GOSIM

GOSIM AI Paris 2025

Linear Next: The Evolution of LLM Architecture

Yiran Zhong

Senior Research Director

MiniMax

① The History of LLMs

② Long Sequence Modeling

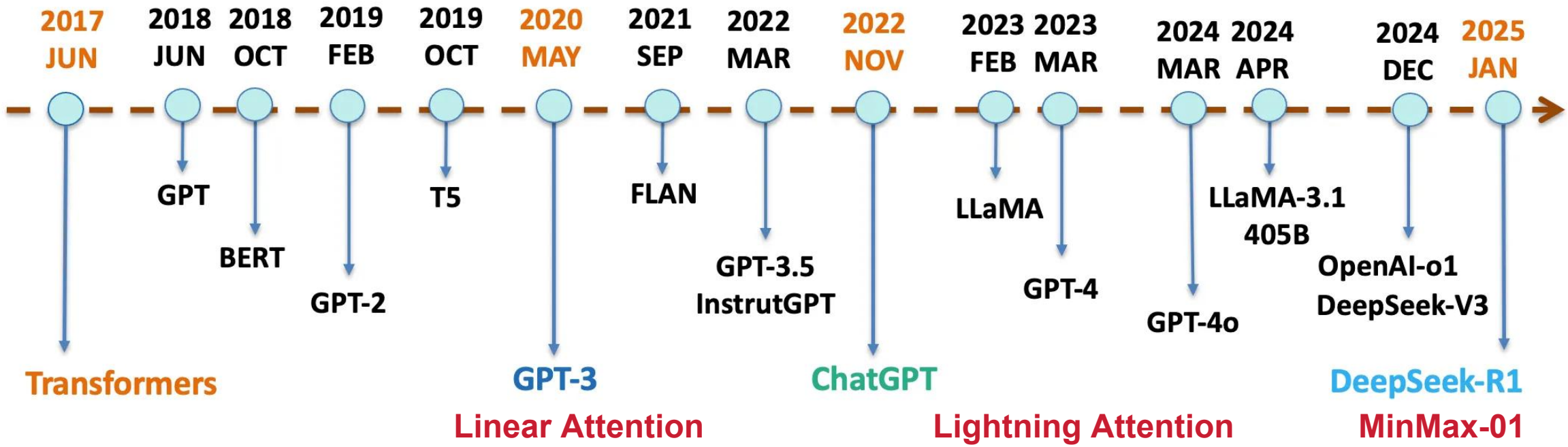
- Sparse Attention
- Linear Attention
- Mixture of Experts

③ MiniMax-01

④ The Benchmark – Linear Next

GOSIM

A Brief History of LLMs



GOSIM

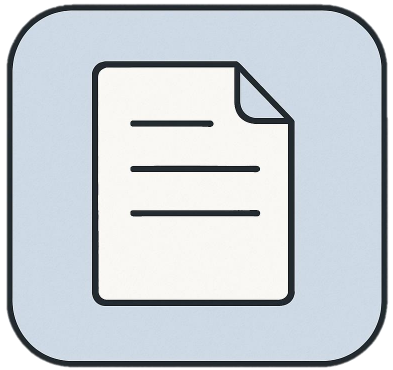
GOSIM AI Paris 2025

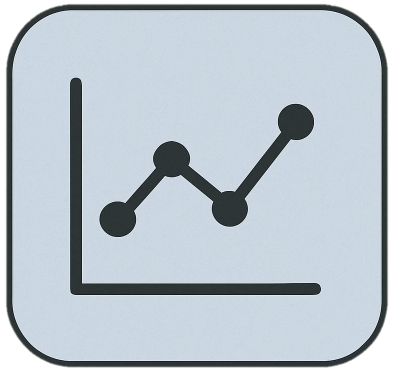
Long Sequence Modeling

GOSIM

What is Long Context Sequence Modeling?

The process of handling and understanding sequence data with long range dependencies, such as text, audio, video, and time series.



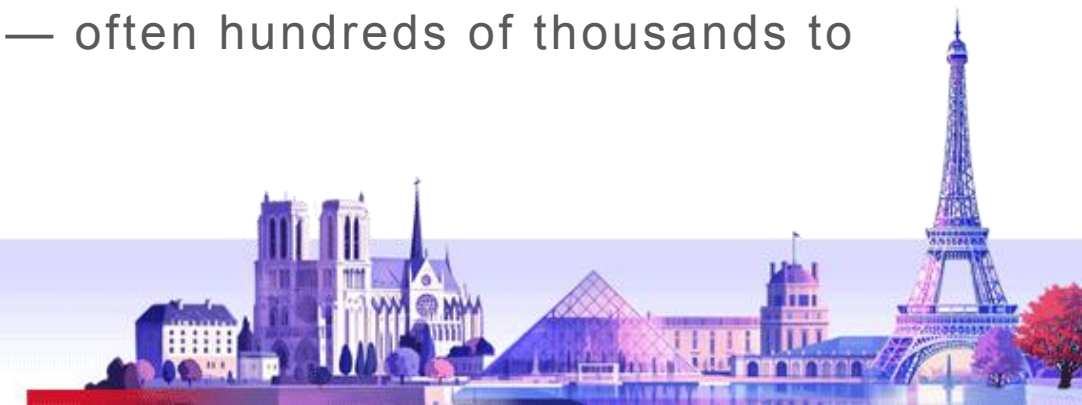


Why it matters?

Key Capabilities for Future AI Needed:

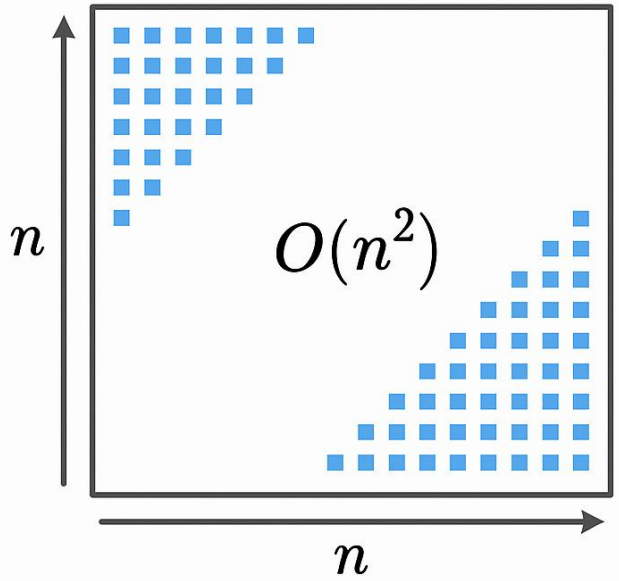
- Deep Research
- Computer Use
- RAG (Retrieval-Augmented Generation)
- Codebase-Level Reasoning

All of these rely on processing long-range context — often hundreds of thousands to millions of tokens.



The Challenges

The computational complexity of softmax attention is $O(n^2)$, n is the sequence length.



Long context data is scarce in public corpora.

Preliminary

- Key Components in Modern Sequence Modeling
  - Token mixing: Interactions on sequence dimensions, i.e., Attention ($O(n^2)$).
  - Channel mixing: Interactions on feature dimensions, i.e., FFN ($O(n)$).

MiniMax-01 int8 H20						
Context length	**Linear attn latency**	**Softmax attn latency**	**Linear attn latency proportion per layer**	**Softmax attn latency proportion per layer**	**Linear attn latency proportion to all**	**Softmax attn latency proportion to all**
8000	11.312	10.913	6%	22%	5%	3%
128000	11.181	36.024	6%	76%	4%	24%
1024000	11.808	228.602	6%	96%	2%	71%

MetaFormer Is Actually What You Need for Vision, CVPR 2022

Solution Overview

GOSIM

Token Mixing

Sparse Attention

Linear Attention

Channel Mixing

MoE

Solution Overview

GOSIM

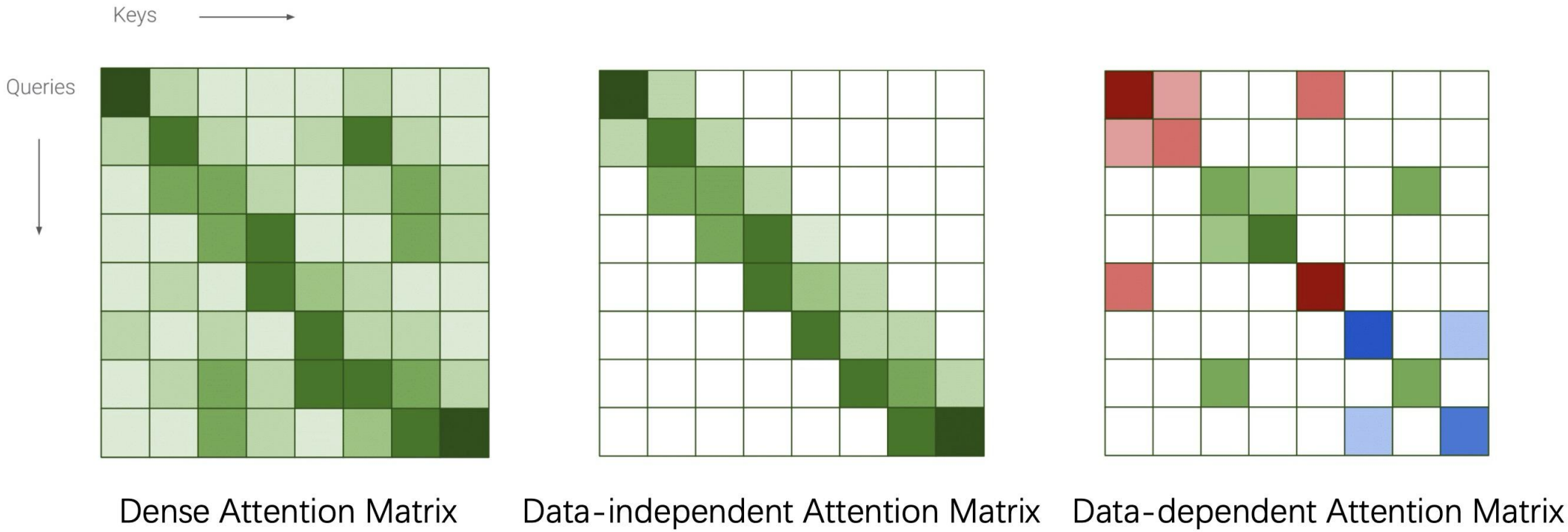
Sparse Attention

Linear Attention

MoE

Sparse Attention

- Data-independent Sparse: Sliding-window attention, Big Bird, Sparse Transformer
- Data-dependent Sparse: Routing Transformer, Reformer, MoBA, NSA



