What could be the Next Generation of LLMs?

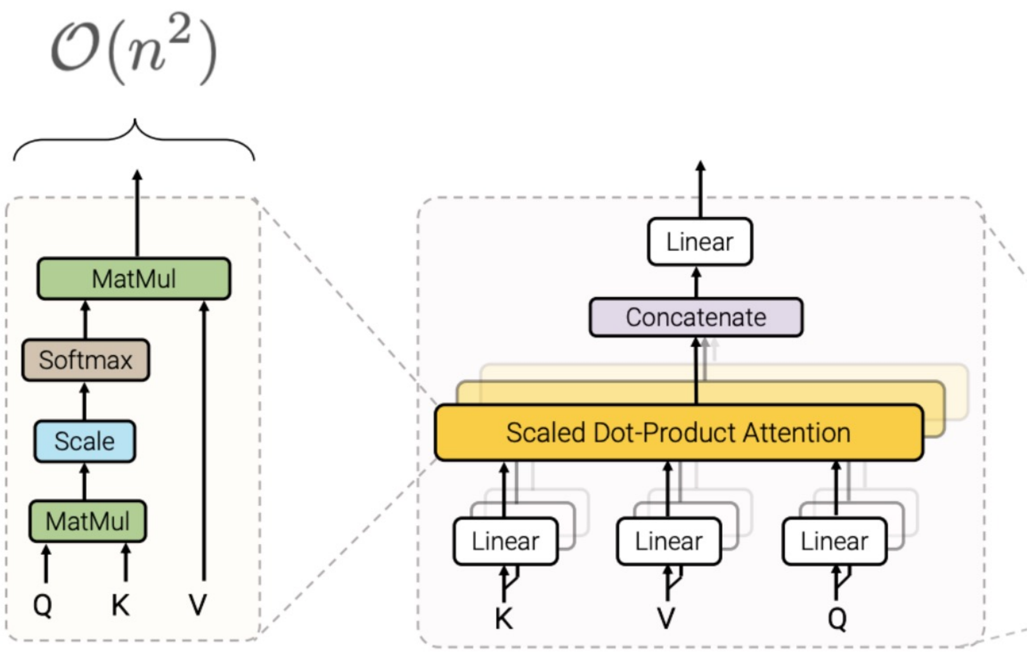
Zilin Xiao

Apr 16th, 2024

2024 Spring COMP 646 Guest Talk

What’s wrong with current LLMs?

- Transformer-based LLMs: suffers from quadratic complexity of self-attention w.r.t. sequence length.
  - This makes it hard to scale native Transformer to long-context problems.



Ref: Yi et al. Efficient Transformers: A Survey

Why do we need long context language model?

- Imagine you have to summarize a 4-page paper within 10mins.
  - If you only have access to a GPT-2 with 512-token context window, you might have to select relevant content as your prompt.
  - If you have a GPT-4-turbo with 128K context window, throwing the entire paper won't be a problem.

- Things become interesting when the context window scales…
  - Chat with a PDF file, a webpage and even a book!
  - Information Retrieval over a massive dataset.
  - Anything else? Be creative!

Why do we need long context language model?

MODEL	DESCRIPTION	CONTEXT WINDOW	TRAINING DATA
gpt-4-turbo	New **GPT-4 Turbo with Vision** The latest GPT-4 Turbo model with vision capabilities. Vision requests can now use JSON mode and function calling. Currently points to gpt-4-turbo-2024-04-09.	128,000 tokens	Up to Dec 2023
gpt-4-turbo-2024-04-09	GPT-4 Turbo with Vision model. Vision requests can now use JSON mode and function calling. gpt-4-turbo currently points to this version.	128,000 tokens	Up to Dec 2023
gpt-4	Currently points to gpt-4-0613. See continuous model upgrades.	8,192 tokens	Up to Sep 2021
gpt-4-0613	Snapshot of gpt-4 from June 13th 2023 with improved function calling support.	8,192 tokens	Up to Sep 2021

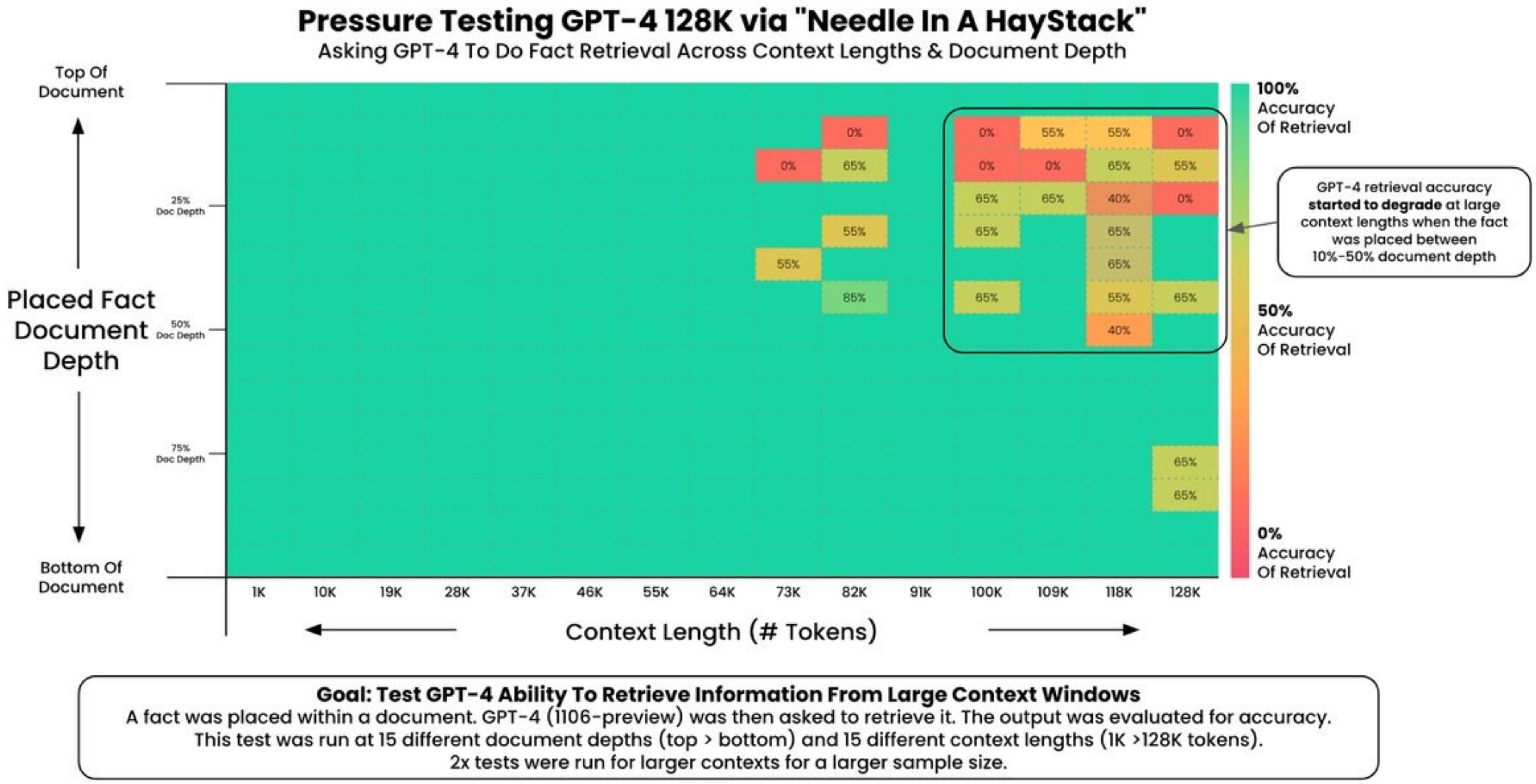
MODEL	DESCRIPTION	CONTEXT WINDOW	TRAINING DATA
gpt-3.5-turbo-0125	New **Updated GPT 3.5 Turbo** The latest GPT-3.5 Turbo model with higher accuracy at responding in requested formats and a fix for a bug which caused a text encoding issue for non-English language function calls. Returns a maximum of 4,096 output tokens. Learn more.	16,385 tokens	Up to Sep 2021
gpt-3.5-turbo	Currently points to gpt-3.5-turbo-0125.	16,385 tokens	Up to Sep 2021
babbage-002	Replacement for the GPT-3 ada and babbage base models.	16,384 tokens	Up to Sep 2021
davinci-002	Replacement for the GPT-3 curie and davinci base models.	16,384 tokens	Up to Sep 2021

From 8k to 128k, 16x increase in context window!
If still in stanard GPT architecture, that means 256x training compute (time and memory)!

Passkey Retrieval: the easiest needle-in-the-haystack experiment

- Prompt: There is an important info hidden inside a lot of irrelevant text. Find it and memorize them. I will quiz you about the important information there.
- Document:
  - The grass is green. The sky is blue. The sun is yellow. Here we go. There and back again. The grass is green. The sky is blue. The sun is yellow. Here we go. There and back again… The pass key is **joidYG+FD)**. Remember it. The grass is green. The sky is blue. The sun is yellow. Here we go. There and back again…
- Query: What is the pass key? The pass key is

How do we evaluate long-context LMs?



