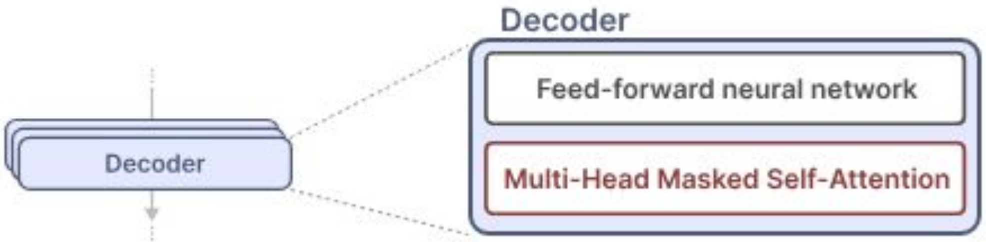
Mamba

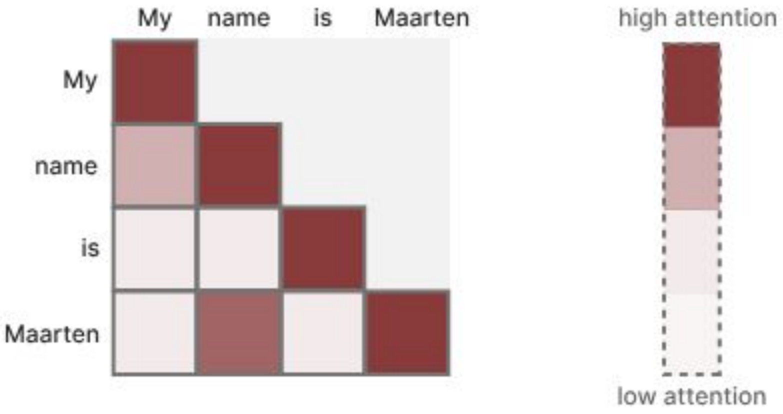
Linear-Time Sequence Modeling with Selective State Spaces

The Problem with Transformers

- Transformer is capable of selectively and individually looking at the past tokens.
- We can create generative models by using only decoders. GPT uses decoder blocks to complete some input text.
- Self-attention enables an uncompressed view of the entire sequence with fast training.



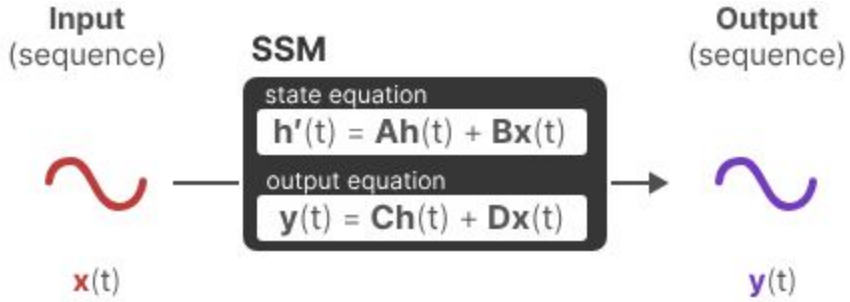
- Self-attention creates a matrix comparing each token with every token that came before (weights - how relevant the token pairs are to one another)
- During training matrix is created in one go.
- Enables parallelization, which speeds up training.



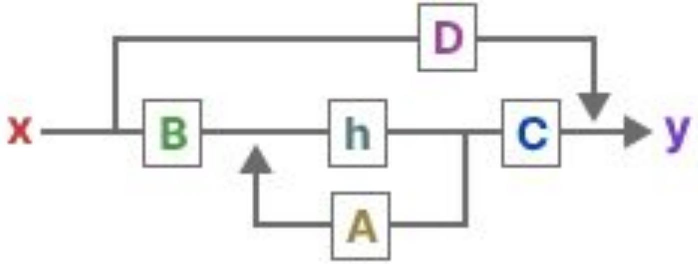
- During Inference, when generating the next token, we need to re-calculate the attention for the entire sequence, even if we already generated some tokens.
- Generating tokens for a sequence of length L needs L^2 computations (costly)
- Inference is slow and scales quadratically with sequence length.