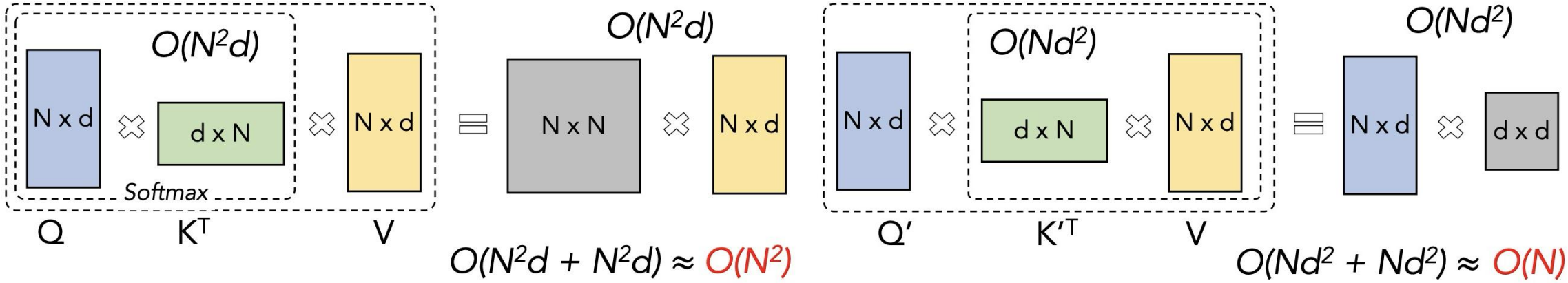
Dense Attention Matrix Data-independent Attention Matrix Data-dependent Attention Matrix

Linear Attention

Replace the standard softmax attention with a **kernelized approximation** that allows decomposition:

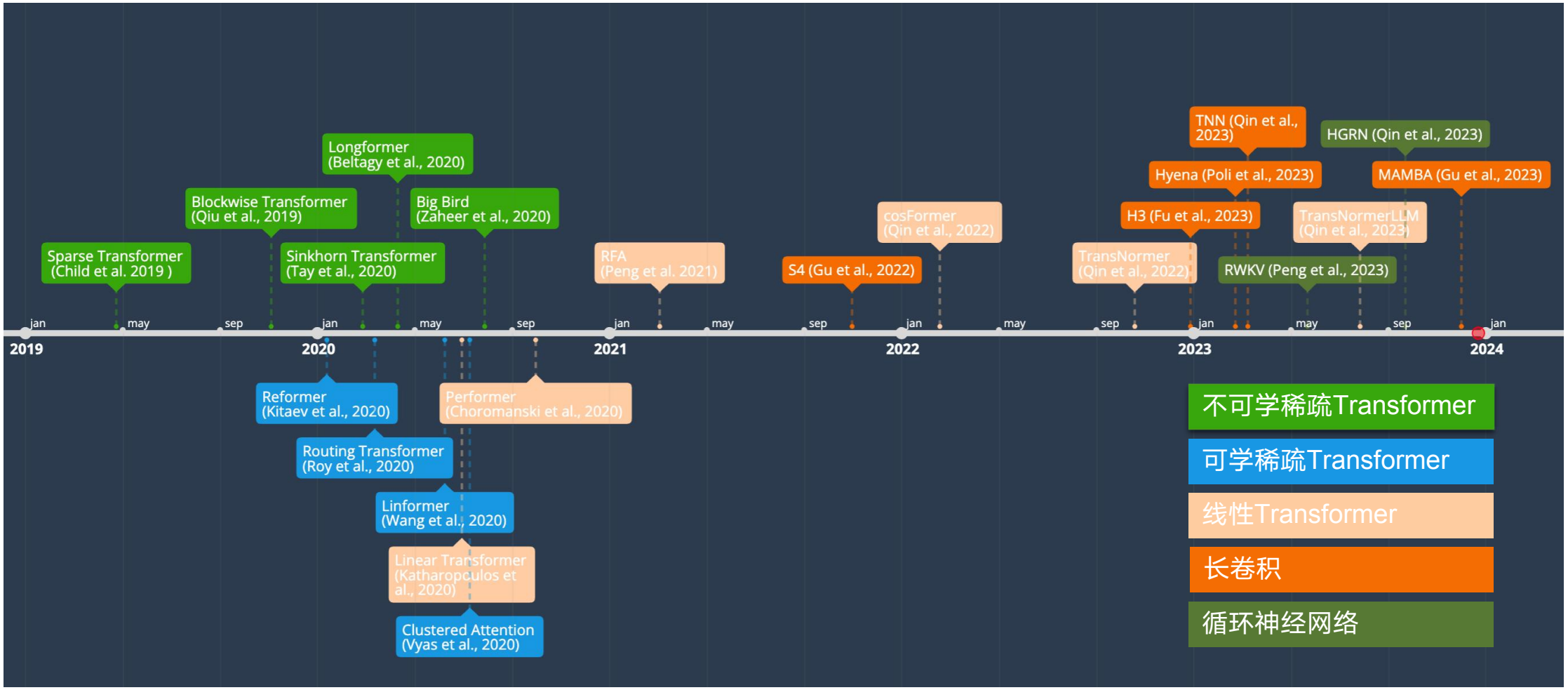
$$\text{Attention}(Q, K, V) = \phi(Q) \cdot (\phi(K)^T \cdot V)$$

- Here, ϕ is a **feature map** (e.g., random projection, cosine kernel)
- Transforms quadratic matrix product into **two linear passes**



Efficient Attention Timeline

GOSIM



Limitations of Current Linear Attention

- Inferior Performance
  - Significant performance gap compared to state-of-the-art softmax attention in language modeling tasks.
- Slow Training Speed
  - Cumulative summation (cumsum) operations slow down training.
  - Many models fallback to standard attention during practical use.

Solution: Lightning Attention

Eliminate cumsum

from *element-wise* to *chunk-wise* computation

$$\mathbf{O} = [(\mathbf{Q}\mathbf{K}^\top) \odot \mathbf{M}]\mathbf{V} \tag{3}$$

We use a tiling technique to compute linear attention in a causal setting. Specifically, we first divide $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ into two blocks by rows:

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{bmatrix}, \mathbf{X}_1 \in \mathbb{R}^{m \times d}, \mathbf{X}_2 \in \mathbb{R}^{(n-m) \times d}, \mathbf{X} \in \{\mathbf{Q}, \mathbf{K}, \mathbf{V}\}.$$

Then, by unfolding Eq. 3, we get (note that $\mathbf{kv}_0 = 0$):

$$\begin{aligned} \mathbf{kv}_s &= \mathbf{kv}_0 + \sum_{j=1}^{s} \mathbf{k}_j \mathbf{v}_j^\top, s = 1, \ldots, m. \\ \mathbf{o}_s^\top &= \mathbf{q}_s^\top \mathbf{kv}_s = \mathbf{q}_s^\top \mathbf{kv}_0 + \mathbf{q}_s^\top \sum_{j=1}^{s} \mathbf{k}_j \mathbf{v}_j^\top. \end{aligned} \tag{5}$$

In block form, we have:

$$\begin{aligned} \mathbf{O}_1 &= \mathbf{Q}_1 \mathbf{kv}_0 + [(\mathbf{Q}_1 \mathbf{K}_1^\top) \odot \mathbf{M}]\mathbf{V}_1 \\ &\triangleq \mathbf{Q}_1 \mathbf{KV}_0 + [(\mathbf{Q}_1 \mathbf{K}_1^\top) \odot \mathbf{M}]\mathbf{V}_1. \end{aligned} \tag{6}$$

The above formula shows that the forward causal linear attention can be divided into two parts:

- The computation within the block $[(\mathbf{Q}_1 \mathbf{K}_1^\top) \odot \mathbf{M}]\mathbf{V}_1$ (intra blocks) can use the Left Product;
- The computation between blocks $\mathbf{Q}_1 \mathbf{KV}_0$ (inter blocks) can use the Right Product.

It is worth noting that the second block can be computed using the same idea as follows:

$$\begin{aligned} \mathbf{kv}_{m+t} &= \mathbf{kv}_m + \sum_{j=m+1}^{m+t} \mathbf{k}_j \mathbf{v}_j^\top, t = 1, \ldots, n-m, \\ \mathbf{o}_{m+t}^\top &= \mathbf{q}_{m+t}^\top \mathbf{kv}_{m+t}, \\ \mathbf{O}_2 &= \mathbf{Q}_2 \mathbf{kv}_m + [(\mathbf{Q}_2 \mathbf{K}_2^\top) \odot \mathbf{M}]\mathbf{V}_2 \\ &\triangleq \mathbf{Q}_2 \mathbf{KV}_1 + [(\mathbf{Q}_2 \mathbf{K}_2^\top) \odot \mathbf{M}]\mathbf{V}_2. \end{aligned} \tag{7}$$

Note that to compute the second block, we have to use $\mathbf{KV}_1 = \mathbf{kv}_m$, which can be computed by:

$$\mathbf{KV}_1 = \mathbf{KV}_0 + \sum_{j=1}^{m} \mathbf{k}_m \mathbf{v}_m^\top = \mathbf{KV}_0 + \mathbf{K}_1^\top \mathbf{V}_1. \tag{8}$$

where $\mathbf{KV}_0 = \mathbf{kv}_0$. By using the above strategy to divide the matrix into multiple blocks, we obtain the Lightning Attention Forward Pass.