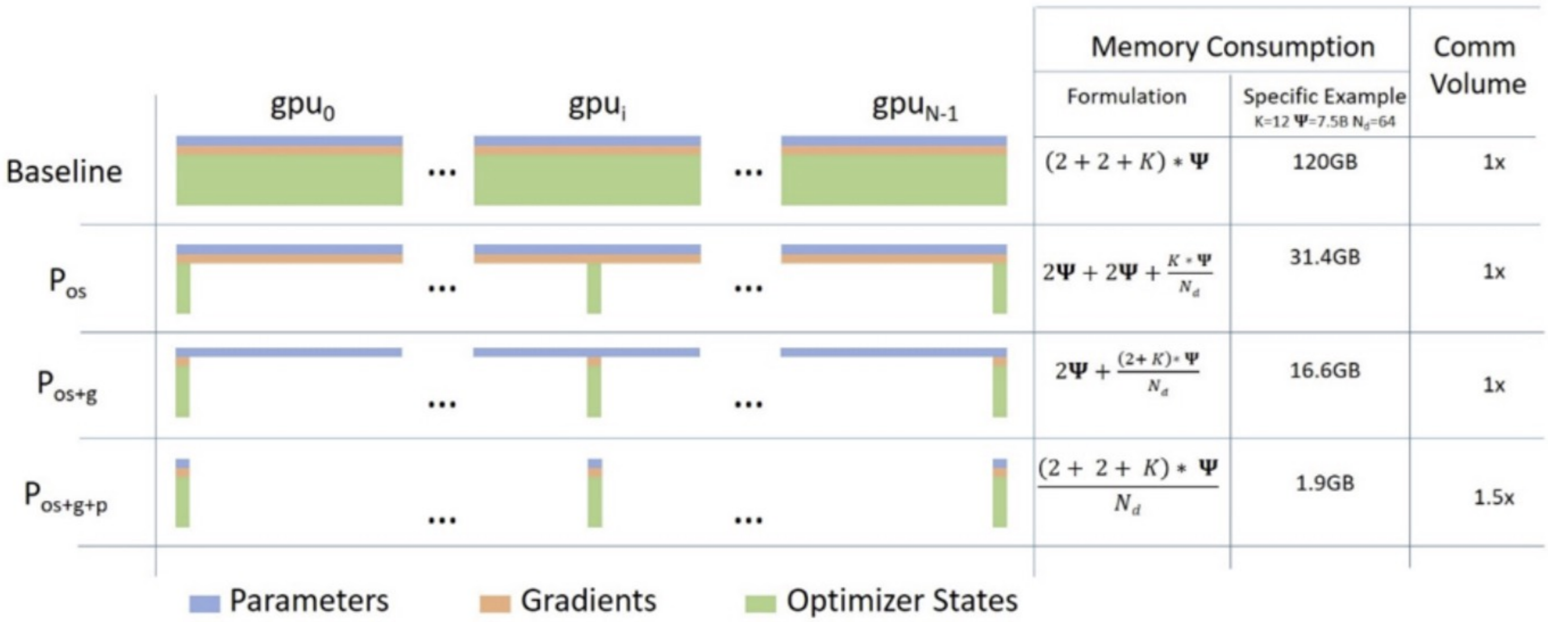
Ref: https://x.com/GregKamradt/status/1722386725635580292

The Future of LLM

- GPT models are growing, but still limited by context length.
  - Training Speed - Cost is quadratic in length
  - Generation Speed - Attention requires full lookback
- Solutions Proposed:
  - A) Approximation (e.g. Sparse, LoRA)
  - B) RAG / Vector-DBs (ANN search, LSH)
  - C) Brute-force compute (tiling, blockwise, e.g. RingAttention)
  - D) **Recurrent Model (What we will mainly discuss today!)**

Ref: https://github.com/srush/do-we-need-attention/blob/main/DoWeNeedAttention.pdf
https://docs.google.com/presentation/d/180lS8XbeR1_bTMaldg21LKYQkjXftHuh9VnZ3xk27qQ/edit

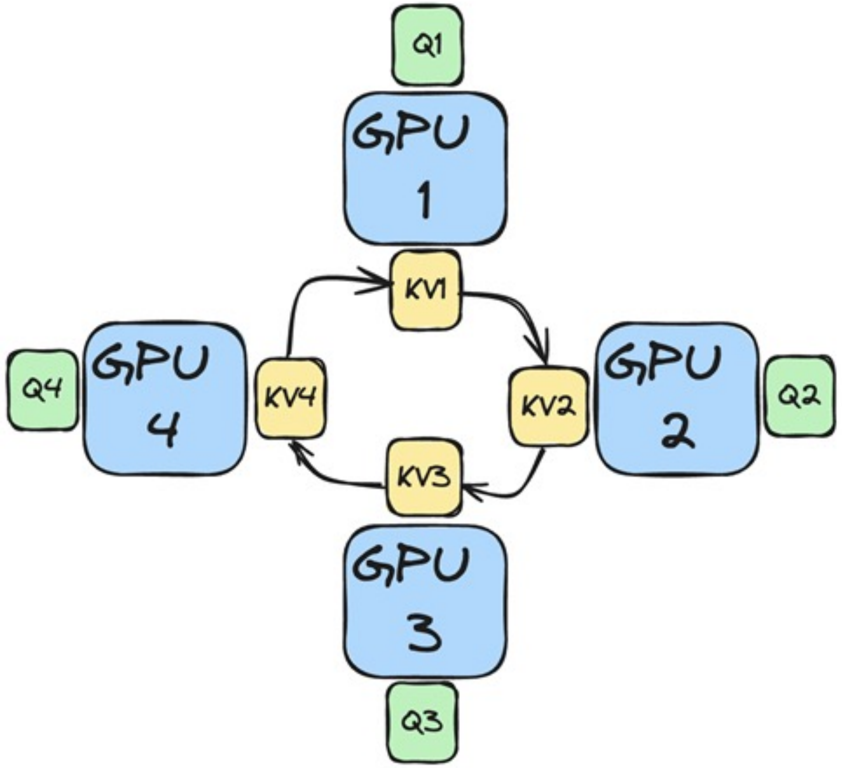
Prerequisite: Zero Redundancy Distributed Training

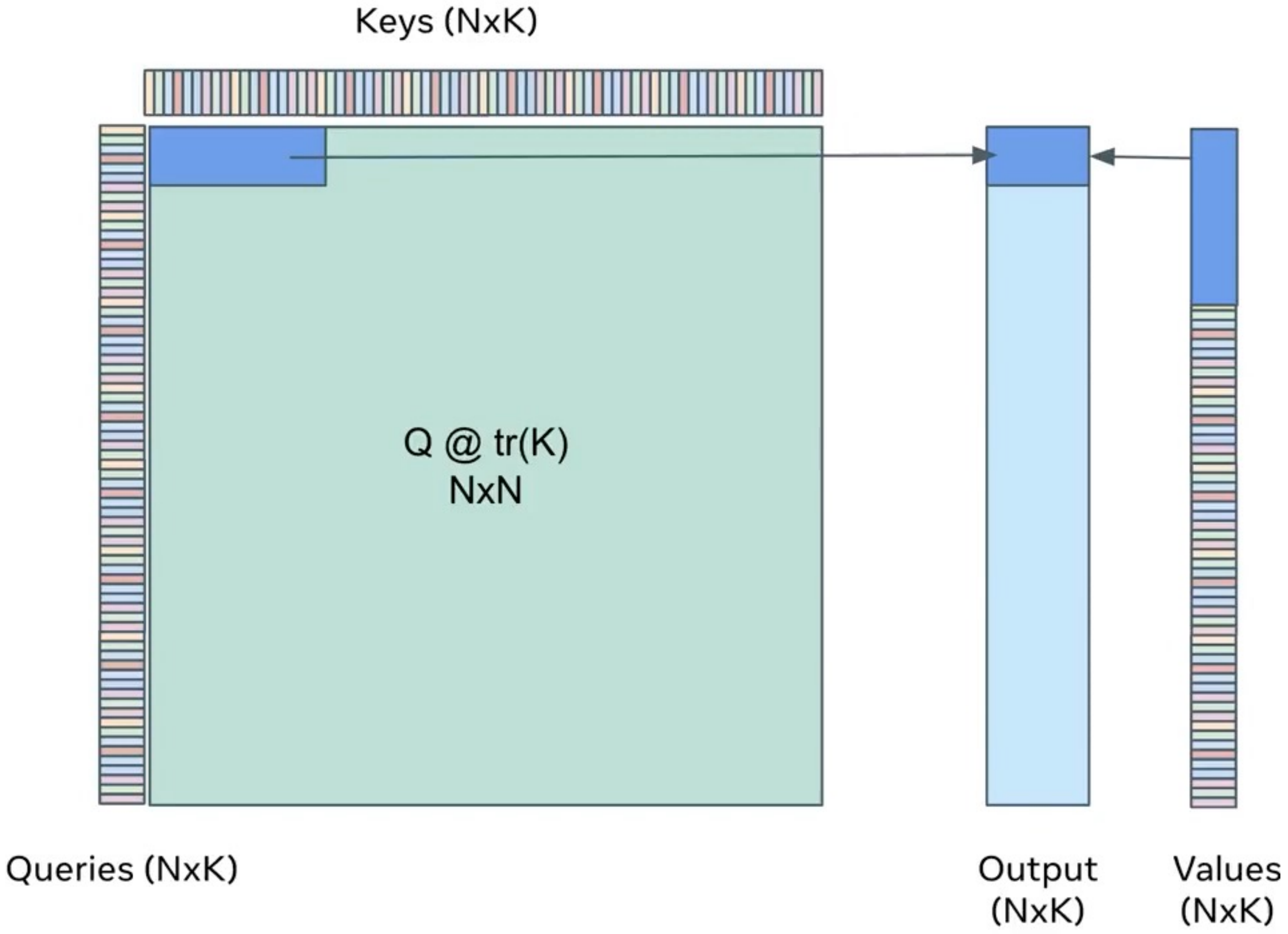


This does not solve long-context problem at all!
Because multi-head self-attention is still performed on single device, which is prohibitive for long-context input.

