Lightning Attention 2:
Handling Unlimited Sequence Lengths

LLM

Attention: Queries, Keys, and Values

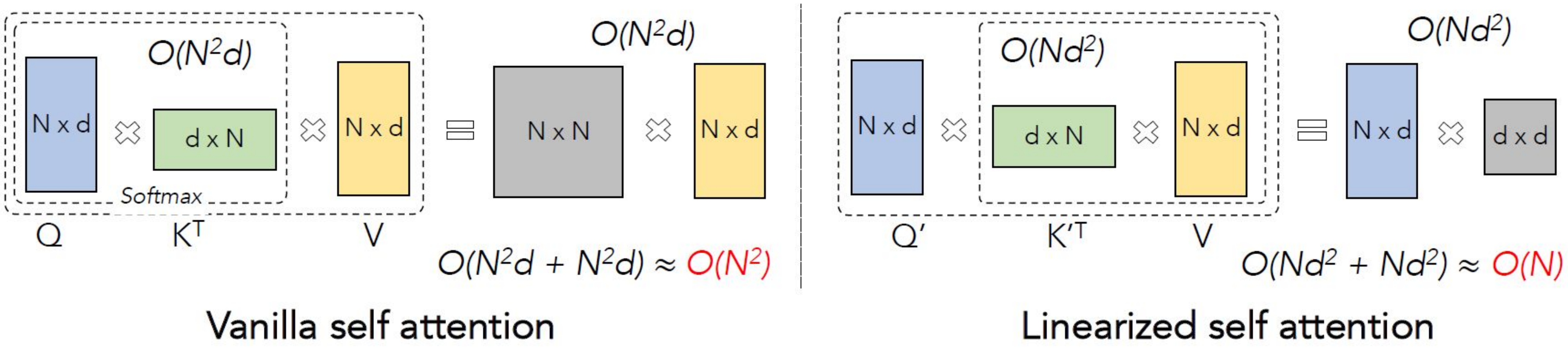
- The attention over the database $\mathcal{D}$

$$\mathrm{Attention}(\mathbf{q}, \mathcal{D}) \stackrel{\mathrm{def}}{=} \sum_{i=1}^{m} \alpha(\mathbf{q}, \mathbf{k}_i)\mathbf{v}_i$$

- where
  - Scalar attention weigths $\alpha(\mathbf{q}, \mathbf{k}_i) \in \mathbb{R}\ (i = 1, \ldots, m)$
  - Dataset $\mathcal{D} \stackrel{\mathrm{def}}{=} \{(\mathbf{k}_1, \mathbf{v}_1), \ldots (\mathbf{k}_m, \mathbf{v}_m)\}$
  - k – key, v – values, q - queries

Attention function

$$\mathcal{O} = \mathcal{A}(x) = [\mathcal{O}_1, \ldots, \mathcal{O}_N]^T, \quad \mathcal{O}_i = \sum_j \frac{\mathcal{S}(Q_i, K_j)}{\sum_j \mathcal{S}(Q_i, K_j)} V_j \qquad \text{output } \mathcal{O} \in \mathbb{R}^{N \times d}$$