Theorem 3.1. The time complexity of Lightning Attention is $O(nd^2 + nBd)$[1].

Decay Matters

Linear attention (Katharopoulos et al. 2020):

$$\text{Parallel training}: \quad \mathbf{O} = \cancel{\text{softmax}}(\mathbf{Q}\mathbf{K}^\top \odot \mathbf{M})\mathbf{V} \quad \in \mathbb{R}^{L\times d}$$

$$\text{Iterative inference}: \quad \mathbf{o_t} = \sum_{j=1}^{t} \frac{\cancel{\exp}(\mathbf{q}_t^\top \mathbf{k}_j)}{\cancel{\sum_{l=1}^{t} \exp(\mathbf{q}_t^\top \mathbf{k}_l)}} \mathbf{v}_j \quad \in \mathbb{R}^{d}$$

where $\mathbf{M}$ is the causal mask for linear attention:

$$\mathbf{M}_{i,j} = \begin{cases} 0 & \text{if } j > i \\ 1 & \text{if } j \le i \end{cases}$$

$$\mathbf{o_t} = \sum_{j=1}^{t} (\mathbf{q}_t^\top \mathbf{k}_j)\mathbf{v}_j$$

$$= \sum_{j=1}^{t} \mathbf{v}_j(\mathbf{k}_j^\top \mathbf{q}_t) \quad \mathbf{k}_j^\top \mathbf{q}_t = \mathbf{q}_t^\top \mathbf{k}_j \in \mathbb{R}$$

$$= \underbrace{(\sum_{j=1}^{t} \mathbf{v}_j \mathbf{k}_j^\top)}_{\mathbf{S}_t \in \mathbb{R}^{d\times d}} \mathbf{q}_t \quad \text{By associativity}$$

Let $\mathbf{S}_t = \sum_{j=1}^{t} \mathbf{v}_j \mathbf{k}_j^\top \in \mathbb{R}^{d\times d}$ be the matrix-valued hidden state, then:

$$\mathbf{S}_t = \mathbf{S}_{t-1} + \mathbf{v}_t \mathbf{k}_t^\top \quad \in \mathbb{R}^{d\times d}$$

$$\mathbf{o}_t = \mathbf{S}_t \mathbf{q}_t \quad \in \mathbb{R}^{d}$$

Copied from Songlin Yang's Blog

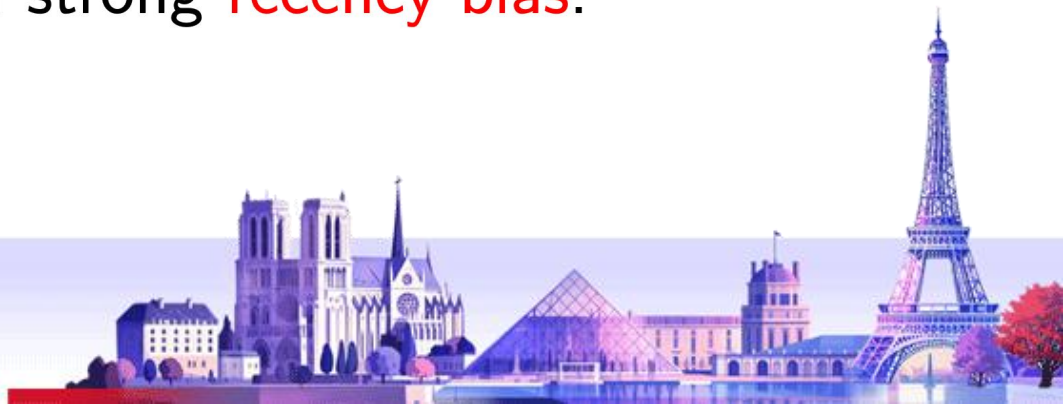
Data-independent Decay

$$\mathbf{S}_t = \gamma \mathbf{S}_{t-1} + \mathbf{v}_t \mathbf{k}_t^\top \quad \in \mathbb{R}^{d \times d}$$

$$\mathbf{o}_t = \mathbf{S}_t \mathbf{q}_t \quad \in \mathbb{R}^{d}$$

- γ is a constant exponential decay factor $0 < \gamma < 1$.
- This mechanism weighs recent tokens more than distant tokens, and language modeling has a strong recency bias.

Copied from Songlin Yang's Blog



Data-dependent Decay

$$\mathbf{S}_t = \gamma_t \mathbf{S}_{t-1} + \mathbf{v}_t \mathbf{k}_t^\top \quad \in \mathbb{R}^{d \times d}$$

$$\mathbf{o}_t = \mathbf{S}_t \mathbf{q}_t \quad \in \mathbb{R}^{d}$$

- $\gamma_t \in (0, 1)$ is a data-dependent decay term that is a function of $\mathbf{x}_t$.
- Enables dynamic control of memory retention/forgetting based on input data.
- Examples: Mamba2 (Dao and Gu 2024), mLSTM (Beck et al. 2024), Gated Retention (Sun et al. 2024b).

Fine-grained Data-dependent Decay

$$\mathbf{S}_t = \mathbf{G}_t \odot \mathbf{S}_{t-1} + \mathbf{v}_t \mathbf{k}_t^\top \qquad \in \mathbb{R}^{d \times d}$$

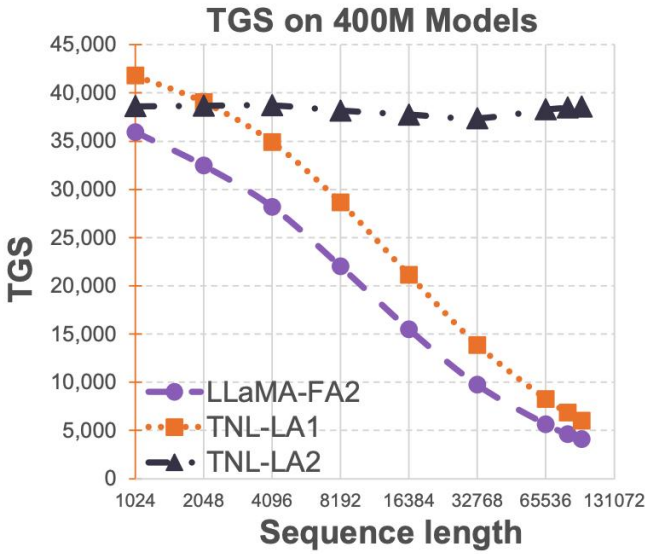
$$\mathbf{o}_t = \mathbf{S}_t \mathbf{q}_t \qquad \in \mathbb{R}^{d}$$

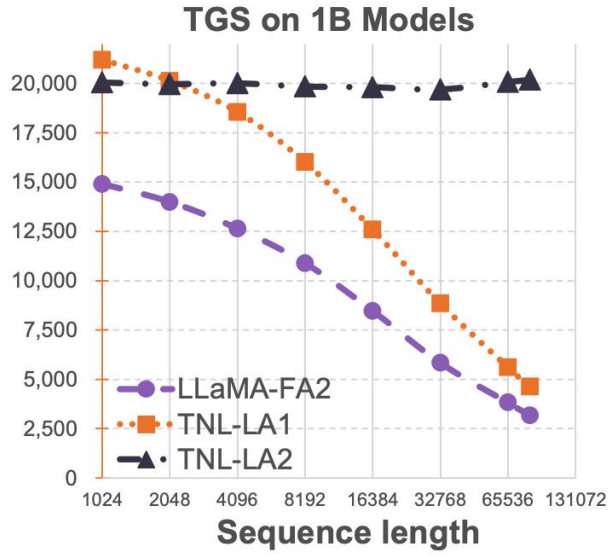
Condition for the matmul-based (chunkwise) parallel form (Yang et al. 2023):

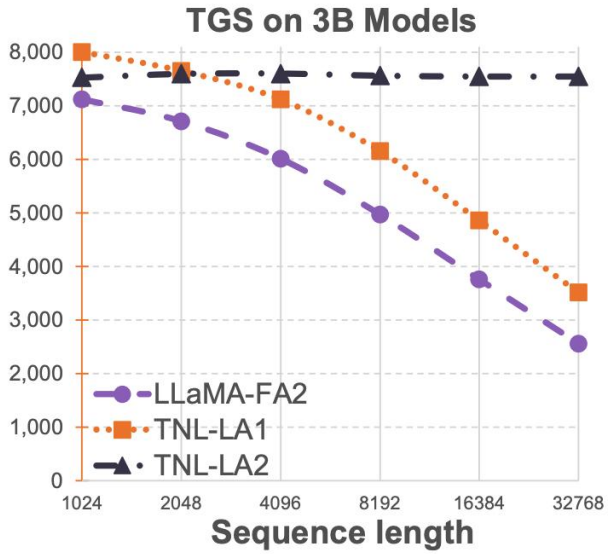
$$\mathbf{G}_t = \boldsymbol{\beta}_t \boldsymbol{\alpha}_t^\top \in \mathbb{R}^{d \times d}, \quad \boldsymbol{\alpha}_t, \boldsymbol{\beta}_t \in \mathbb{R}^{d}$$

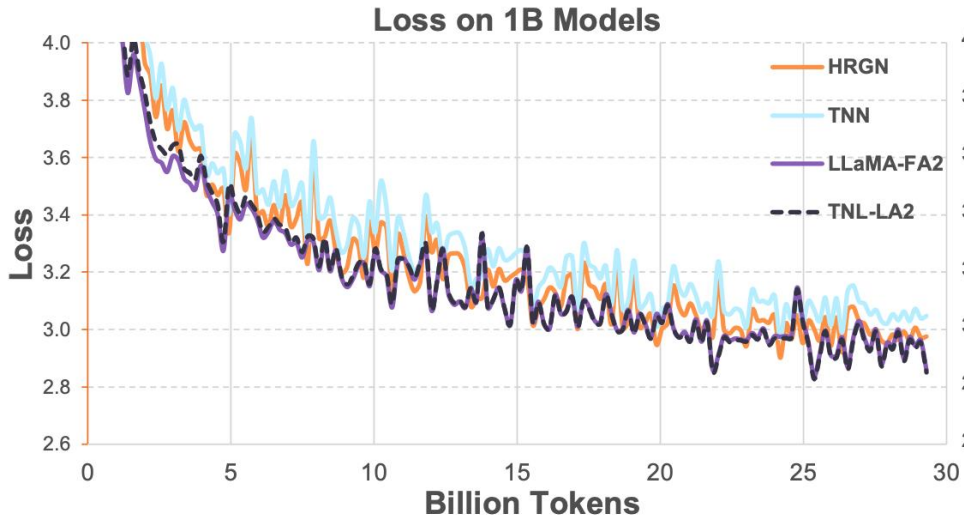
- Mamba-1 (Gu and Dao 2023) does not conform to this structure, preventing the use of tensor cores.
- It is common to set $\boldsymbol{\beta}_t = \mathbf{1}$ in practice, examples: GLA (Yang et al. 2023), RWKV6 (Peng et al. 2024), GSA (Zhang et al. 2024), HGRN2 (Qin et al. 2024a).

