Learning When to Attend: Conditional Memory Access for Long-Context LLMs

Sakshi Choudhary [*1] **Aditya Chattopadhyay** [*2] **Luca Zancato** [2] **Elvis Nunez** [2] **Matthew Trager** [2] **Wei Xia** [2] **Stefano Soatto** [2]

Abstract

Language models struggle to generalize beyond pretraining context lengths, limiting long-horizon reasoning and retrieval. Continued long-context training is effective but expensive due to the quadratic cost of Attention. We observe that most tokens do not require (Global) Attention over the entire sequence and can rely on local context. Based on this, we propose *L2A* (Learning To Attend), a layer that enables token-wise conditional long-range memory access by learning *when* to invoke Global Attention. Applied to Qwen2.5 and Qwen3 models, *L2A* extends the effective context length from 32K to 128K tokens, matching standard long-context training within 3% while skipping Global Attention for ∼80% of tokens and outperforming prior baselines. We further design custom Triton kernels to efficiently implement *L2A* on GPUs, achieving ∼2× improvements in training throughput and time-to-first-token over FlashAttention. Additionally, *L2A* enables post-training pruning of sparse Global Attention layers, reducing KV cache memory by up to 50% with negligible performance loss.

1. Introduction

Large Language Models (LLMs) are being increasingly deployed in long-context scenarios such as multi-turn conversations (Maharana et al., 2024) and code understanding (Jimenez et al., 2023). However, training long-context LLMs is computationally expensive due to the quadratic complexity of Attention (Liu et al., 2024a). Prior approaches either approximate Attention during training (Yuan et al., 2025; Nunez et al., 2025; Mohtashami & Jaggi, 2023) or extend context lengths without retraining (Jin et al., 2024; An et al., 2024). However, these often underperform compared to directly fine-tuning LLMs at the target context length (Lu et al., 2024), a strategy commonly referred to as continued long-context pre-training (CLP).

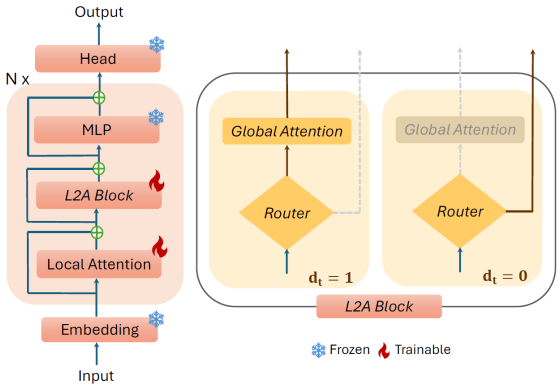

Figure 1. **Overview of *L2A*.** All tokens first apply Local Attention, followed by token-wise routing: tokens with $\mathbf{d}_t = 1$ invoke Global Attention, whereas tokens with $\mathbf{d}_t = 0$ bypass it. This enables efficient long-context modeling by invoking Global Attention selectively. The combined {Local Attention + *L2A* Block} forms the *L2A* layer.

While Attention enables effective long-context understanding via full-context retrieval (Zancato et al., 2024), not all tokens require access to the entire context (Stewart et al., 2024).In fact, for most tokens, the recent past provides sufficient context. Motivated by this, we propose *L2A*, a sequence modeling layer that adaptively decides, at each token, whether to apply Global Attention or rely on Local Attention (Figure 1). Specifically, *L2A* first computes Local Attention, and then uses a learned router to selectively trigger Global Attention for tokens requiring long-range dependencies. This treats Global Attention as a conditional long-term memory retrieval mechanism, while maintaining efficiency through sparsity. The model is trained end-to-end with a next-token prediction objective and a sparsity-inducing regularizer. We show that this strategy preserves long-context performance while substantially reducing the computational cost of long-context fine-tuning.