$$\mathcal{S}(Q, K) = \exp(QK^T)$$

the dot-product attention with softmax normalization

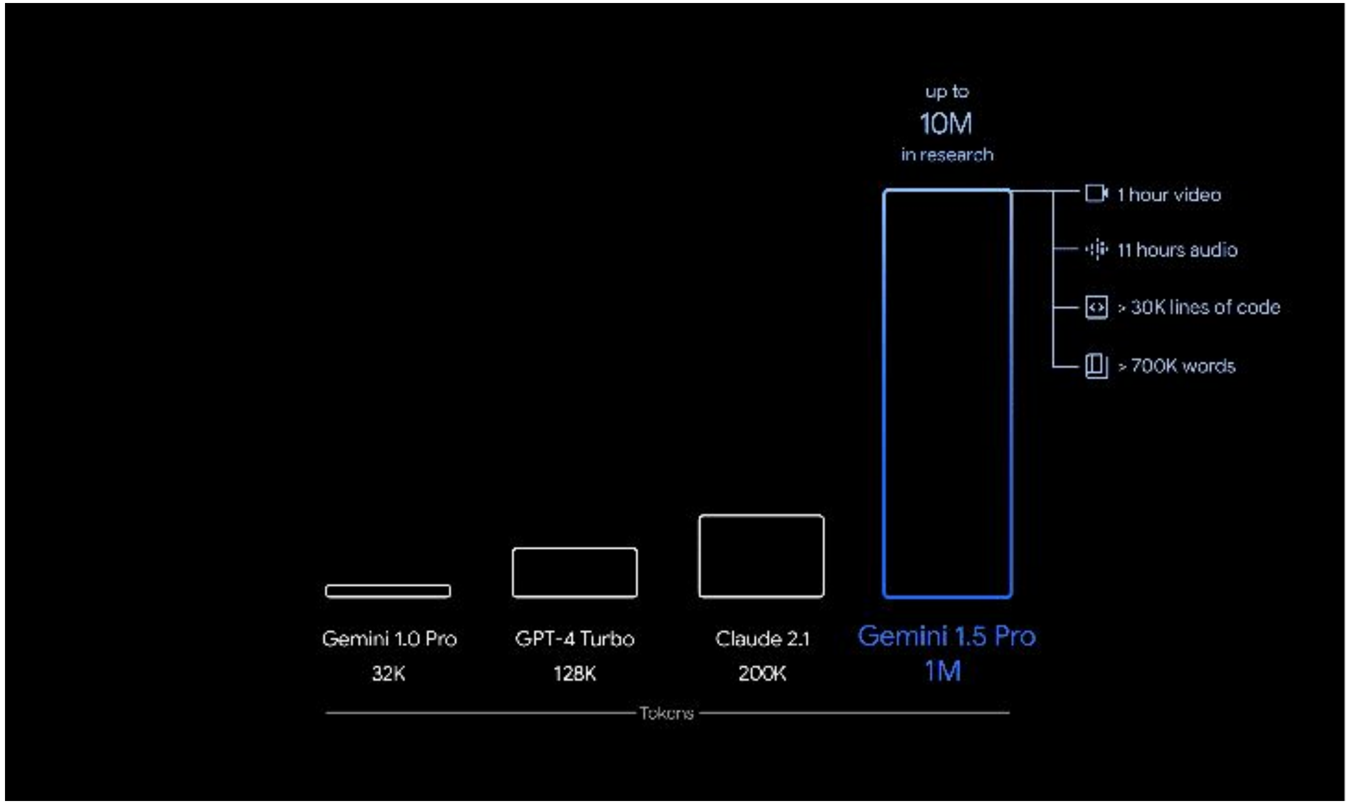
$$\mathbf{O} = \mathrm{Norm}((\mathbf{Q}\mathbf{K}^\top)\mathbf{V})$$

TransNormer

Source: Qin, Zhen, et al. "cosformer: Rethinking softmax in attention." *arXiv preprint arXiv:2202.08791* (2022)

Context Window Size

- Model architecture - The design of the transformer model itself, including the number of layers and the attention mechanism
- Maximum input sequence length - the maximum amount of information or text the model can consider at once
- Memory constraints - Processing longer sequences requires more memory for storing the intermediate states, attention scores, and other necessary computations.
- **Attention mechanism**: The computational complexity of the original transformer architecture grows quadratically with the length of the input sequence, making it challenging to model extremely long sequences.