State Space Model (SSM)

- SSM models also processes sequences of information, like text and signals.
- At time t, SSMs have:
  - input sequence x(t)
  - state representation h(t)
  - predicted output sequence y(t)
- Predict the state of a system based on observed data (input sequence and previous state)



- Matrices A, B, C, and D are parameters, they are learnable.
- Matrix A describes how all the internal states are connected. It is updated after the state representation h(t) has been updated.

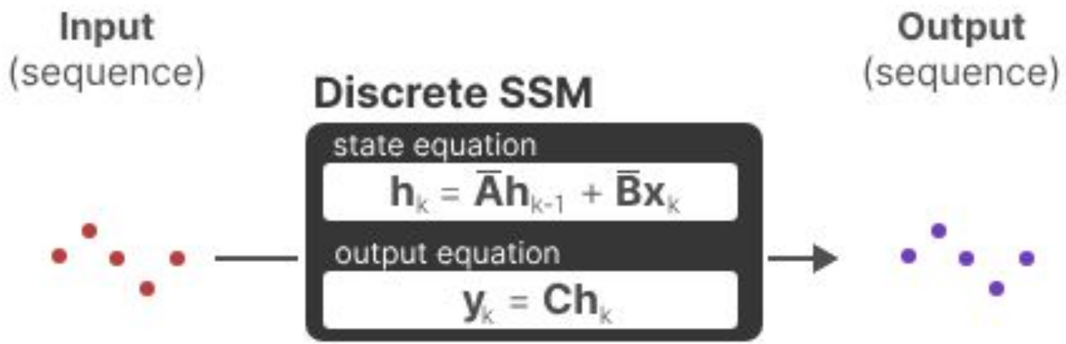


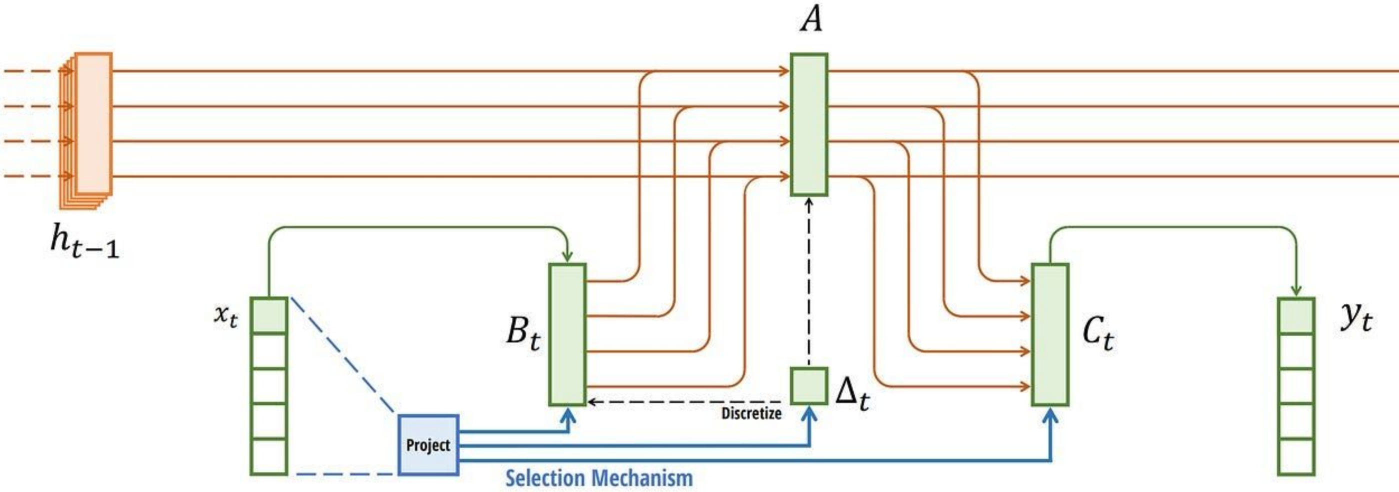
Discrete SSM

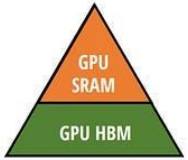
- Finding the state representation h(t) is analytically challenging if you have a continuous signal.
- We generally have a discrete input (like a textual sequence), so we want to discretize the model (Zero-order hold technique)
- How long we hold the value is represented by a new learnable parameter, called the step size Δ.
-