Ref: https://www.deepspeed.ai/2021/03/07/zero3-offload.html

RingAttention: a type of Tensor Parallelism

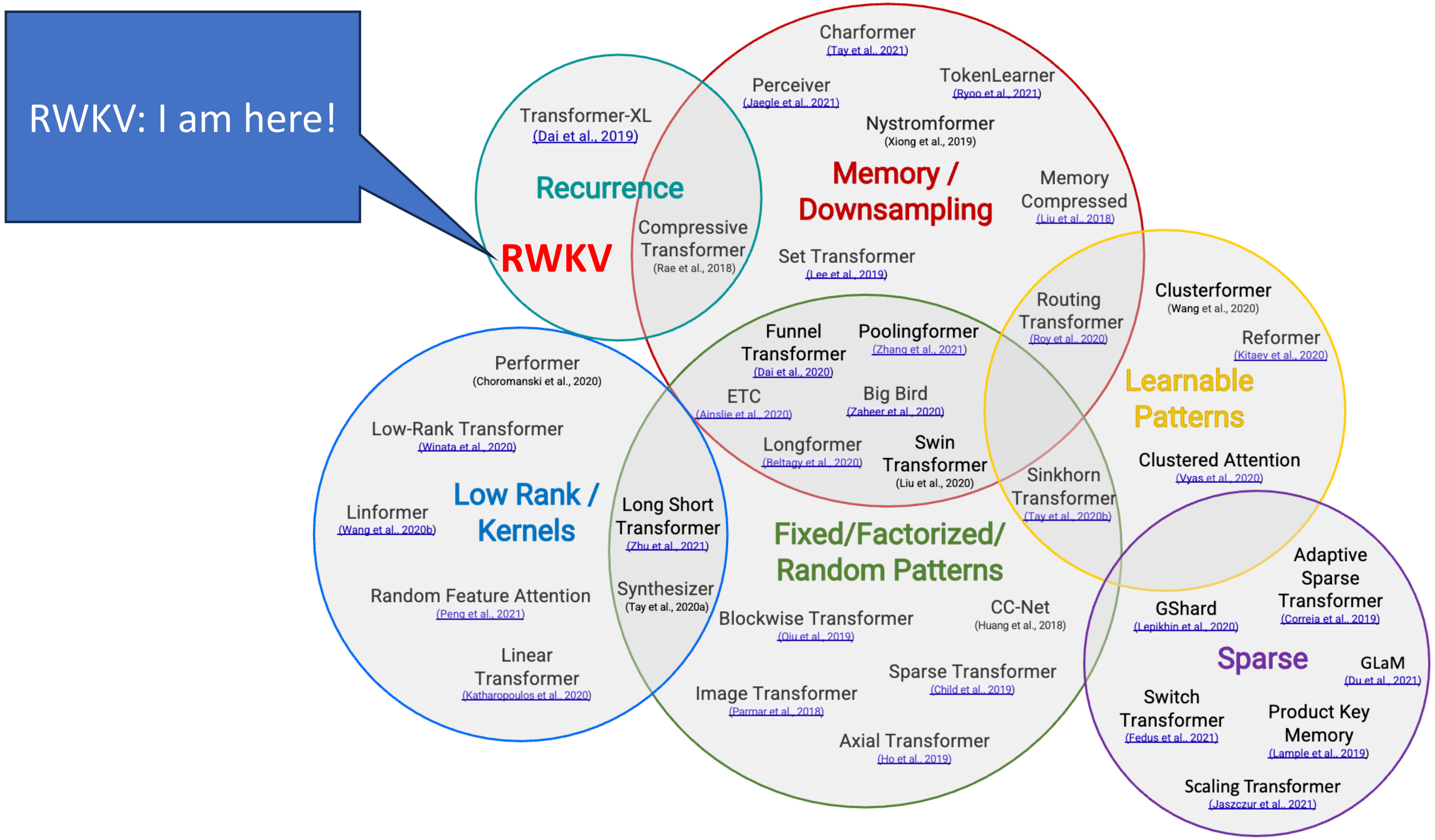




Pro: it's still self-attention without any sparse approximation. And it is widely adopted in modern LLM product.
Con: Still quadric complexity!