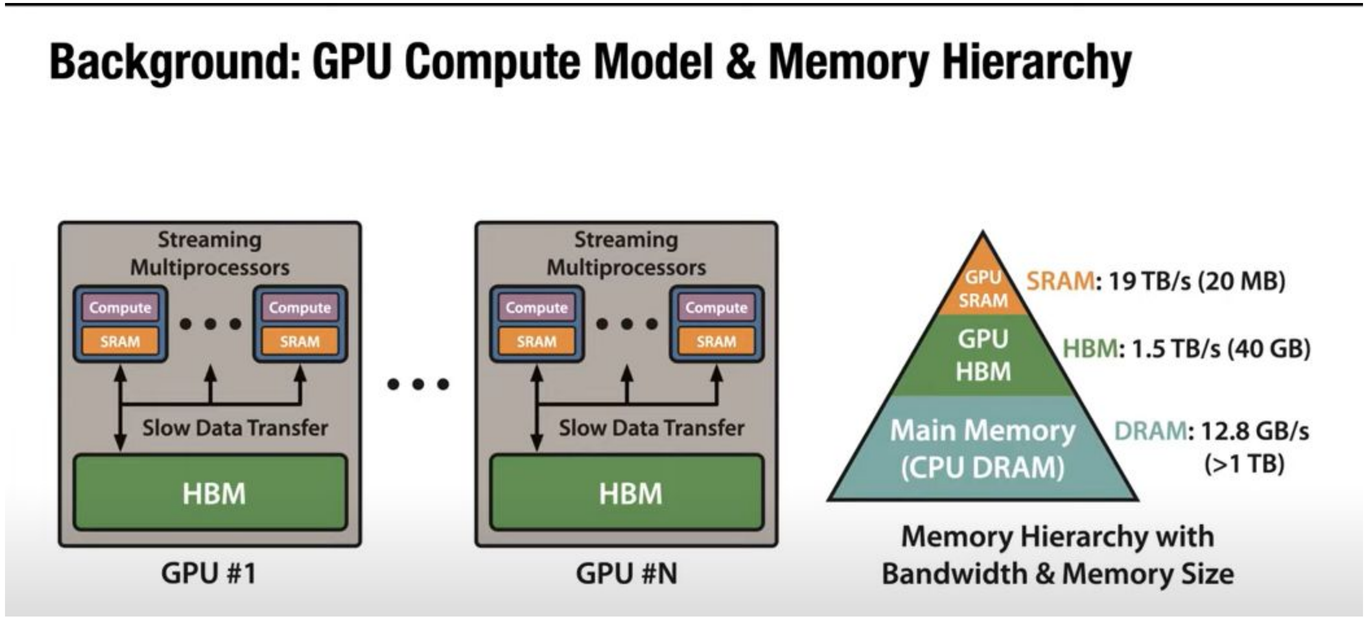


Source: https://analyticsindiamag.com/google-gemini-1-5-crushes-chatgpt-and-claude-with-largest-ever-1-mn-token-context-window/

Challenges of Linear Attention

- The dominance of memory access (I/O) on the GPU could impact the overall computation speed of attention.
  - Solved by Lightning Attention 1
- The cumulative summation (cumsum) needed by the linear attention kernel trick prevents it from reaching its theoretical training speed in the causal setting.
  - Solved by Lightning Attention 2

Structural framework of Lightning Attention 2 (Forward Pass)