Discretized matrix **A** $$\bar{\mathbf{A}} = \exp(\Delta \mathbf{A})$$

Discretized matrix **B** $$\bar{\mathbf{B}} = (\Delta \mathbf{A})^{-1}(\exp(\Delta \mathbf{A}) - I) \cdot \Delta \mathbf{B}$$

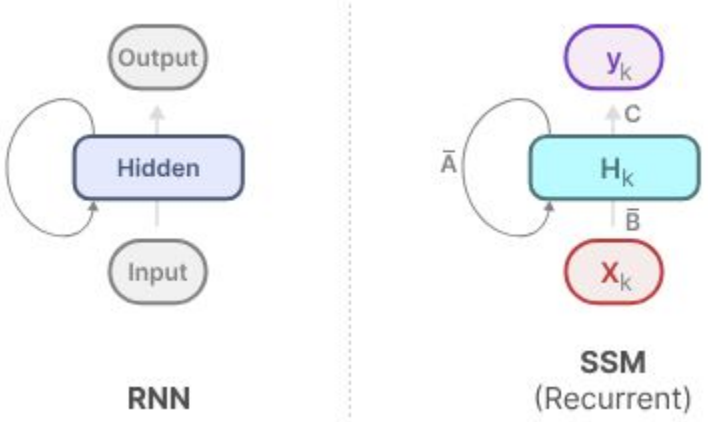

Input
(sequence)
Discrete SSM
state equation
$h_k = \bar{A}h_{k-1} + \bar{B}x_k$
output equation
$y_k = Ch_k$
Output
(sequence)


A
h_{t-1}
h_t
x_t
B_t
Δ_t
C_t
y_t
Project
Discretize
Selection Mechanism


GPU
SRAM
GPU HBM

Recurrent Representation

- Using Discrete SSM we can make the SSM model as Recurrent model (like RNN)
- It gives the advantage and disadvantage of RNN, fast inference (scales linearly) and slow training.
-



Convolution Representation

$$\text{kernel} \longrightarrow \overline{K} = (C\overline{B}, C\overline{AB}, \dots, C\overline{A}^{k}\overline{B}, \dots)$$

$$\underset{\text{output}}{y} = \underset{\text{input}}{x} * \underset{\text{kernel}}{\overline{K}}$$

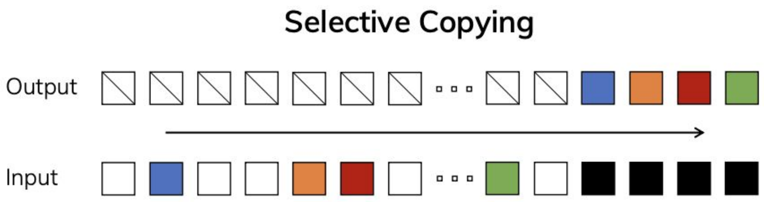
- By representing discrete SSM as convolution, it can be trained parallel like CNNs.
- Due to fixed kernel size, inference is not fast as recurrent SSM.

Linear State-Space Layer (LSSL)

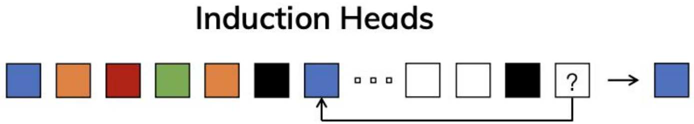
- During training, we use convolutional SSM representation - Parallelizable
- During inference, we use recurrent SSM representation - Faster, efficient, scales linearly
- These representations have property Linear Time Invariance (LTI)
- SSM param like A, B and C are fixed for all timesteps and remains same for every sequence (not content aware) - disadvantage

Mamba - Selective SSM

- SSM perform poorly on language modeling and generation, ability to focus on or ignore particular inputs.
- The Selective Copying task has random spacing in between inputs and requires time-varying models that can selectively remember or ignore inputs depending on their content, requires content-aware reasoning.