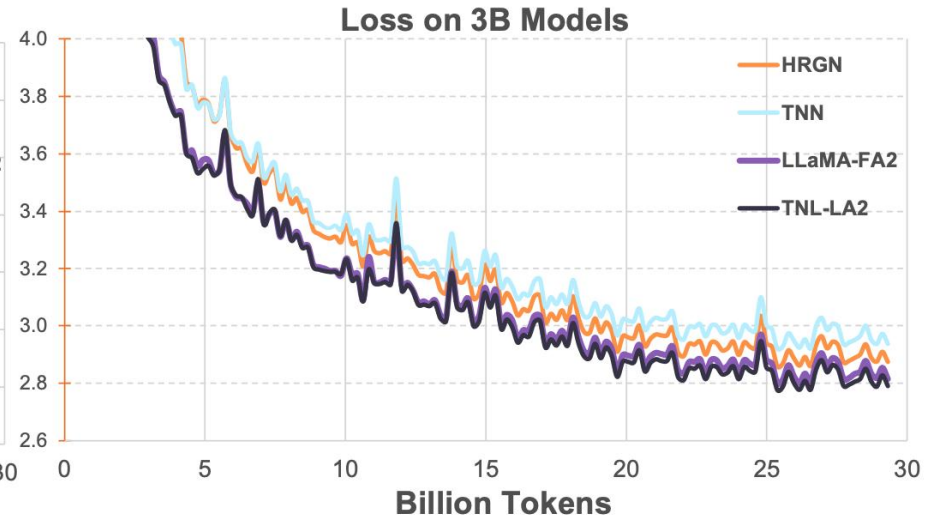
Performance











Modern Linear Attention

GOSIM

Change the update rule

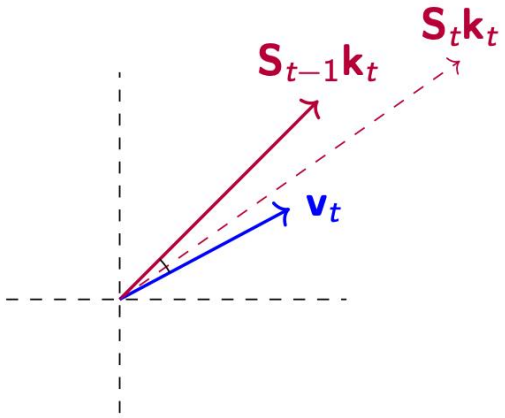
Hebbian update rule: $\mathbf{S}_t = \mathbf{S}_{t-1} + \mathbf{v}_t \mathbf{k}_t^\top$

Delta rule: $\mathbf{S}_t = \mathbf{S}_{t-1} - \beta_t \left(\mathbf{S}_{t-1}\mathbf{k}_t - \mathbf{v}_t\right) \mathbf{k}_t^\top$

Both Hebbian and delta update rules can be regarded as optimizing online learning objective via test-time SGD.

Hebbian Rule

Maximize alignment
= Minimize angle difference + enlarge $|\mathbf{Sk}|$



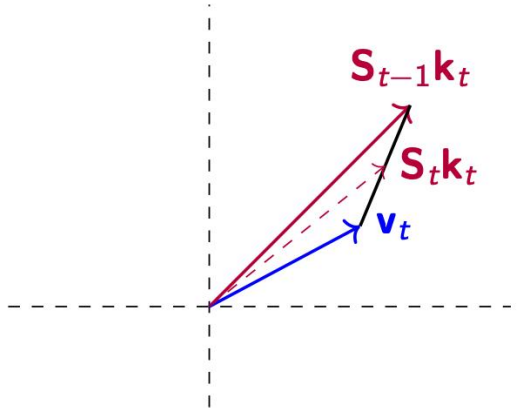
Objective: $\mathcal{L}_t(\mathbf{S}) = -\langle \mathbf{S}\mathbf{k}_t, \mathbf{v}_t \rangle$

SGD update: $\mathbf{S}_t = \mathbf{S}_{t-1} - \beta_t \nabla \mathcal{L}_t(\mathbf{S}_{t-1}) = \mathbf{S}_{t-1} + \beta_t \mathbf{v}_t \mathbf{k}_t^\top$

- Linear attention may favor increasing $|\mathbf{Sk}|$ over angle alignment, leading to numerical instabilities.
- Mamba2 uses decay ($\mathbf{S}_t = \alpha_t \mathbf{S}_{t-1} + \mathbf{v}_t \mathbf{k}_t^\top$) to control $|\mathbf{Sk}|$, stabilizing the training process.

Delta Rule

Directly minimize Euclidean distance



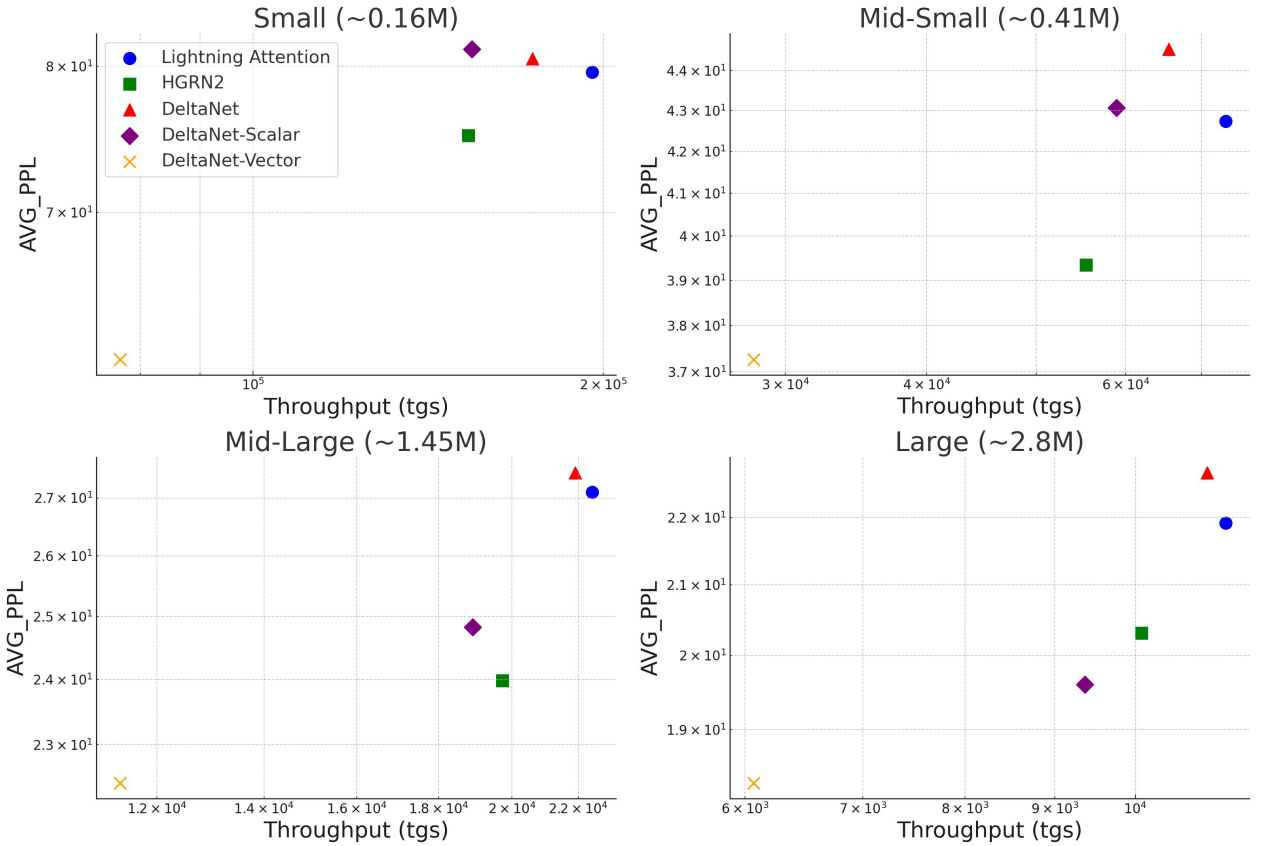
Objective: $\mathcal{L}_t(\mathbf{S}) = \frac{1}{2}\|\mathbf{S}\mathbf{k}_t - \mathbf{v}_t\|^2$

SGD update: $\mathbf{S}_t = \mathbf{S}_{t-1} - \beta_t \nabla \mathcal{L}_t(\mathbf{S}_{t-1}) = \mathbf{S}_{t-1} - \beta_t(\mathbf{S}_{t-1}\mathbf{k}_t - \mathbf{v}_t)\mathbf{k}_t^\top$