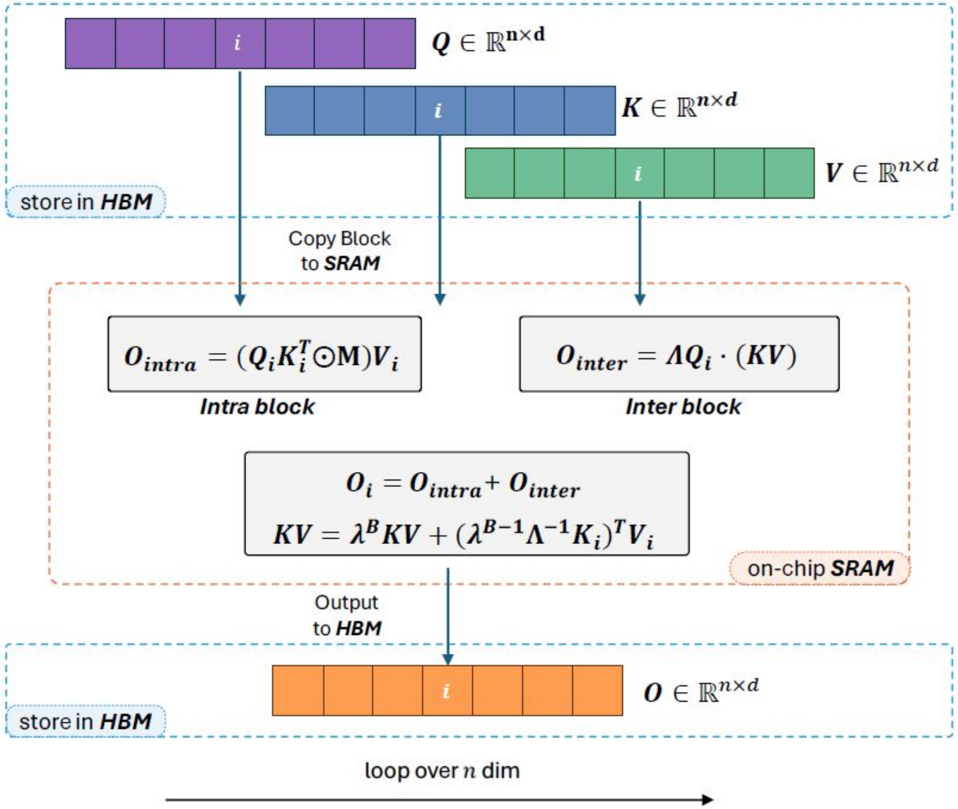


Figure 2. **Structural framework of Lightning Attention-2** is detailed in its algorithmic schematic. During the i-th iteration, the tiling blocks of matrices $\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i$ are transferred from High Bandwidth Memory (HBM) to Static Random-Access Memory (SRAM). Within the SRAM, the outputs $\mathbf{O}_{\text{intra}}$ and $\mathbf{O}_{\text{inter}}$ are computed independently, followed by an update to the $\mathbf{KV}$ matrix. Subsequently, the final output $\mathbf{O}_i$, which is the sum of $\mathbf{O}_{\text{intra}}$ and $\mathbf{O}_{\text{inter}}$, is written back from SRAM to HBM.

Sources:
(left)
https://yashugupta-gupta11.medium.com/understanding-flash-attention-fueling-large-language-models-b037ad02c456
(right) Qin, Zhen, et al. "Lightning Attention-2: A Free Lunch for Handling Unlimited Sequence Lengths in Large Language Models." arXiv preprint arXiv:2401.04658 (2024).

Intricate details of the Lightning Attention-2

$$\mathbf{O} = \mathrm{Norm}(\mathbf{Q}(\mathbf{K}^\top \mathbf{V})), \tag{2}$$

3.2.1. Forward Pass

We ignore the Norm$(\cdot)$ operator in eq. (2) to simplify the derivations. During forward pass of Lightning Attention-2, the t-th output can be formulated as

$$\mathbf{o}_t = \mathbf{q}_t \sum_{s \le t} \lambda^{t-s} \mathbf{k}_s^\top \mathbf{v}_s. \tag{3}$$

In a recursive form, the above equation can be rewritten as

$$\begin{aligned} \mathbf{kv}_0 &= 0 \in \mathbb{R}^{d\times d}, \\ \mathbf{kv}_t &= \lambda \mathbf{kv}_{t-1} + \mathbf{k}_t^\top \mathbf{v}_t, \\ \mathbf{o}_t &= \mathbf{q}_t(\mathbf{kv}_t), \end{aligned} \tag{4}$$

where

$$\mathbf{kv}_t = \sum_{s\le t} \lambda^{t-s} \mathbf{k}_s^\top \mathbf{v}_s. \tag{5}$$

To perform tiling, let us write the equations in block form. Given the total sequence length n and block size B, $\mathbf{X}$ is divided into $T = \frac{n}{B}$ blocks $\{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_T\}$ of size $B \times d$ each, where $\mathbf{X} \in \{\mathbf{Q}, \mathbf{K}, \mathbf{V}, \mathbf{O}\}$.