Ref: https://twitter.com/fvsmassa/status/1580229170629849089

Alternatives towards Long-Context Problems



Ref: Yi et al. Efficient Transformers: A Survey

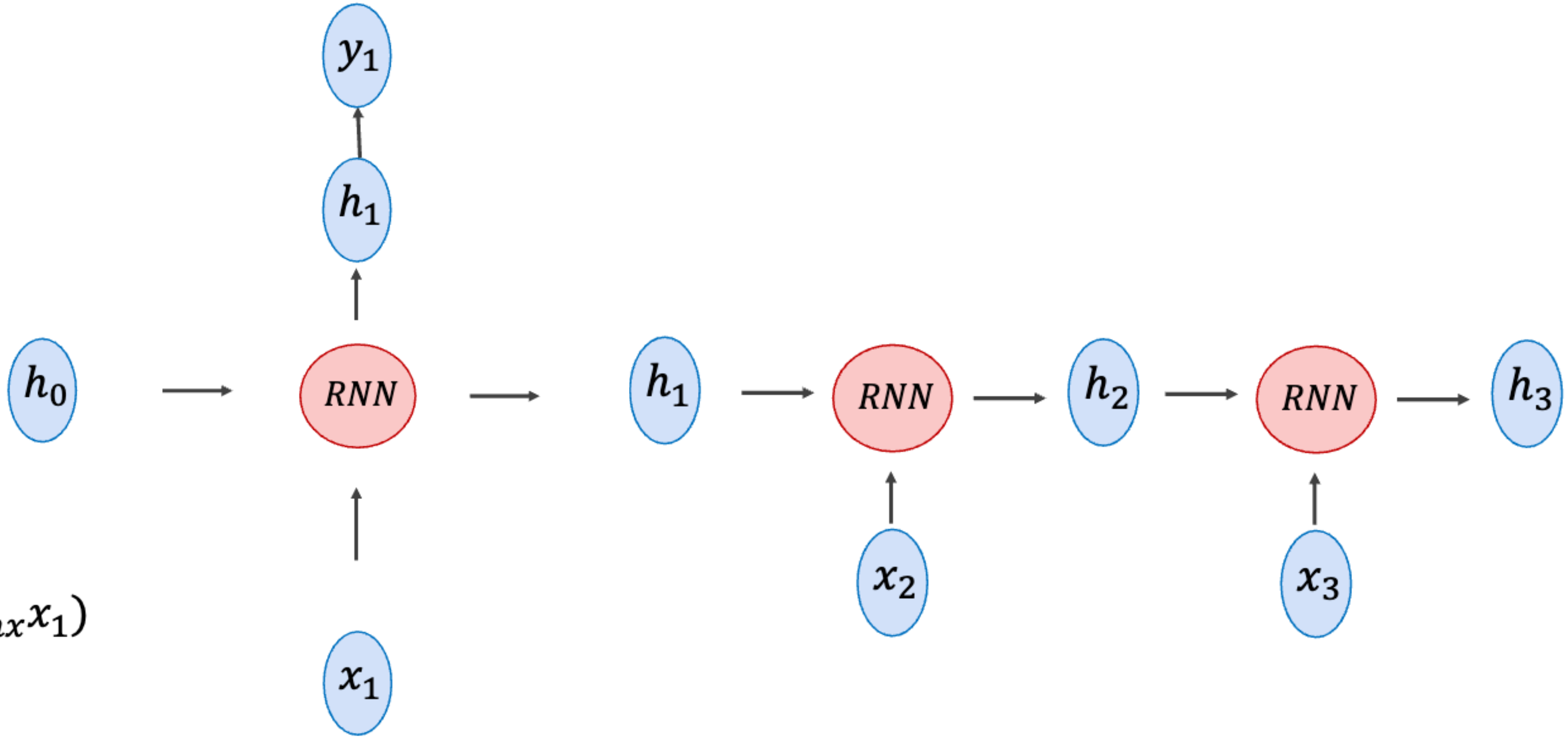
RWKV Explained

RWKV: Reinventing RNNs for the Transformer Era

Bo Peng[1,2]* Eric Alcaide[2,3,4]* Quentin Anthony[2,5]*
Alon Albalak[2,6] Samuel Arcadinho[2,7] Stella Biderman[2,8] Huanqi Cao[9] Xin Cheng[10]
Michael Chung[11] Xingjian Du[1] Matteo Grella[12] Kranthi Kiran GV[2,13] Xuzheng He[2]
Haowen Hou[14] Jiaju Lin[1] Przemysław Kazienko[15] Jan Kocoń[15] Jiaming Kong[16]
Bartłomiej Koptyra[15] Hayden Lau[2] Krishna Sri Ipsit Mantri[17] Ferdinand Mom[18,19]
Atsushi Saito[2,20] Guangyu Song[21] Xiangru Tang[22] Bolun Wang[23] Johan S. Wind[24]
Stanisław Woźniak[15] Ruichong Zhang[9] Zhenyuan Zhang[2] Qihang Zhao[25,26]
Peng Zhou[23] Qinghua Zhou[5] Jian Zhu[27] Rui-Jie Zhu[28,29]

[1]Generative AI Commons [2]EleutherAI [3]U. of Barcelona [4]Charm Therapeutics [5]Ohio State U. [6]U. of C., Santa Barbara
[7]Zendesk [8]Booz Allen Hamilton [9]Tsinghua University [10]Peking University [11]Storyteller.io [12]Crisis24 [13]New York U.
[14]National U. of Singapore [15]Wroclaw U. of Science and Technology [16]Databaker Technology [17]Purdue U. [18]Criteo AI Lab
[19]Epita [20]Nextremer [21]Moves [22]Yale U. [23]RuoxinTech [24]U. of Oslo [25]U. of Science and Technology of China
[26]Kuaishou Technology [27]U. of British Columbia [28]U. of C., Santa Cruz [29]U. of Electronic Science and Technology of China

Prerequisite: Recurrent Neural Network Recall



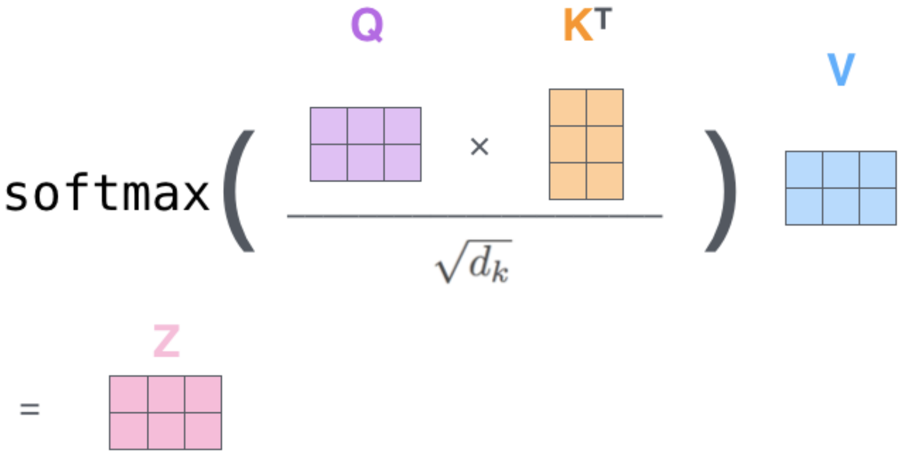
$$h_1 = \tanh(W_{hh}h_0 + W_{hx}x_1)$$

$$y_1 = \text{softmax}(W_{hy}h_1)$$

Ref: https://www.cs.rice.edu/~vo9/deep-vislang/lectures/lecture08.pdf

Prerequisite: AFT (Attention Free Transformer)

$$\mathrm{Attn}(Q,K,V) = \mathrm{softmax}(QK^\top)V,$$



decomposed as $$\mathrm{Attn}(Q,K,V)_t = \frac{\sum_{i=1}^{T} e^{q_t^\top k_i} v_i}{\sum_{i=1}^{T} e^{q_t^\top k_i}}.$$

$$\mathrm{Attn}^+(W,K,V)_t = \frac{\sum_{i=1}^{t} e^{w_{t,i}+k_i} v_i}{\sum_{i=1}^{t} e^{w_{t,i}+k_i}},$$

$\{ w_{t,i} \} \in R^{T\times T}$ is the learned pair-wise position biases, and each $w_{t,i}$ is a scalar

Ref: [1] https://jalammar.github.io/illustrated-transformer/ [2] Peng, Bo, et al. "Rwkv: Reinventing rnns for the transformer era." *arXiv preprint arXiv:2305.13048* (2023).

Prerequisite: AFT (Attention Free Transformer)

$$\mathrm{Attn}^{+}(W, K, V)_t = \frac{\sum_{i=1}^{t} e^{w_{t,i}+k_i} v_i}{\sum_{i=1}^{t} e^{w_{t,i}+k_i}},$$

$\{ w_{t,i} \} \in R^{T\times T}$ is the **learned pair-wise position biases,**
and each $w_{t,i}$ is a scalar

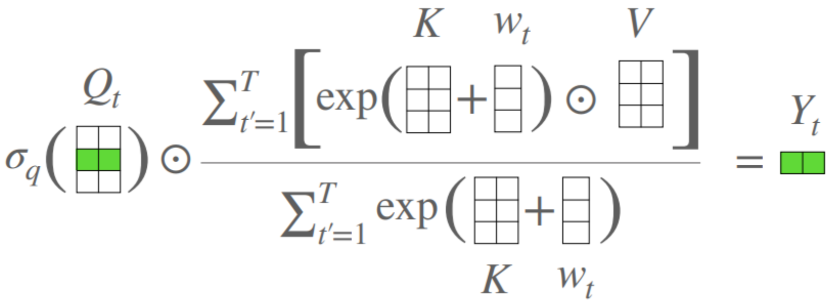
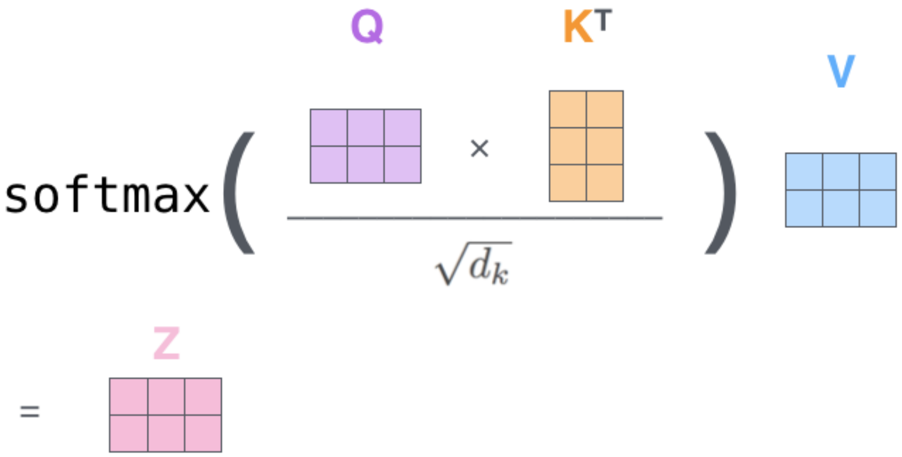


Figure 2: An illustration of AFT defined in Equation 2, with $T = 3, d = 2$.

$$\mathrm{Attn}(Q, K, V)_t = \frac{\sum_{i=1}^{T} e^{q_t^\top k_i} v_i}{\sum_{i=1}^{T} e^{q_t^\top k_i}}.$$



Ref: Zhai et al. An Attention Free Transformer

RWKV Explained

$$w_{t,i} = -(t-i)w,$$

$$\text{Attn}^+(W,K,V)_t = \frac{\sum_{i=1}^{t} e^{w_{t,i}+k_i} v_i}{\sum_{i=1}^{t} e^{w_{t,i}+k_i}},$$

where $w \in (R_{\geq 0})^d$, with d the number of channels. We require w to be non-negative to ensure that $e^{w_{t,i}} \leq 1$ and the per-channel weights decay backwards in time.

