk & Griffin
- HGRN2
- RWKV+

# Gated Linear RNNs with State Expansion







# RWKV, Eagle, Finch

## Basic Idea

## History Samples

- LSTM & GRU
- SSM & Mamba
- Hawk & Griffin
- HGRN2
- RWKV+

## 2024: Eagle-Finch RWKV