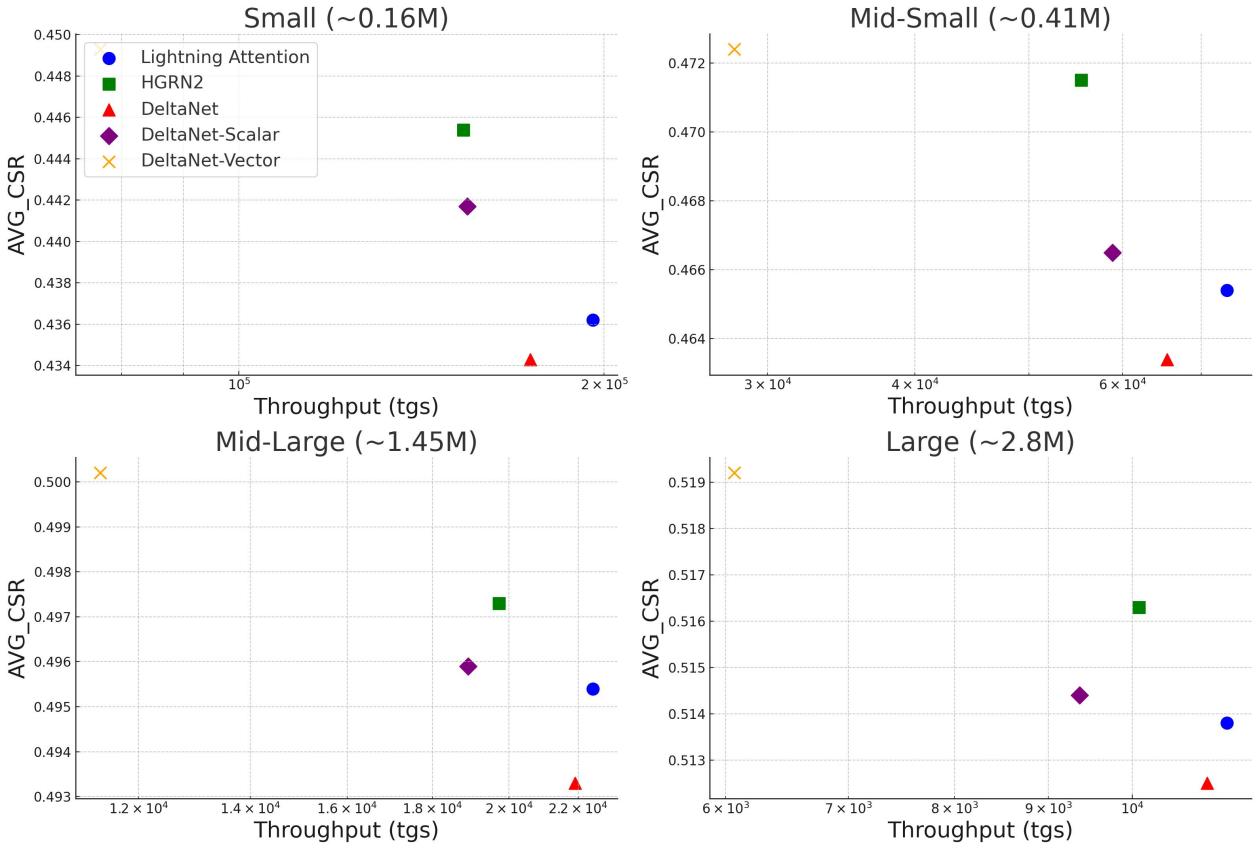
- Better numerical stability: the norm of $\mathbf{S}_t$ is controlled.
- Better in-context associative recall: directly optimizes key-value association prediction (Liu et al. 2024)

Hebbian vs Delta Rule

Comparison Across Different Parameter Scales (Throughput vs AVG_PPL)

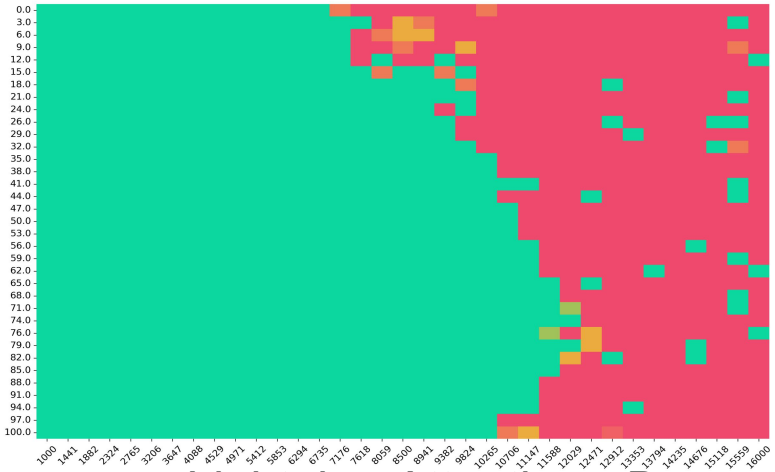


Comparison Across Different Parameter Scales (Throughput vs AVG_CSR)



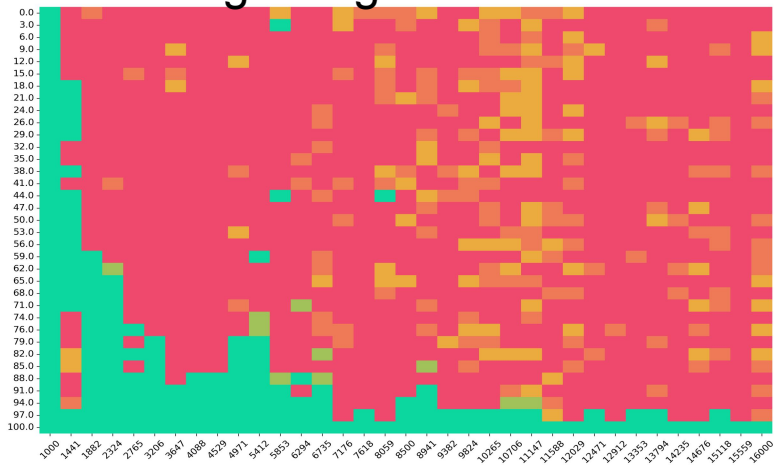
Limitation of Linear Attention

LLaMA-7B



HGRN2-7B

Lightning Attention-7B



Mamba-7B

Solution Overview

GOSIM

Sparse Attention

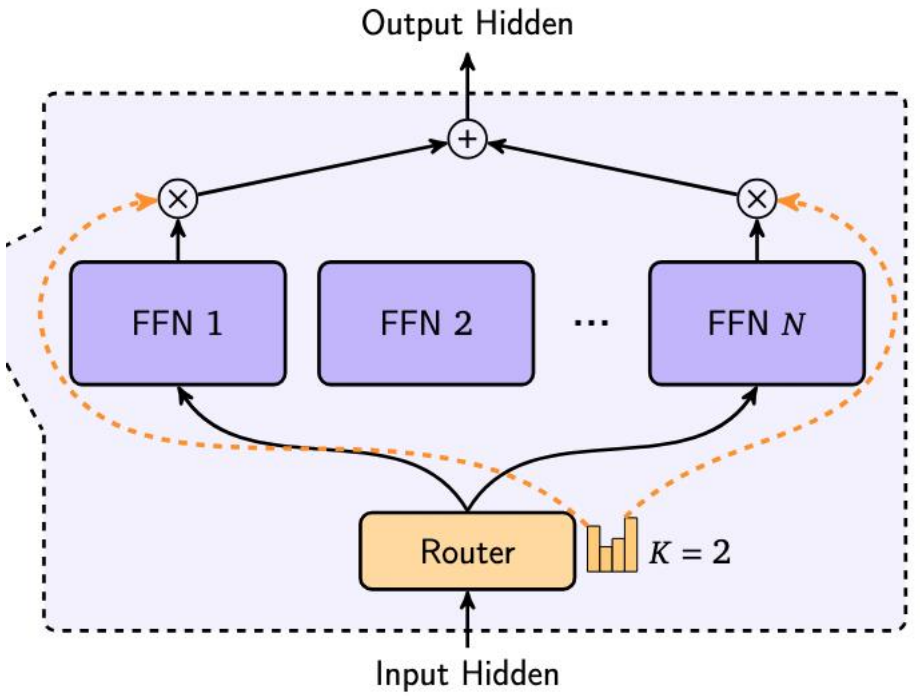
Linear Attention

MoE

Mixture of Experts (MoE)-Unlocking Capacity Without Linear Cost

Key Components

- Gating Network: selects top-k experts
- Experts: usually FFNs or other modules
- Aggregation: weighted sum of expert outputs



Why MoE?