- The Induction Heads task is an example of associative recall that requires retrieving an answer based on context, requires context-aware reasoning.



- SSM (conv/recu) perform poorly in selective copying and induction heads tasks, since it is Linear Time Invariant.
- But these tasks are easy for Transformer, they can dynamically change their attention based on the input sequence.

Selectively retain information

- The recurrent SSM creates a small state that is quite efficient as it compresses the entire history.
- Transformer model does not compress the history (using attention matrix), so it is more powerful but less efficient.
- Mamba has a small state which is powerful as the Transformer and more efficient, since it compress data selectively.

- Mamba takes matrix B, C and step size Δ(discretization param) dependent on the input (dynamic) by incorporating the sequence length and batch size of the input - (content-awareness).
- They selectively choose what to keep and what to ignore in the hidden state.

Scan operation

- Since the matrices are now dynamic, they cannot be calculated using the convolution representation since it assumes a fixed kernel.
- We can only use the recurrent representation and lose the parallelization.
- Using Selective scan algorithm, we can calculate the sequences in parts and iteratively combine them, so it gives the parallelization ability for Mamba model.
- It makes the training in Mamba faster and parallelizable.

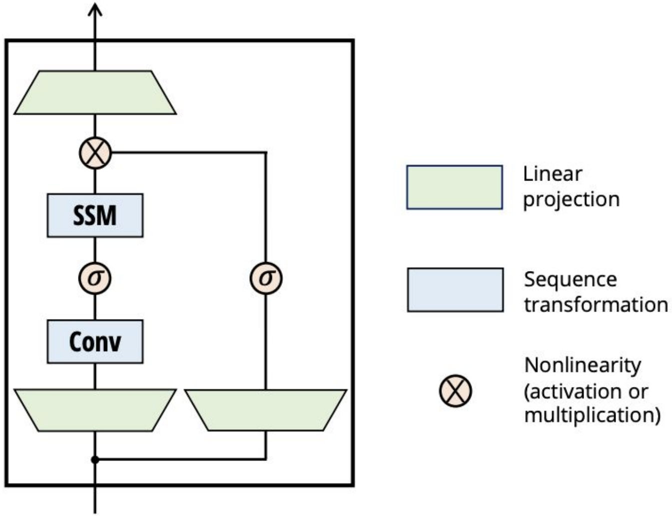
Hardware-aware Algorithm

- Disadvantage of recent GPUs is their limited transfer (IO) speed between their small but highly efficient SRAM and their large but slightly less efficient DRAM.
- Frequently copying information between SRAM and DRAM becomes a bottleneck.
- Mamba limits the number of times we need to go from DRAM to SRAM and vice versa.
- It uses kernel fusion which allows the model to prevent writing intermediate results and continuously performing computations until it is done.

Mamba block

Like decoder in Transformers, we can stack multiple Mamba blocks and use their output as the input for the next Mamba block.

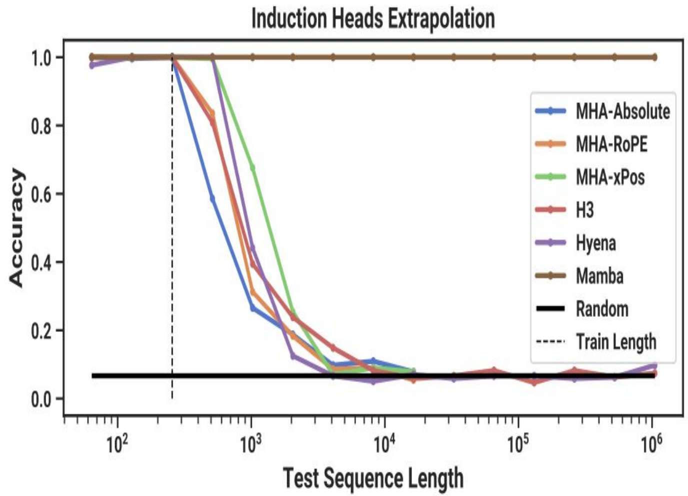
- It starts with a linear projection to expand
- upon the input embeddings.
- A convolution is applied to prevent
- independent token calculations.
- For σ we use the SiLU / Swish activation