Given $\mathbf{KV}_t$, the output of $(t+1)$-th block, i.e., $tB+r$, with $1 \le r \le B$ is

$$\begin{aligned} &\mathbf{o}_{tB+r} \\ =&\mathbf{q}_{tB+r} \sum_{s \le tB+r} \lambda^{tB+r-s} \mathbf{k}_s^\top \mathbf{v}_s \\ =&\mathbf{q}_{tB+r} \left(\sum_{s=tB+1}^{tB+r} \lambda^{tB+r-s} \mathbf{k}_s^\top \mathbf{v}_s + \lambda^r \sum_{s \le tB} \lambda^{tB-s} \mathbf{k}_s^\top \mathbf{v}_s \right) \\ =&\mathbf{q}_{tB+r} \sum_{s=tB+1}^{tB+r} \lambda^{tB+r-s} \mathbf{k}_s^\top \mathbf{v}_s + \lambda^r \mathbf{q}_{tB+r} \mathbf{kv}_{tB}. \end{aligned} \tag{7}$$

Rewritten in matrix form, we have

the Hadamard product, or element-wise multiplication of two matrices

$$\begin{aligned} \mathbf{O}_{t+1} =& \underbrace{[(\mathbf{Q}_{t+1}\mathbf{K}_{t+1}^\top) \odot \mathbf{M}]\mathbf{V}_{t+1}}_{\text{Intra Block}} \\ &+ \underbrace{\Lambda \mathbf{Q}_{t+1}(\mathbf{KV}_t)}_{\text{Inter Block}}, \end{aligned} \tag{8}$$

Throughout Comparison

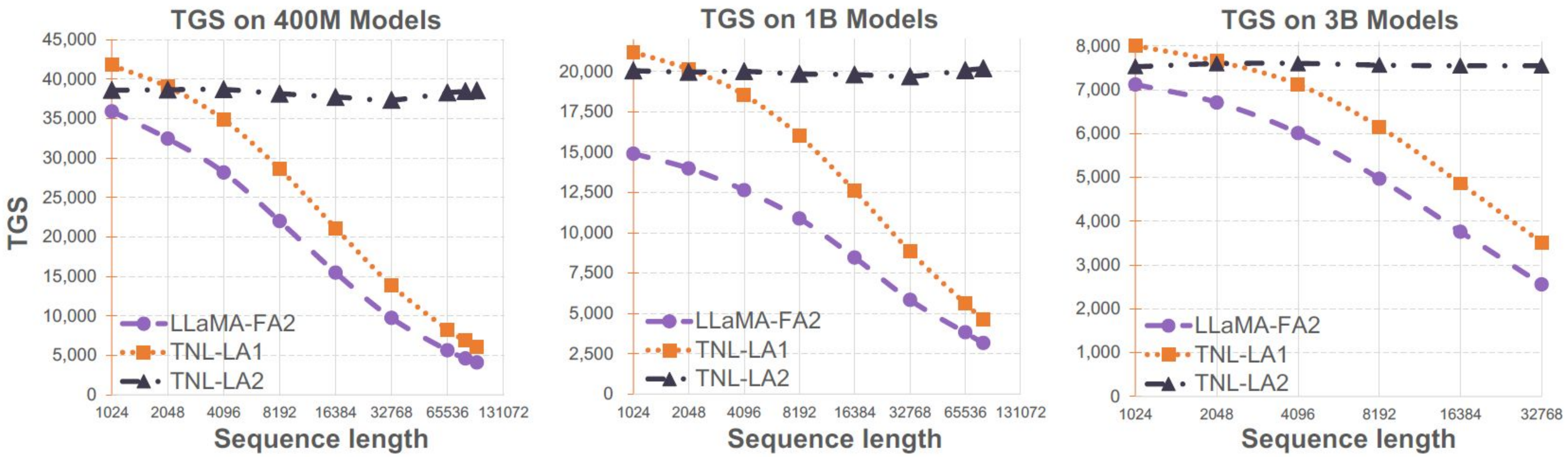


Figure 1. **Speed Showdown: FlashAttention vs. Lightning Attention in Expanding Sequence Lengths and Model Sizes.** The diagram above provides a comparative illustration of training speed, Token per GPU per Second (TGS) for LLaMA with FlashAttention-2, TransNormerLLM with Lightning Attention-1 and TransNormerLLM with Lightning Attention-2, implemented across three model sizes: 400M, 1B, and 3B from left to right. It is strikingly evident that Lightning Attention-2 manifests a consistent training speed irrespective of the increasing sequence length. Conversely, the other methods significantly decline training speed as the sequence length expands.