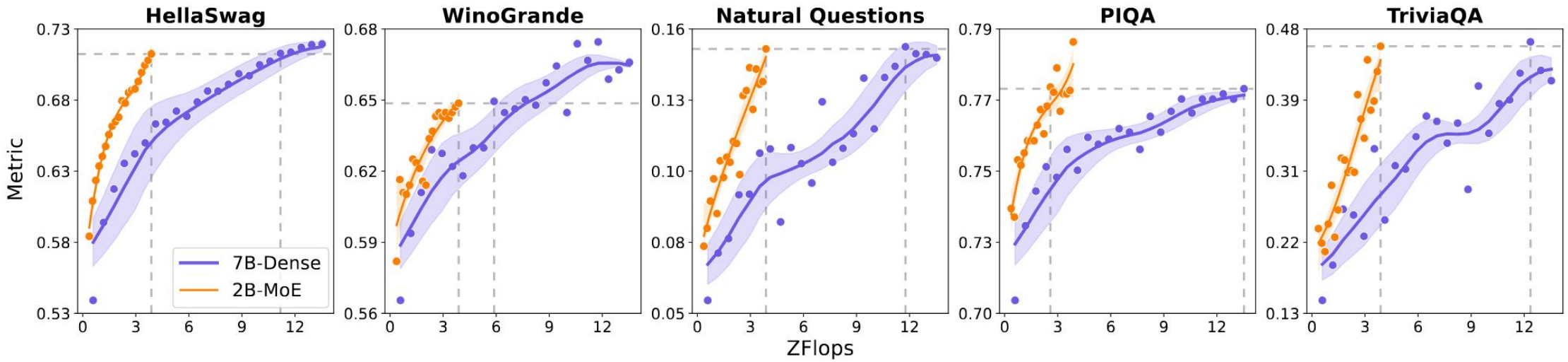
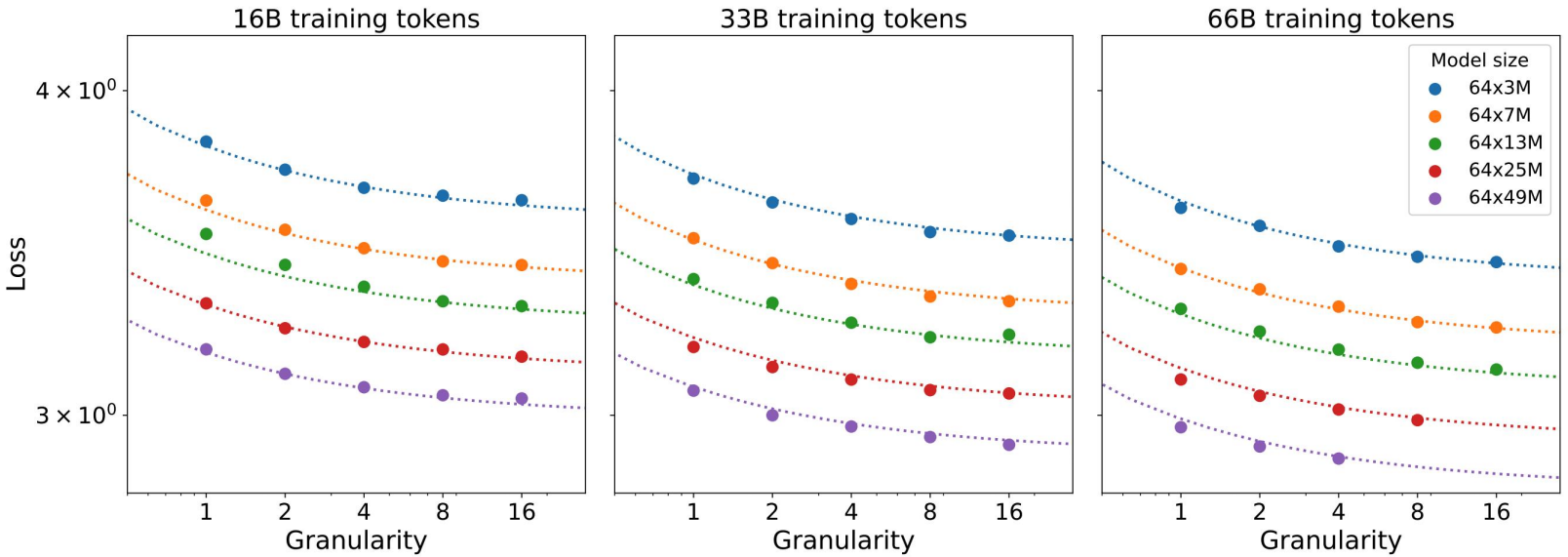


Figure 4 | **Isoflop Comparison: MoE vs. Dense on various benchmarks.** Both models are trained on 1 trillion tokens. The gray dashed lines indicate the difference in the computation required for the two models to achieve the same performance.

Large Experts *vs* Small Experts

Let d_{expert} be the hidden dimension of a single expert. Granularity is defined as

$$G = \frac{d_{\text{ff}}}{d_{\text{expert}}}.$$



Greater granularity implies a larger number of experts. As shown, when total parameters and activation remain constant, fine-grained multi-expert setups achieve better performance.

Scaling laws for fine-grained mixture of experts, ICLR 2024 workshop

Solution Overview

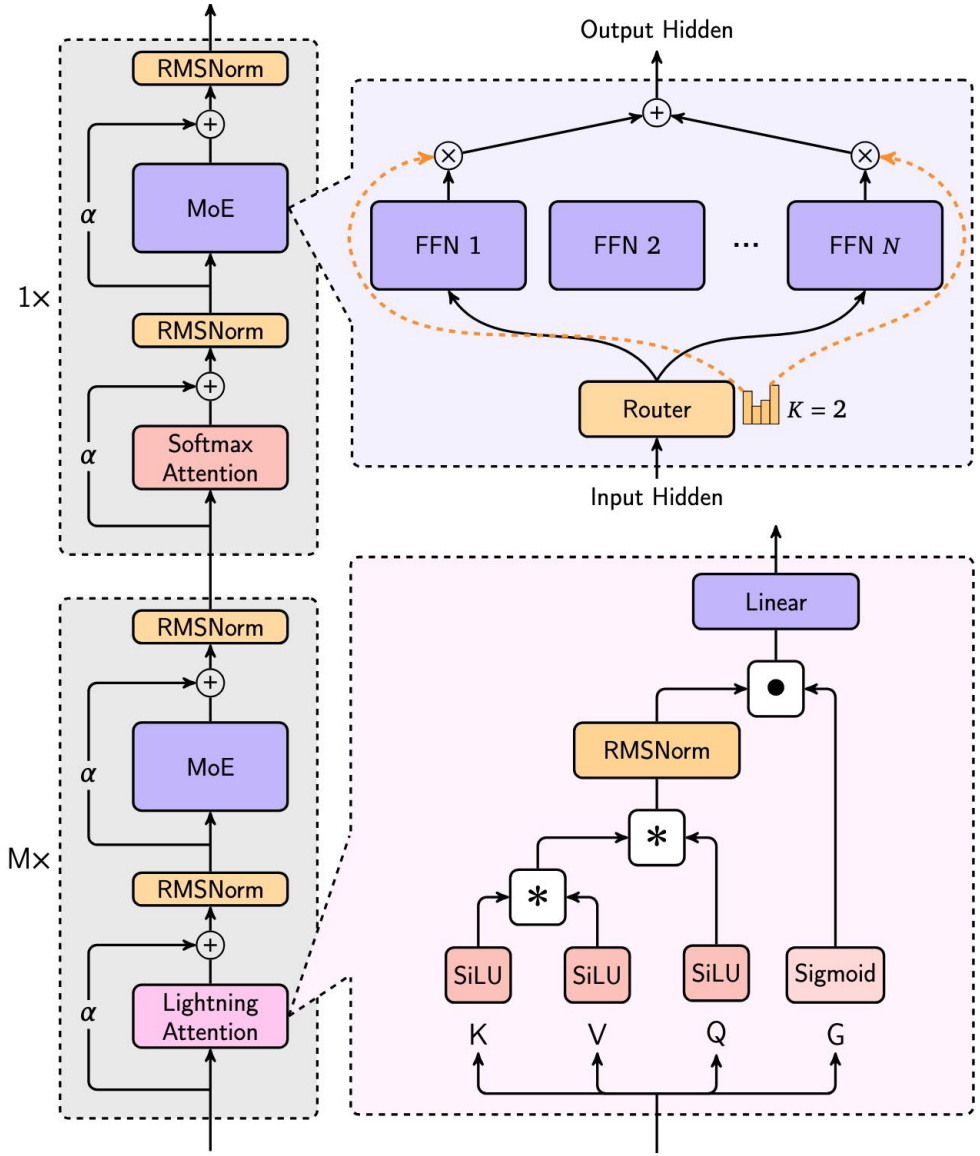
GOSIM

Sparse Attention

Linear Attention

MoE

MiniMax-01



MoE Structure:

456B total parameters with 45.9B activated parameters.

Hybrid Structure:

Lightning attention : Softmax attention = 7 : 1.

Total layers: 80

Head number: 64

Head dimension: 128

Positional Embedding: Half-RoPE only on softmax attention.

Hidden size: 6144.

Expert number: 32 with top-2 routing

Expert hidden size: 9216

Optimization

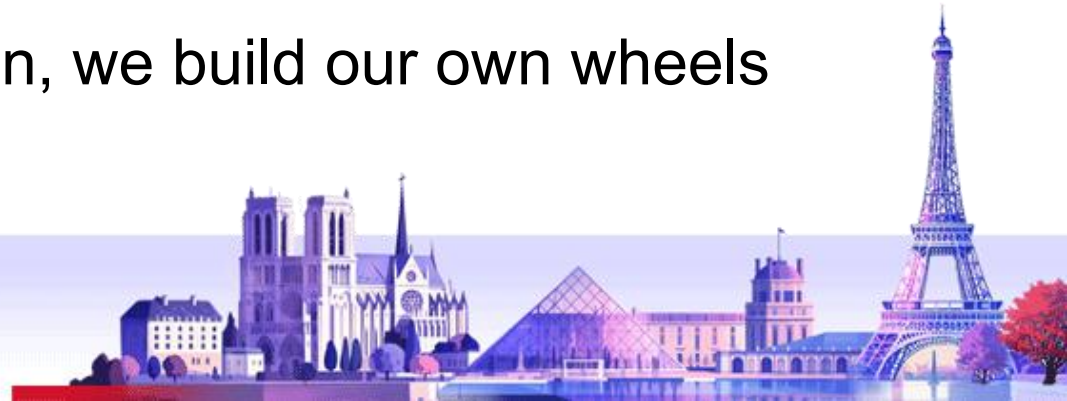
GOSIM

Mitigating the all-to-all (a2a) communication overhead.