- Each $w_{t,i}$ in RWKV be a channel-wise time decay vector multiplied by the relative position. **Not a standalone learnable position embedding any more!**

RWKV Block v.s. Transformer Block

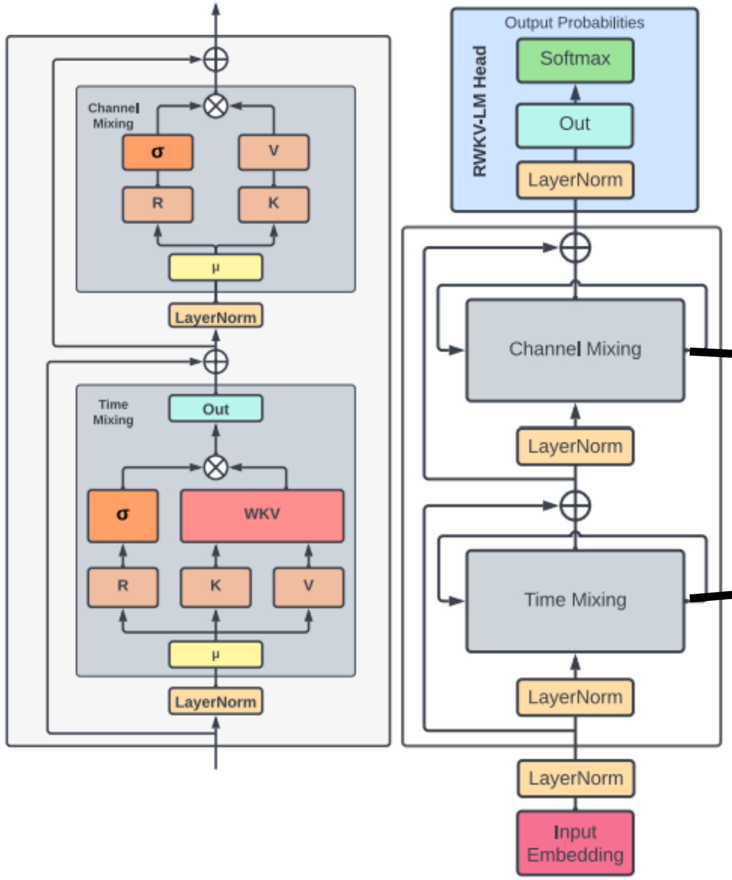


Figure 2: RWKV block elements (left) and RWKV residual block with a final head for language modeling (right) architectures.

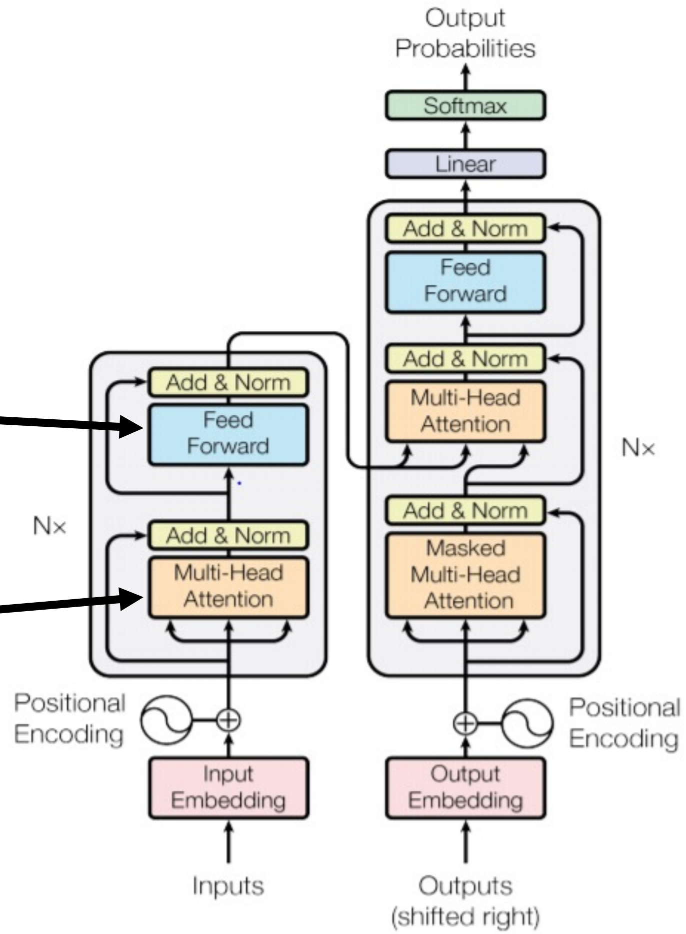


Figure 1: The Transformer - model architecture.

RWKV

- time-mixing and channel-mixing blocks
- R: **Receptance vector** acting as the acceptance of past information.
  - Sounds like what? Forget Gate!
- W: Weight is the positional weight decay vector. A trainable model parameter.
- K: Key is a vector analogous to K in traditional attention.
- V : Value is a vector analogous to V in traditional attention

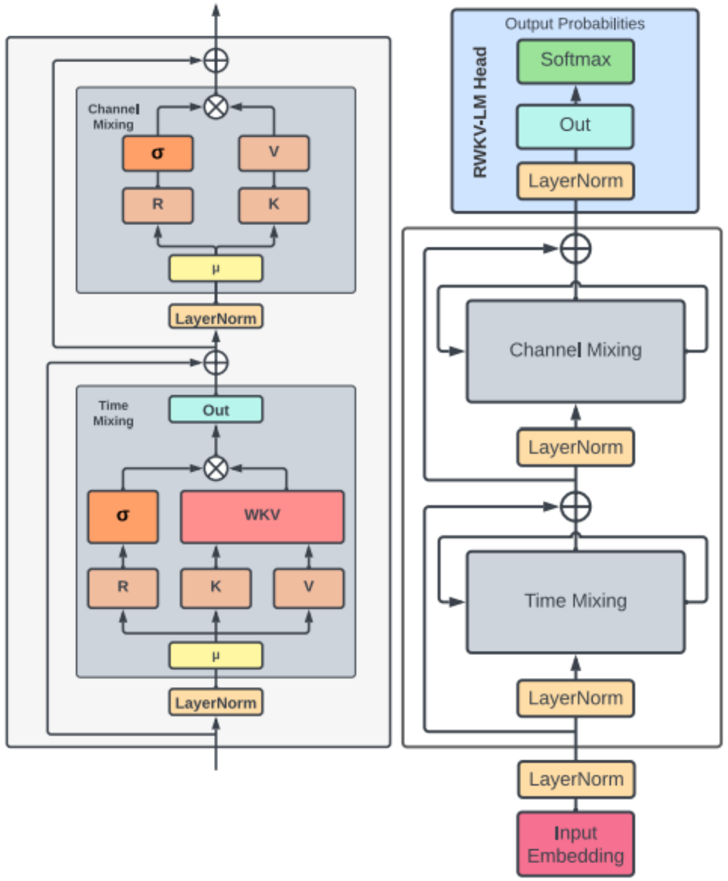


Figure 2: RWKV block elements (left) and RWKV residual block with a final head for language modeling (right) architectures.

RWKV Time-mixing Block

The time-mixing block is given by:

Token shift: only see **one step** before!

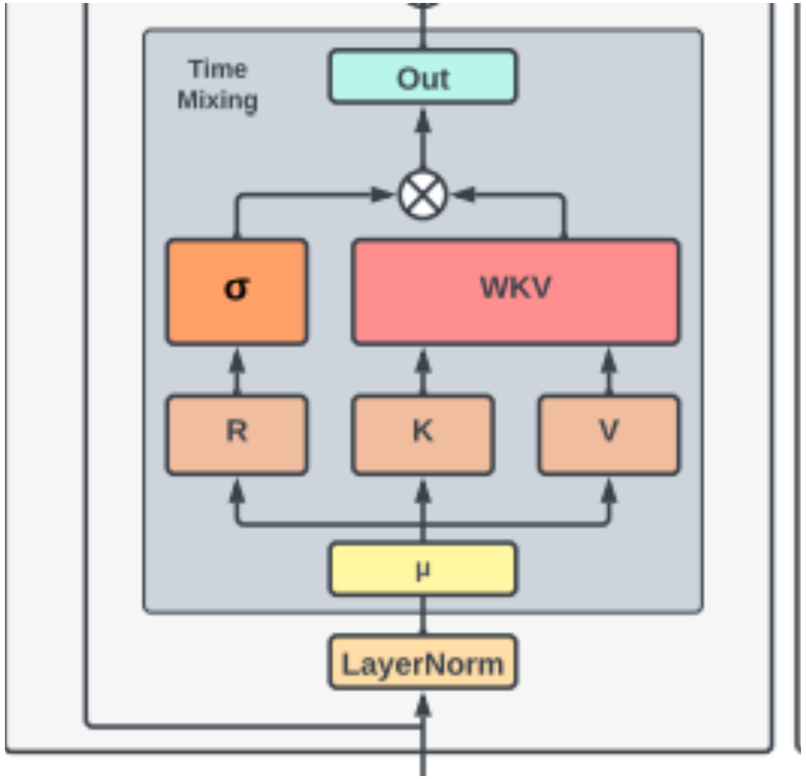
$$\boxed{\begin{aligned} r_t &= W_r \cdot (\mu_r x_t + (1-\mu_r)x_{t-1}), \quad (11)\\ k_t &= W_k \cdot (\mu_k x_t + (1-\mu_k)x_{t-1}), \quad (12)\\ v_t &= W_v \cdot (\mu_v x_t + (1-\mu_v)x_{t-1}), \quad (13)\end{aligned}}$$

$$\boxed{wkv_t = \frac{\sum_{i=1}^{t-1} e^{-(t-1-i)w+k_i} v_i + e^{u+k_t} v_t}{\sum_{i=1}^{t-1} e^{-(t-1-i)w+k_i} + e^{u+k_t}},} \quad (14)$$