Speed and Memory Usage Comparison

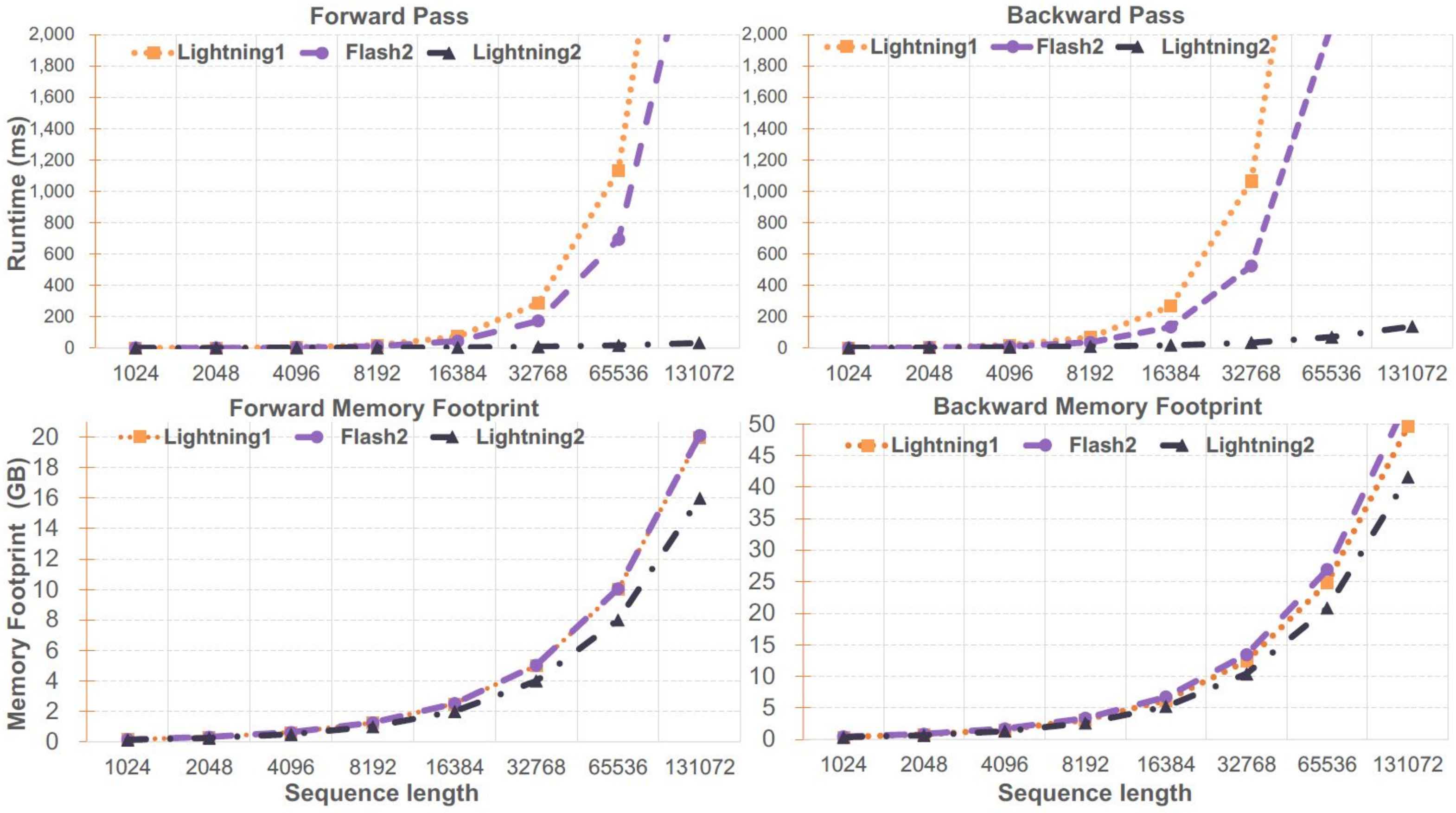


Figure 3. **Comparative Analysis of Speed and Memory Usage: FlashAttention vs. Lightning Attention.** Upper Section: Runtime in milliseconds for the forward and backward pass across varying sequence lengths. Lower Section: Memory utilization during the forward and backward pass at different sequence lengths.