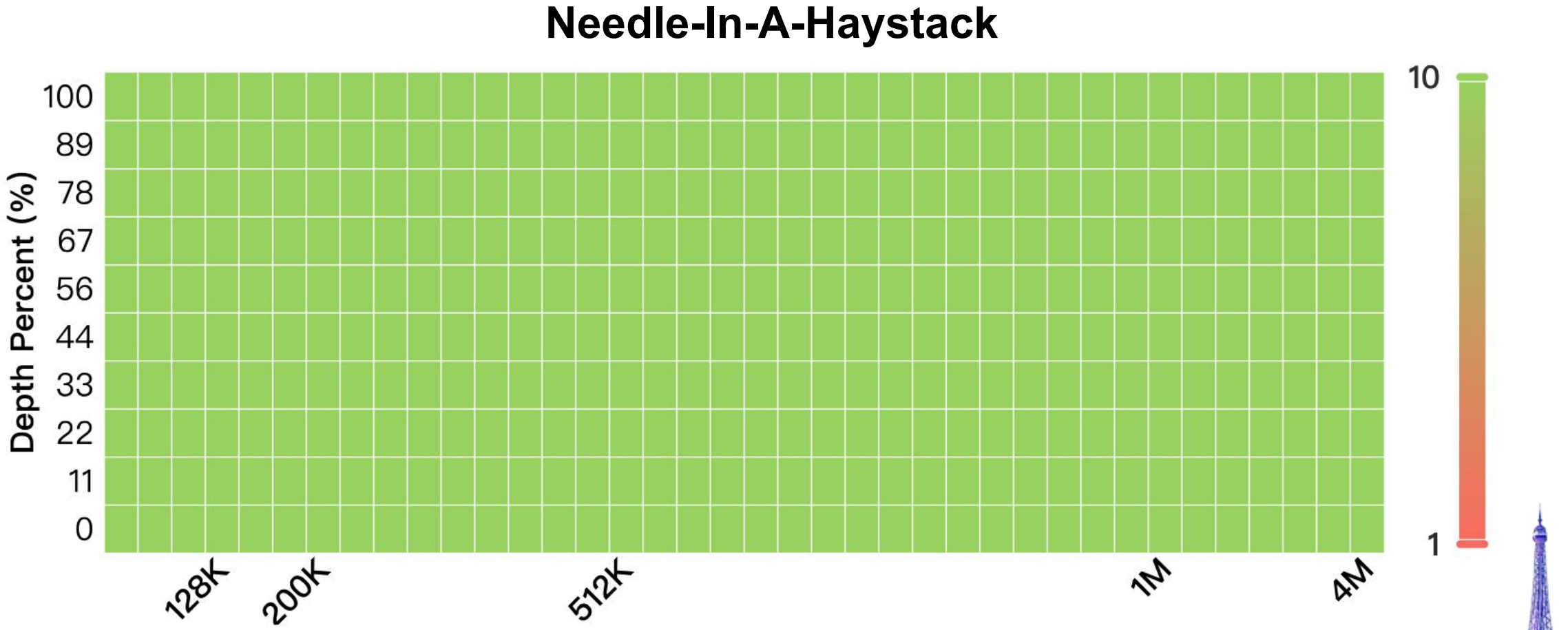
Support at least 1 million token context window in both training and inference.

Lightning attention inference optimization

Without open-source frameworks to rely on, we build our own wheels



Performance

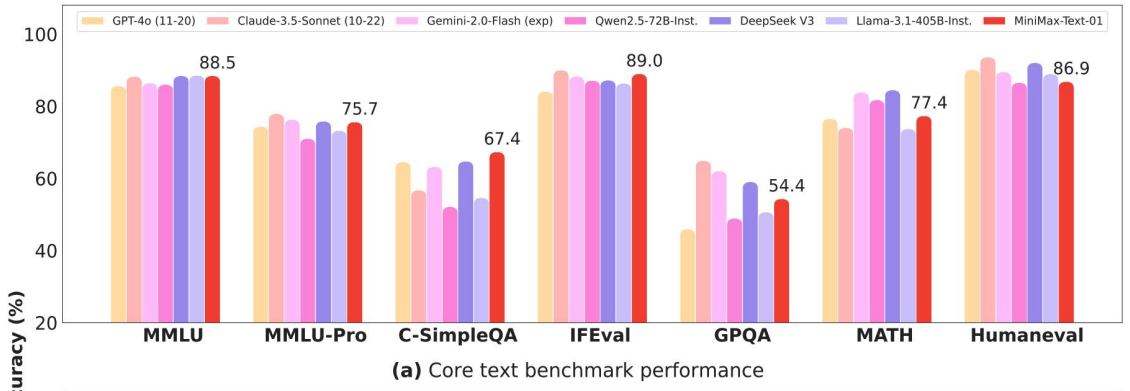


Accuracy

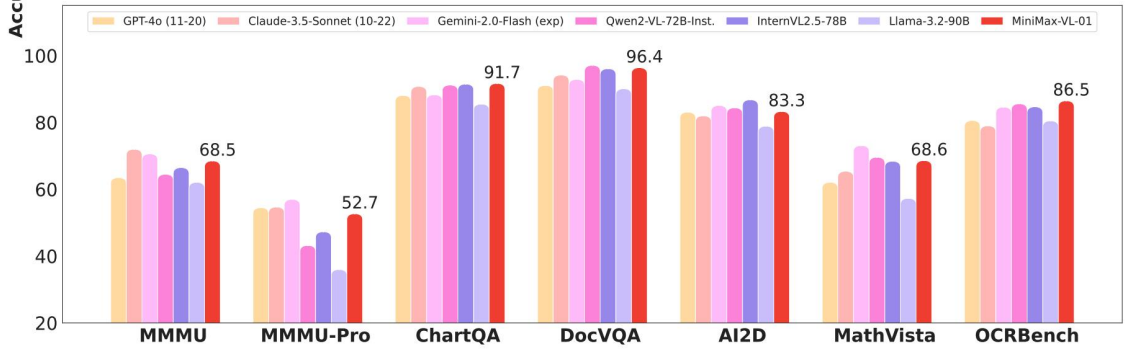
- On par with GPT-4o at the scale of 456B parameters.

Speed

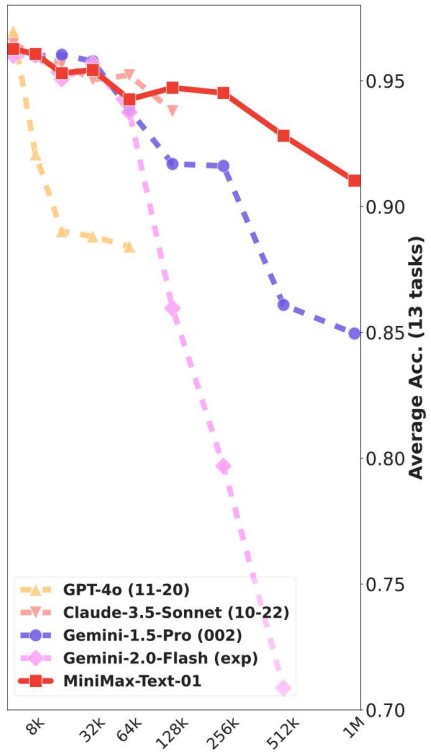
- More than 2700 times faster than Transformer at 1M context length.



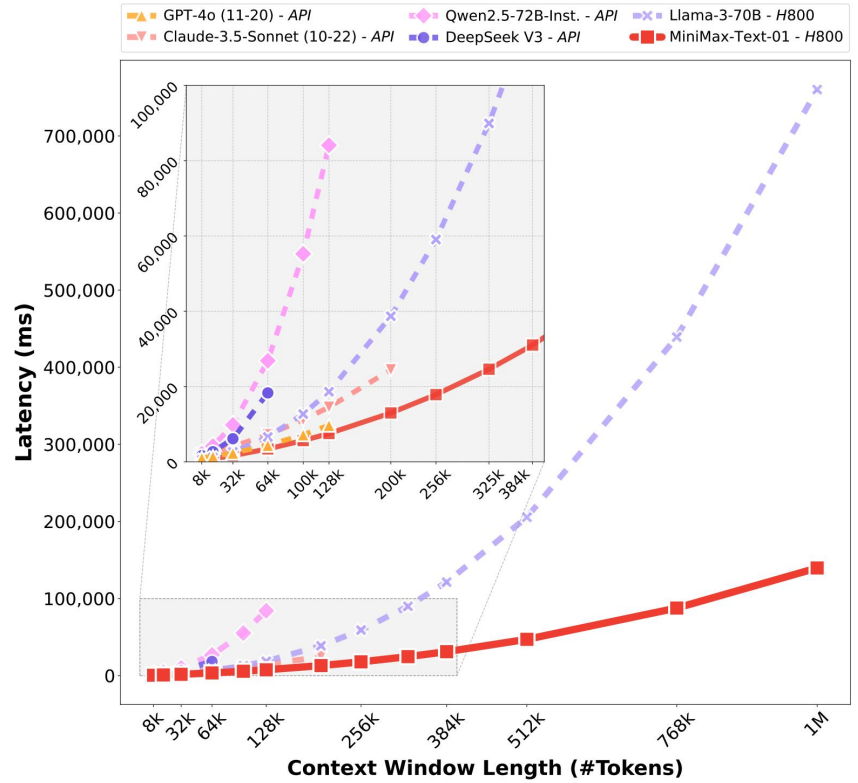
(a) Core text benchmark performance



(b) Core multimodal benchmark performance



(c) Long-context RULER performance



The Benchmark – Linear Next

...

Training Framework

GOSIM

🔥 Flame: Flash Linear Attention Made Easy

Welcome to 🔥 `flame`, a minimal and efficient framework built on `torchtitan` for training Flash Linear Attention (FLA) models with blazing efficiency.