AFT

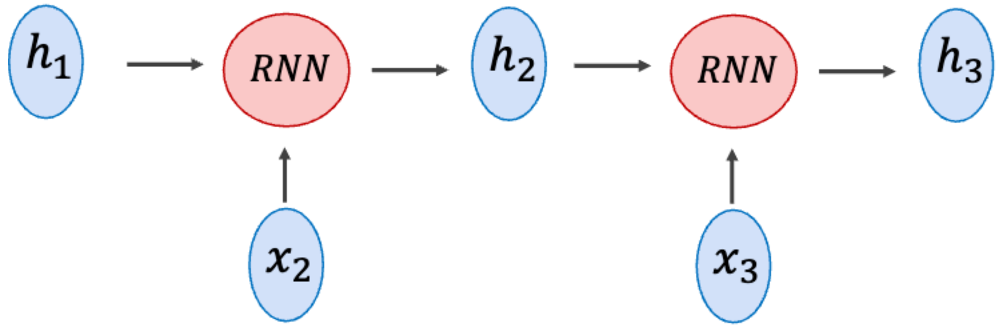
addictive operation replacing multiplication

$$o_t = W_o \cdot (\sigma(r_t) \odot wkv_t), \quad (15)$$



RWKV Parallel Training

- Let’s think: which factor prevents RNN parallel training?



  - At each timestep RNN has to conditioned on previous hidden states!

- In RWKV, this is not a problem! RWKV does not explicitly rely on hidden states, but instead uses AFT to capture context.

RWKV Parallel Training

hidden states propagation prevents efficicent training!

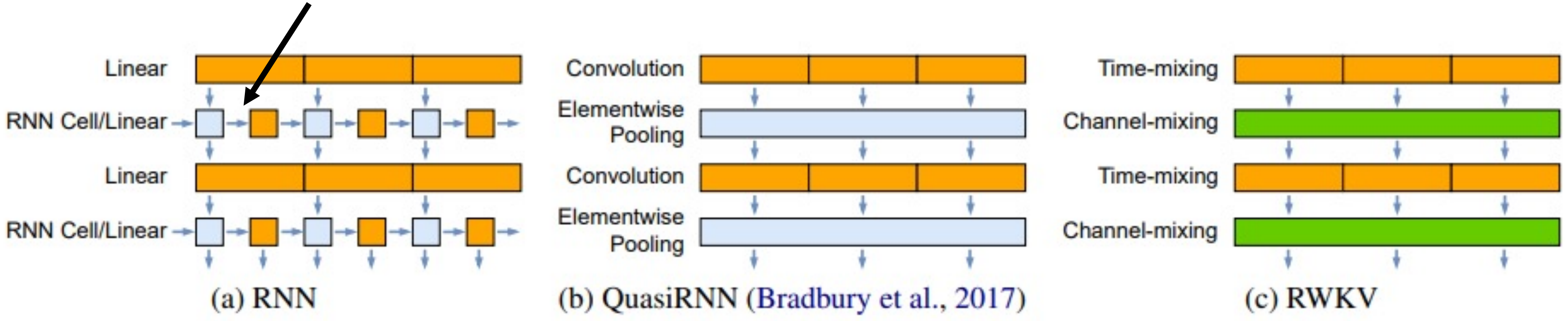
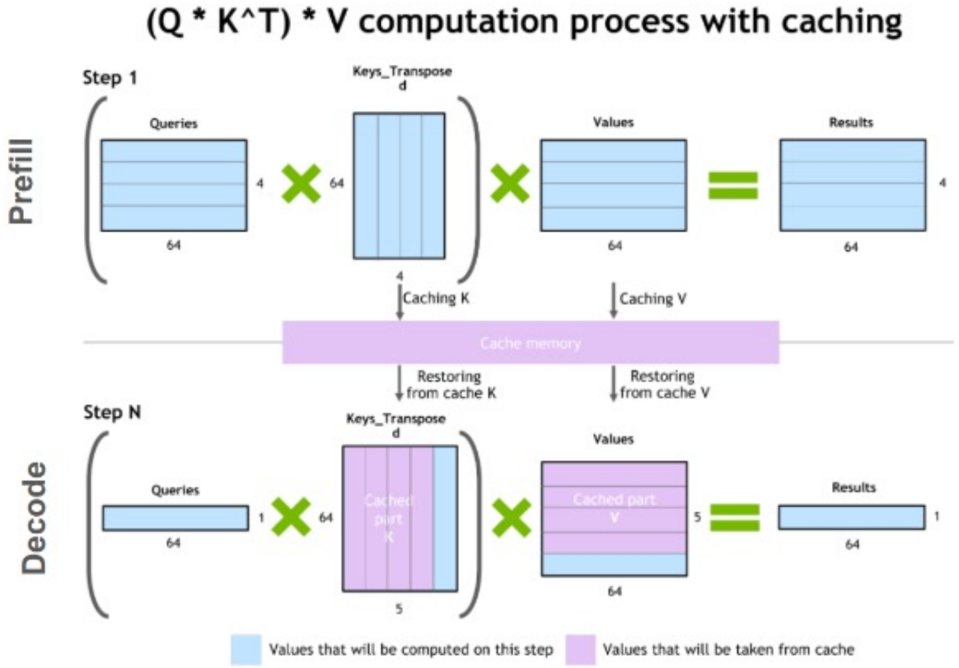


Figure 1: Computation structure of the RWKV in comparison to QRNN and RNN (Vanilla, LSTM, GRU, etc) architectures. Color codes: orange indicates time-mixing, convolutions or matrix multiplications, and the continuous block indicates that these computations can proceed simultaneously; blue signifies parameterless functions that operate concurrently along the channel or feature dimension (element-wise). Green indicates channel-mixing.

RWKV Sequential Decoding

- Question First: what's the time complexity for GPT to decode a sequence of length *n*?
  - native: $O(n^2)$;
  - with key-value caching: $O(n)$; at each timestep, we only compute intermediate activations for current position and reuse all previous key-values.



Ref: https://developer.nvidia.com/blog/mastering-llm-techniques-inference-optimization/

RWKV Sequential Decoding

- RWKV Time-mixing Block can be seen as an RNN cell (Appendix D),
  - which means a native O(n) decoder!

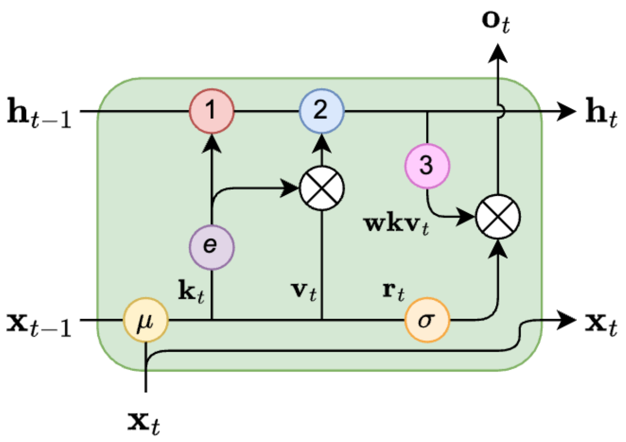


Figure 8: RWKV time-mixing block formulated as an RNN cell. Color codes: yellow (μ) denotes the token shift, red (1) denotes the denominator, blue (2) denotes the numerator, and pink (3) denotes the fraction computations in 16. h denotes the numerator-denominator tuple.