Empirical Evaluation - Synthetic Tasks

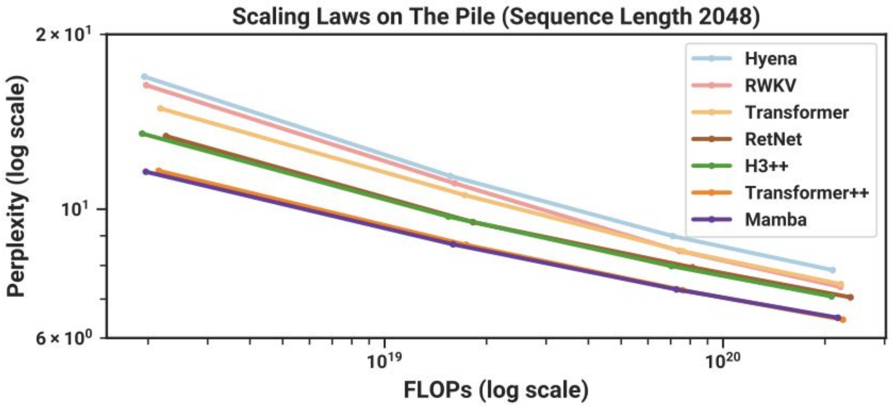
Mamba demonstrates the ability to solve important synthetic tasks such as selective copying and induction heads, extrapolating solutions indefinitely long (>1M tokens).

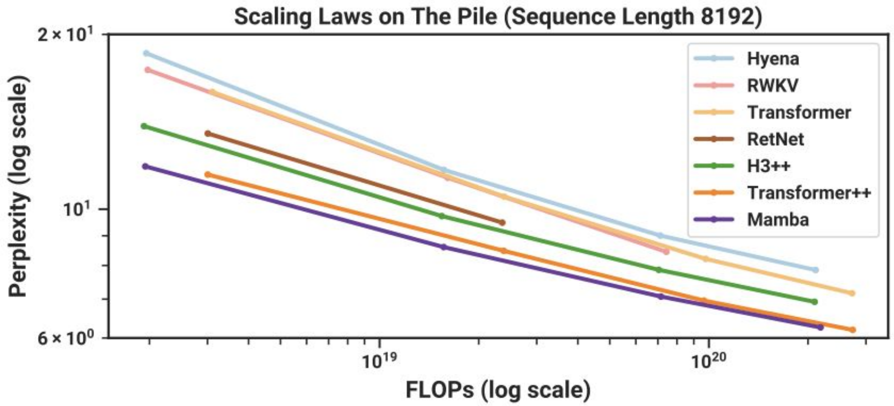
Model	Arch.	Layer	Acc.
S4	No gate	S4	18.3
-	No gate	S6	**97.0**
H3	H3	S4	57.0
Hyena	H3	Hyena	30.1
-	H3	S6	**99.7**
-	Mamba	S4	56.4
-	Mamba	Hyena	28.4
Mamba	Mamba	S6	**99.8**



Empirical Evaluation - Language Modeling

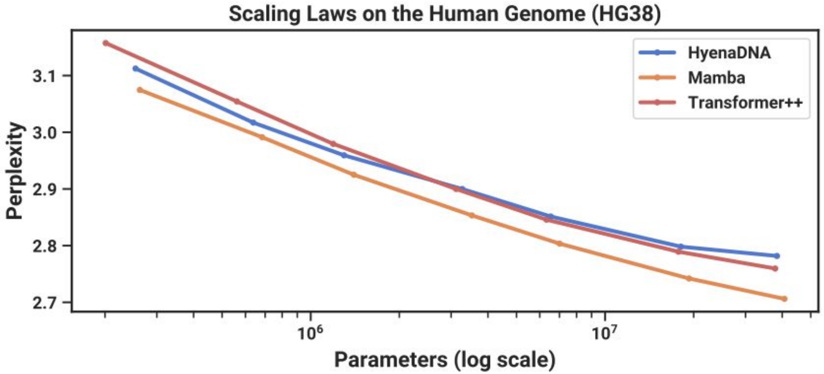
- Mamba achieves Transformer-quality performance in pretraining perplexity and downstream evaluations, with 5× generation throughput compared to Transformers of similar size.
- Mamba scales better than all other models as the sequence length grows.