Performance Comparison

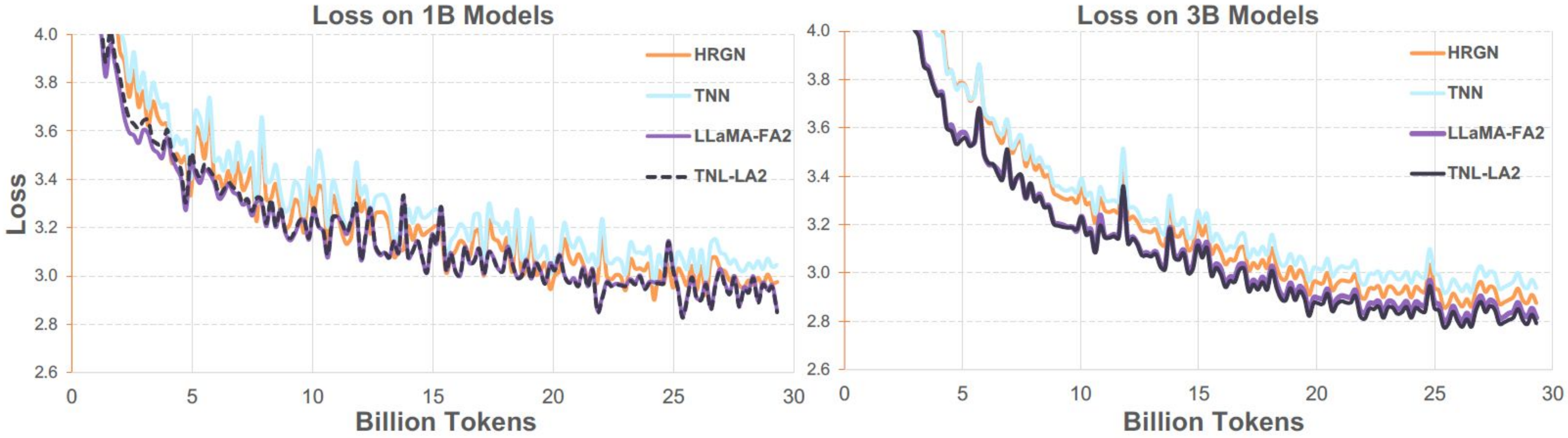


Figure 4. **Performance Comparison of HGRN, TNN, LLaMA with FlashAttention2 and TransNormerLLM with Lightning Attention-2.** For the 1B model, we used 16×A800 80G GPUs with a batch size of 12 per GPU; for the 3B model, we scaled up to 32×A800 80G GPUs and a batch size of 30 per GPU. The training context length was set to 2K.

Table 3. **Performance Comparison on Commonsense Reasoning and Aggregated Benchmarks.** TNL-LA2: TransNormerLLM with Lightning Attention-2. PS: parameter size (billion). T: tokens (billion). HS: HellaSwag. WG: WinoGrande.

Model	PS	T	BoolQ	PIQA	HS	WG	ARC-e	ARC-c	OBQA	CSR	C-Eval	MMLU	C-Eval	MMLU
	B	B	acc	acc	acc_norm	acc	acc	acc_norm	acc_norm	avg.	acc-0shot	acc-0shot	acc-5shot	acc-5shot
Pythia	12	50.3	**62.14**	71.76	51.89	55.64	59.22	28.75	**32.80**	51.74	22.36	25.80	21.43	26.10
TNL-LA2	15	49.8	62.08	**72.52**	**55.55**	**57.14**	**62.12**	**31.14**	32.40	**53.28**	**25.55**	**26.60**	**26.18**	**27.50**
Pythia	12	100.6	62.20	73.23	58.83	59.35	63.76	31.91	32.80	54.58	24.00	24.80	24.45	24.40
TNL-LA2	15	99.7	**63.98**	**74.70**	**61.09**	**61.33**	**65.95**	**34.64**	**35.60**	**56.76**	**26.70**	**26.90**	**25.38**	**27.40**