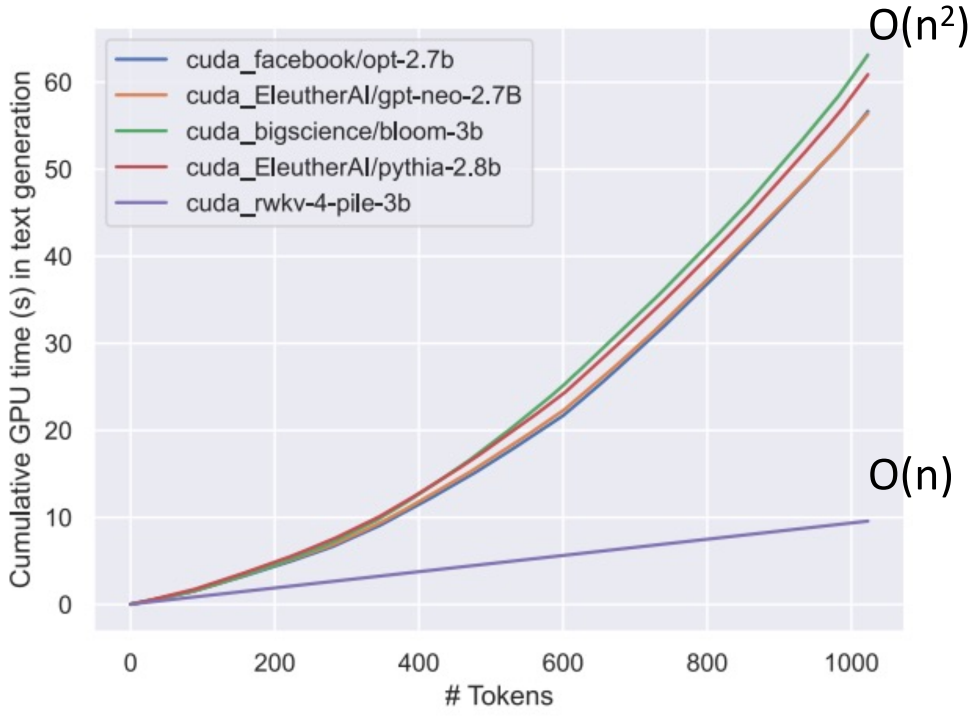
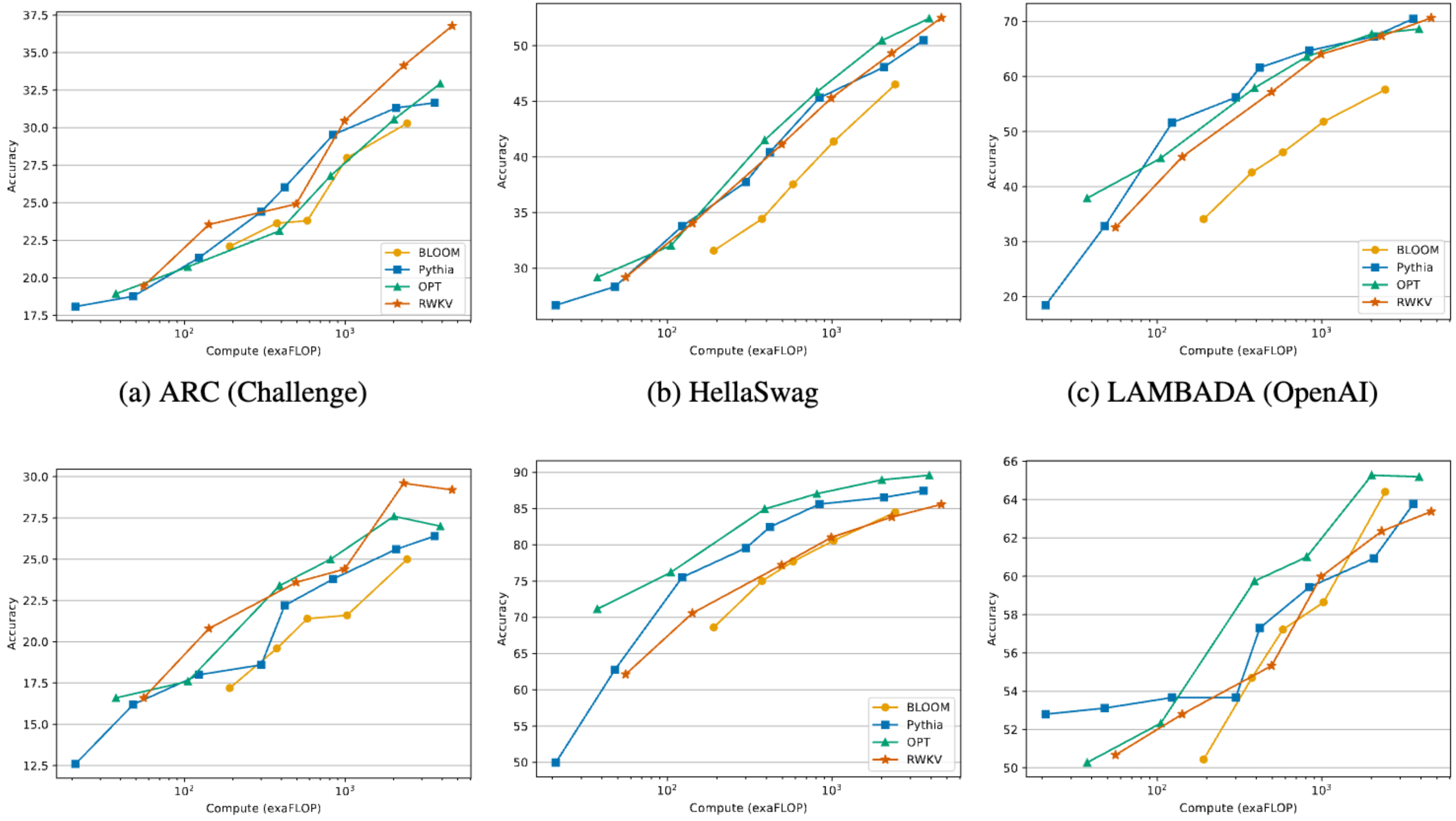


Figure 6: Cumulative time during text generation for different LLMs.

RWKV Results



(a) ARC (Challenge) (b) HellaSwag (c) LAMBADA (OpenAI)

(d) OpenBookQA (e) ReCoRD (f) Winogrande

Figure 5: Zero-Shot Performance of RWKV on common language modeling evaluation benchmarks. Additional plots can be found in Appendix J.

RWKV Long-range Arena Results

Table 4: Evaluation on Long Range Arena. Other models reported in the literature (Gu et al., 2022; Alam et al., 2023). **Bolded** values are the best.

MODEL	LISTOPS	TEXT	RETRIEVAL	IMAGE	PATHFINDER	PATH-X	AVG
Transformer	36.37	64.27	57.46	42.44	71.40	✗	53.66
Reformer	37.27	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	65.90	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	77.80	✗	54.42
Nyströmformer	37.15	65.52	79.56	41.58	70.94	✗	57.46
Luna-256	37.25	64.57	79.29	47.38	77.72	✗	59.37
Hrrformer	39.98	65.38	76.15	50.45	72.17	✗	60.83
S4	**59.60**	**86.82**	**90.90**	**88.65**	**94.20**	**96.35**	**86.09**
RWKV	55.88	86.04	88.34	70.53	58.42	✗	72.07

RWKV -> EMNLP'23 Findings

Paper Decision

Decision Program Chairs (emnlp-2023-pc@googlegroups.com, juancitomiguelito@gmail.com, juancarabina@meta.com, hbouamor@cmu.edu, +2 more)

07 Oct 2023, 14:38 (modified: 01 Dec 2023, 15:17) Everyone Revisions

Decision: Accept-Findings

Comment:

Summary of the paper: RWKV is the largest RNN model trained to date in NLP that rivals transformers in performance. The results show that the model has impressive performance, making it a worthwhile subject of further study.

This manuscript has a lot of positives. The idea presented in the paper is very ambitious and relevant to the NLP community. The proposed method has comparable training speed as compared to transformers with much faster inference and lower memory footprint.

Some of the core criticisms of the paper are on the empirical evaluations and the paper not being well written. Multiple details are missing including experiments such as actual compute time comparison, evaluation beyond 4K token length to showcase the use of RNN style method. RNN style methods trade-off accuracy and compute time (because of information bottleneck), an evaluation of this trade-off would be an interesting addition.

Another recurrent model: Mamba

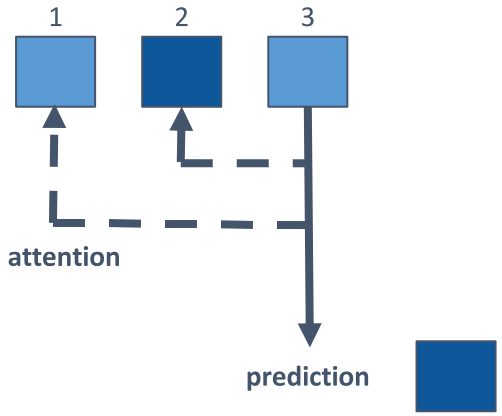
Mamba: Linear-Time Sequence Modeling with Selective State Spaces

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

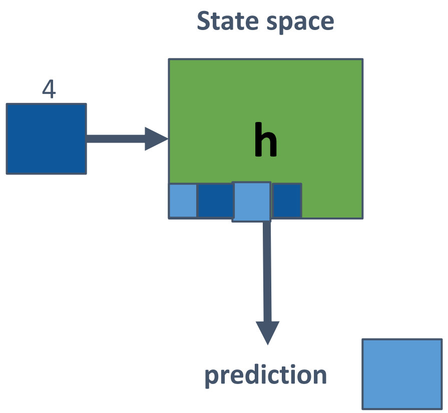
[1]Machine Learning Department, Carnegie Mellon University
[2]Department of Computer Science, Princeton University
agu@cs.cmu.edu, tri@tridao.me

- Mamba is motivated by “Selective State Space Model” (S4), another RNN variant.
- Encourage you to read it after class. We will introduce it at an intuitive level.