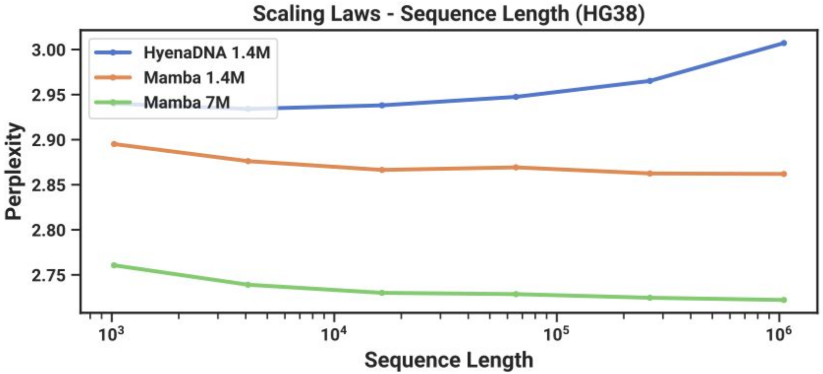


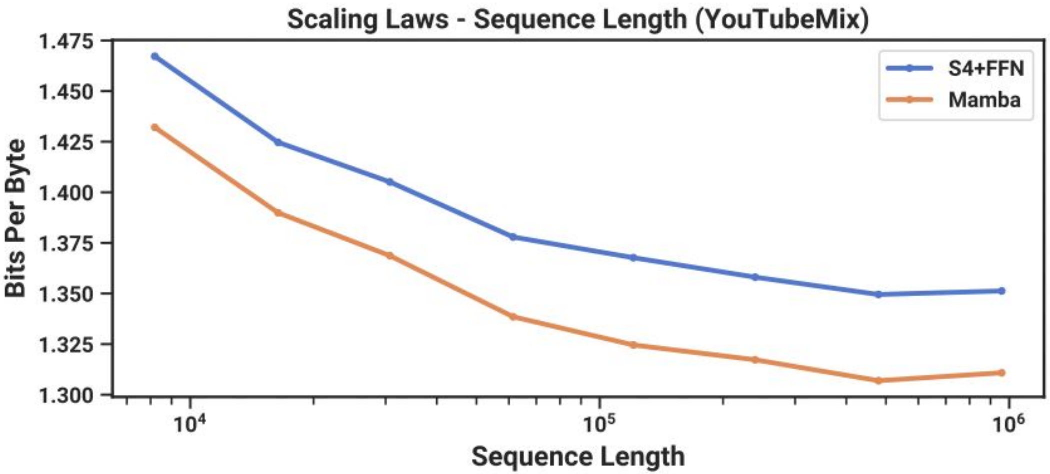


Empirical Evaluation - Audio and Genomics Modeling

- Mamba outperforms prior models on modeling audio waveforms and DNA sequences, demonstrating improved performance with longer context up to million-length sequences.
- Mamba facilitates better performance with increasing context length and model size







Scaling Laws - Sequence Length (YouTubeMix)
S4+FFN
Mamba
Bits Per Byte
1.475
1.450
1.425
1.400
1.375
1.350
1.325
1.300
10^4
10^5
10^6
Sequence Length

Computational Efficiency

- Mamba's efficient scan is 40× faster than a standard implementation during training, and achieves 5× higher throughput than Transformers during inference, making it a practical and efficient choice for large-scale sequence modeling tasks.
- The model's linear scaling in sequence length during training and its ability to handle long contexts effectively position it as a scalable solution for diverse applications.

Future Research

Open-Source Model Code: The authors have open-sourced the model code and pre-trained checkpoints to facilitate further research and application, empowering the broader research community to leverage and build upon the Mamba architecture.

Conclusion

- Mamba perform context-dependent reasoning while scaling linearly in sequence length.
- It matches or exceeds the performance of strong Transformer models.
- Mamba is positioned as a general sequence model backbone with the potential to impact various domains, such as genomics, audio, and video.