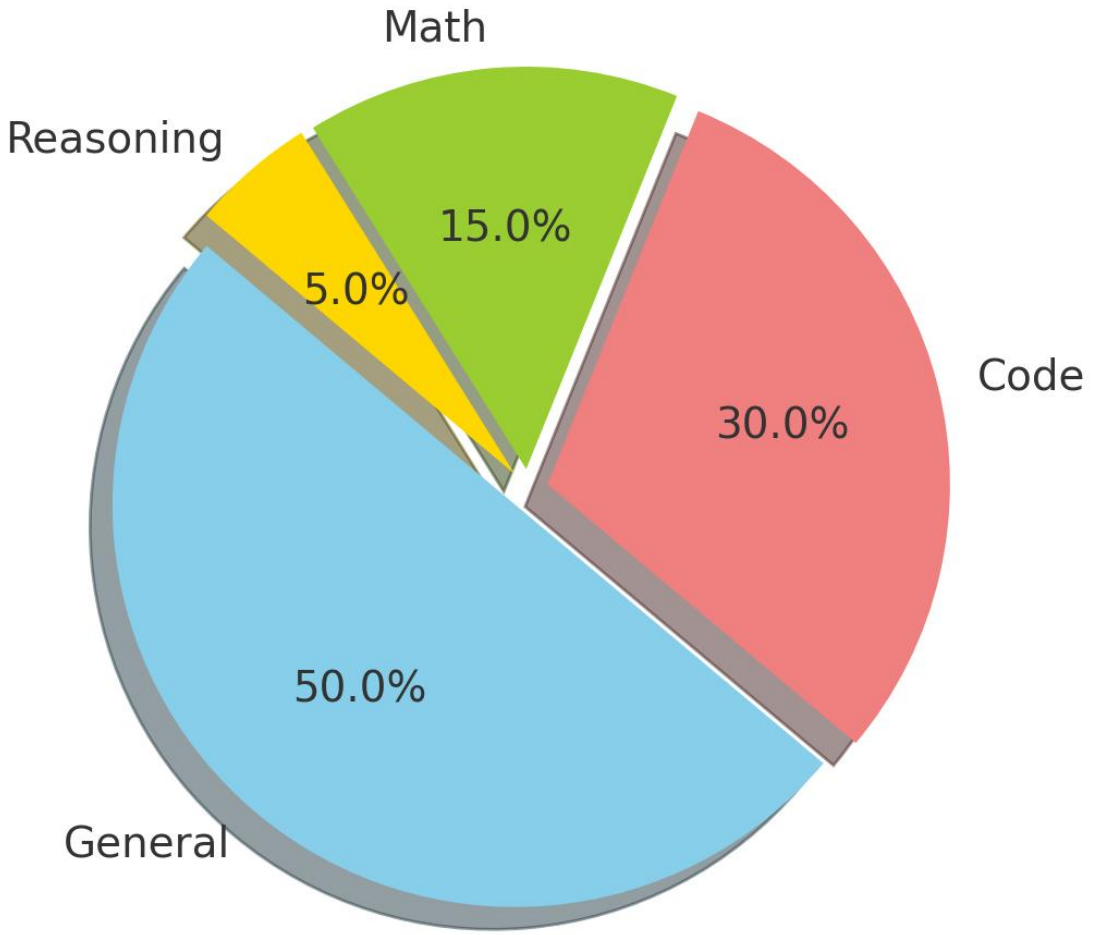
Feature Highlights:

- 🚀 Minimal, easy-to-use, extensible training framework
- 🤗 Seamless integration with `fla` and `transformers`
- 🔄 Zero-cost data preprocessing: online tokenization, dataset shuffling, and multiple datasets support
- 🔮 4D parallelism (coming soon)

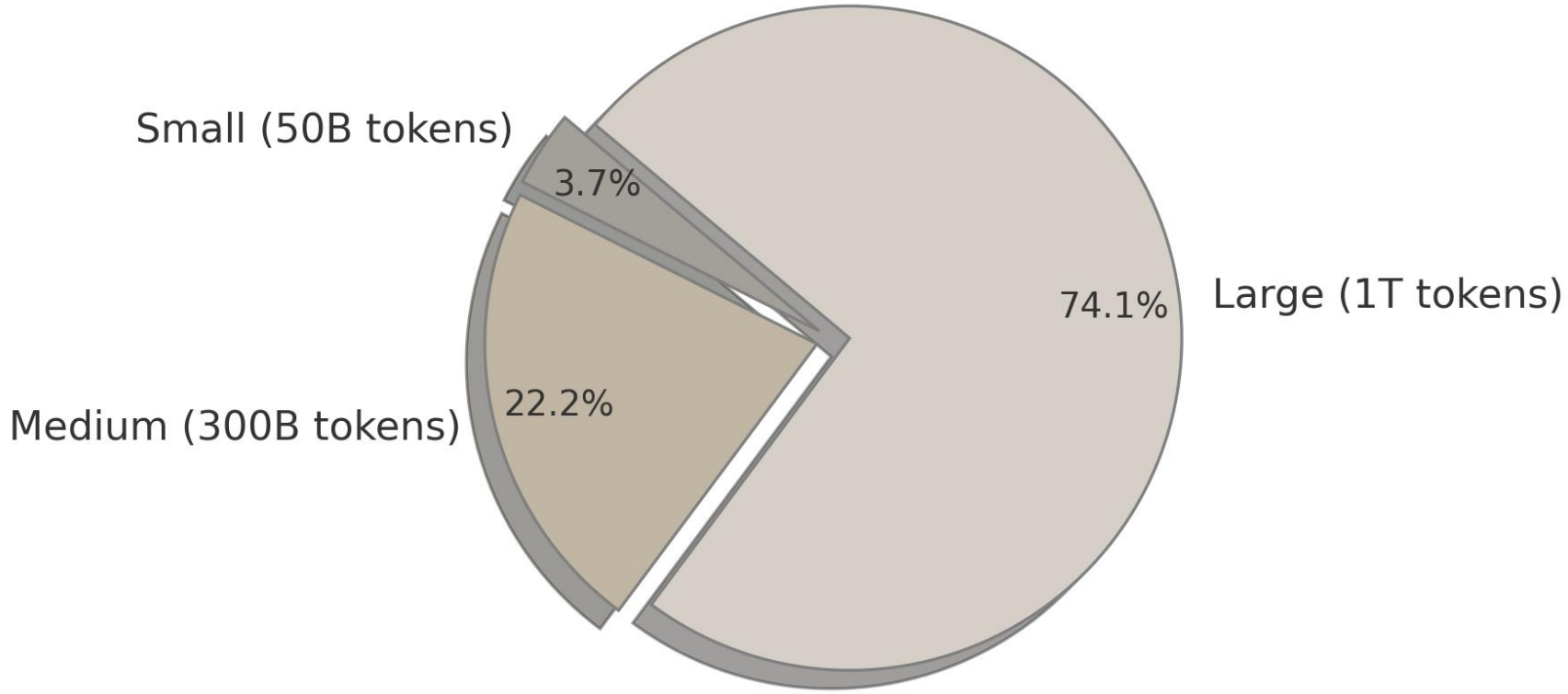
https://github.com/Doraemonzzz/flame

Training Corpus

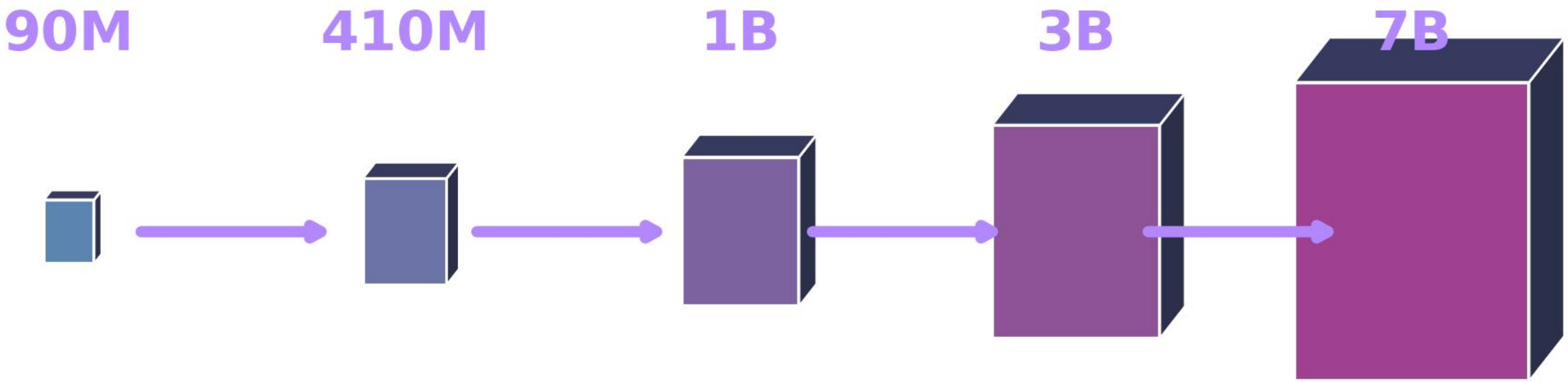
- **General**
  - DCLM-pro,
  - Cosmopedia-v2,
  - Fineweb-edu,
- **Coding**
  - The-stack v2
- **Math**
  - Finemath-shards,
- **Reasoning**
  - Natural_Reasoning



Training Budget



Model Size



Model Variants

GOSIM

Linear Attention

- Lightning Attention
- HGRN1
- HGRN2
- cosFormer1
- cosFormer2
- GLA
- MetaLA
- GSA
- DeltaNet
- Gated DeltaNet
- RWKV7
- Titan
- TTT
- Mamba
- Mamba2

Sparse Attention

- NSA
- MoBA

Hybrid

- MiniMax 01
- ...

Benchmarks

GOSIM

- Knowledge QA
  - MMLU
  - MMLU pro
  - SimpleQA
- General Reasoning
  - GPQA
  - BBH
  - HellaSwag
  - KOR-Bench
- Math
  - GSM8k
  - DROP
  - Math
- Code
  - HumanEval
  - LiveCodeBench
- Long Context
  - NIAH
  - HELMET

GOSIM

GOSIM AI Paris 2025

THANK YOU