Structured State-Space Sequence (S4): Intuitive Understanding



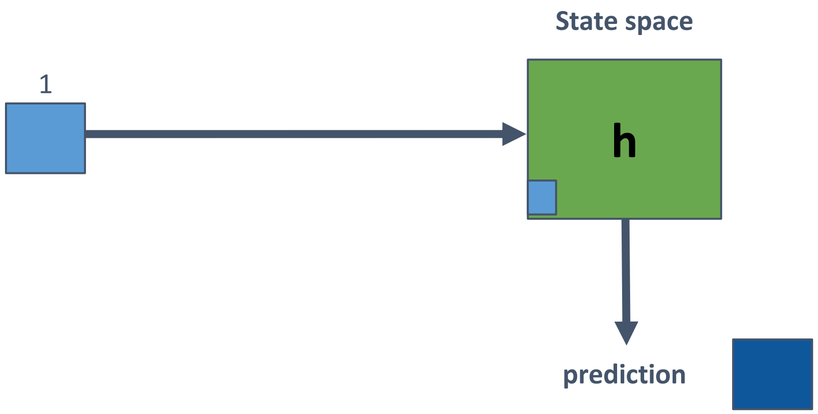
Transformer Attention

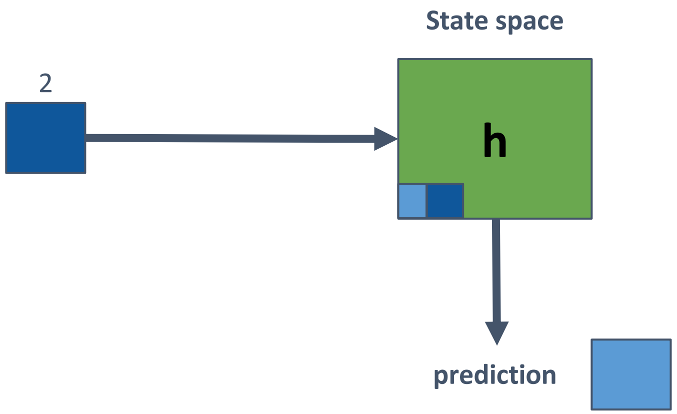


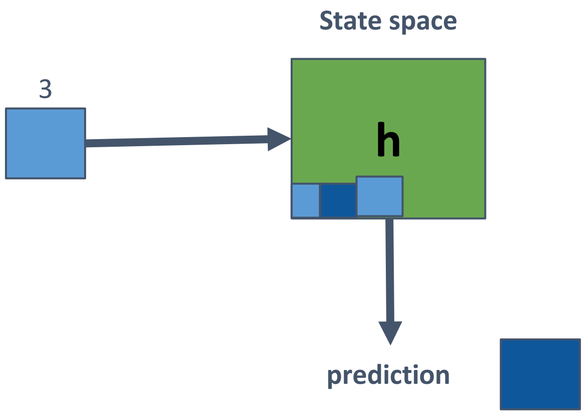
S4 Model

Ref: https://www.youtube.com/watch?v=iskuX3Ak9Uk

Structured State-Space Sequence (S4): Intuitive Understanding





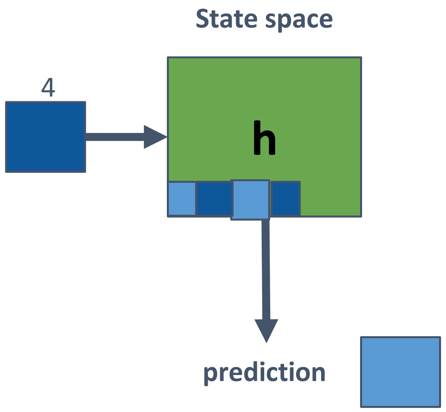


$$\Delta h = f(h, x) \; = \delta(Ah + Bx)$$

$$h_2 = h_1 + \delta(Ah_1 + Bx_1)$$

$$x_2 = Ch_2 \cdot x_1$$

Mamba (S4 + Selective) Algorithms



Algorithm 1 SSM (S4)

Input: $x : (\mathtt{B}, \mathtt{L}, \mathtt{D})$
Output: $y : (\mathtt{B}, \mathtt{L}, \mathtt{D})$

$\boldsymbol{A} : (\mathtt{D}, \mathtt{N}) \leftarrow \mathsf{Parameter}$
▷ Represents structured $N \times N$ matrix
$\boldsymbol{B} : (\mathtt{D}, \mathtt{N}) \leftarrow \mathsf{Parameter}$
$\boldsymbol{C} : (\mathtt{D}, \mathtt{N}) \leftarrow \mathsf{Parameter}$
$\Delta : (\mathtt{D}) \leftarrow \tau_\Delta(\mathsf{Parameter})$
$\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}} : (\mathtt{D}, \mathtt{N}) \leftarrow \mathsf{discretize}(\Delta, \boldsymbol{A}, \boldsymbol{B})$
$y \leftarrow \mathsf{SSM}(\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}}, \boldsymbol{C})(x)$
▷ Time-invariant: recurrence or convolution
return y

Algorithm 2 SSM + Selection (S6)

Input: $x : (\mathtt{B}, \mathtt{L}, \mathtt{D})$
Output: $y : (\mathtt{B}, \mathtt{L}, \mathtt{D})$

$\boldsymbol{A} : (\mathtt{D}, \mathtt{N}) \leftarrow \mathsf{Parameter}$
▷ Represents structured $N \times N$ matrix
$\boldsymbol{B} : (\mathtt{B}, \mathtt{L}, \mathtt{N}) \leftarrow \boxed{s_B(x)}$
$\boldsymbol{C} : (\mathtt{B}, \mathtt{L}, \mathtt{N}) \leftarrow \boxed{s_C(x)}$
$\Delta : (\mathtt{B}, \mathtt{L}, \mathtt{D}) \leftarrow \tau_\Delta(\mathsf{Parameter} \boxed{+ s_\Delta(x))}$
$\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}} : (\mathtt{B}, \mathtt{L}, \mathtt{D}, \mathtt{N}) \leftarrow \mathsf{discretize}(\Delta, \boldsymbol{A}, \boldsymbol{B})$
$y \leftarrow \mathsf{SSM}(\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}}, \boldsymbol{C})(x)$
▷ Time-varying: recurrence (*scan*) only
return y

- Mamba improves S4 by:
  - Now B, C and delta is dependent on current time step input x.
  - B -> B(x); C-> C(x); delta -> delta(x)
  - During this process, the model selectively chooses which part of hidden states to use depending on current input.

Mamba and its extension

- Researchers have been migrating mamba into various domains.

(Arxiv 23.12.01) Mamba: Linear-Time Sequence Modeling with Selective State Spaces Paper Code Stars 9k

(Arxiv 24.01.08) MoE-Mamba: Efficient Selective State Space Models with Mixture of Experts Paper

(Arxiv 24.01.24) MambaByte: Token-free Selective State Space Model Paper Code Stars 591

(Arxiv 24.01.31) LOCOST: State-Space Models for Long Document Abstractive Summarization Paper Code Stars 12

(Arxiv 24.02.01) BlackMamba: Mixture of Experts for State-Space Models Paper Code Stars 188

(Arxiv 24.02.06) Can Mamba Learn How to Learn? A Comparative Study on In-Context Learning Tasks Paper

(Arxiv 24.02.08) Mamba-ND: Selective State Space Modeling for Multi-Dimensional Data Paper

(Arxiv 24.02.15) Hierarchical State Space Models for Continuous Sequence-to-Sequence Modeling Paper Code Stars 23

(Arxiv 24.02.19) Pan-Mamba: Effective pan-sharpening with State Space Model Paper Code Stars 39

(Arxiv 24.02.23) State Space Models for Event Cameras Paper

Ref: https://github.com/yyyujintang/Awesome-Mamba-Papers

Is recurrent model the future arch of LLM?

- Yes and No!
- Yes: the community is continuously contributing works towards recurrent model, and some of them have amazing designs! Another GPT might hide in them.

S4 [Gu et al., 2022a]
DSS [Gupta, 2022]
GSS [Mehta et al., 2022]
S4D [Gu et al., 2022b]
H3 [Dao et al., 2022]
S5 [Smith et al., 2022]
BiGS [Wang et al., 2022]

QRNN [Bradbury et al., 2016]
Mega [Ma et al., 2022]
SGConv [Li et al., 2022]
Hyena [Poli et al., 2023]
LRU [Orvieto et al., 2023]
RWKV [Peng et al., 2023]
MultiRes [Shi et al., 2023]

Is recurrent model the future arch of LLM?

- Yes and No!
- No: in standard evaluation setting (no long-context ability needed), they are unable to match their transformer counterpart with similar size and FLOPs.

Lossless long-context is everything

- *If you had a context length of 1 billion tokens, none of the problems you see today would be problems.*
    - Zhiling Yang, Author of Transformer-XL, Founder of Moonshot AI. Raised $1B in Series B in Feb. 2024.
- The next generation of LLM should have:
    - scalability
    - generalization ability
- Is it still with Transformer-like block? Is it still trained with next token prediction loss?
    - We don’t know.

Ref: https://foresightnews.pro/article/detail/53994

Questions?