Unlike existing approaches that compute Attention for every query over a sparsified set of past key–value tokens (Yuan et al., 2025; Chen et al., 2024), *L2A* computes exact Attention over the full context, but only for a sparse

[*]Equal contribution . Work done during an internship at AWS Agentic AI. [1]Department of Electrical and Computer Engineering, Purdue University, USA [2]AWS Agentic AI, USA. Correspondence to: Aditya Chattopadhyay <achatto@amazon.com>.

Proceedings of the 43^{rd} International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026.

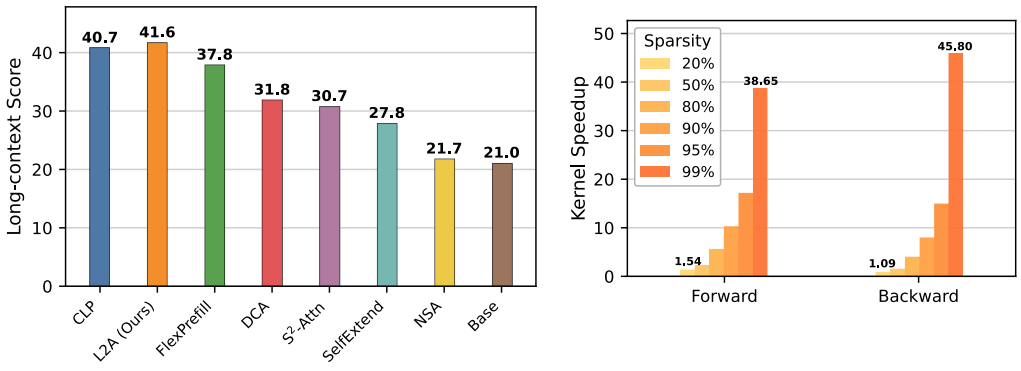

Figure 2. **Long-context performance and efficiency of *L2A* at 128K context length.** *Left*: Performance comparison of *L2A* against CLP for base model and other baselines on Qwen 2.5 7B, where *L2A* outperforms existing approaches. *Right*: Kernel-level speedups achieved by *L2A* relative to FlashAttention-2, with $\sim$10$\times$ and $\sim$8$\times$ speedups for the forward and backward passes at upto 90% sparsity in practice.

subset of query tokens. This shifts the focus from *what* to retrieve to *when* retrieval is necessary, enabling token-wise adaptive computation. For instance, in needle-in-a-haystack tasks (Kamradt, 2024), long-term memory access might be needed only at specific token positions, whereas for in-context learning, Global Attention may be triggered frequently to compare and reuse provided examples.

L2A is a drop-in replacement for standard Attention layers, enabling direct fine-tuning of off-the-shelf LLMs on long contexts via continued pre-training. To realize practical speedups, we design custom Triton kernels that yield up to 2$\times$ improvement in training throughput and time-to-first-token compared to FlashAttention-2 (FA-2) (Dao, 2023). Empirically, *L2A* extends the context length of Qwen2.5 and Qwen3 models from 32K to 128K context. Across long-context benchmarks such as HELMET (Yen et al., 2025), BabiLong (Kuratov et al., 2024), and MRCR (Vodrahalli et al., 2024), it matches CLP performance within 1.5–3%, while reducing training costs by allowing 75–80% of tokens to skip Global Attention. Moreover, *L2A* consistently outperforms sparse Attention-based training methods and training-free baselines across different model scales (1.5B and 7B). Figure 2 presents an overview of these results.

Despite *L2A*'s selectivity and higher efficiency, it requires storing the whole past KV cache, which can be expensive for very long sequences. To address this, we propose a post-training layer-pruning strategy that retains only a local cache (4K tokens) for layers in which Global Attention is rarely activated (for sparsity $\geq$ 95%). Compared to a similarly-sized Transformer, this reduces KV cache size by up to 50% at 128K context length with minimal performance degradation ($\leq$ 1%).

Contributions. (1) We introduce *L2A*, a token-wise conditional Attention mechanism for efficient long-context training and inference. (2) We design Triton kernels that exploit the sparsity arising from conditional long-term memory access to achieve upto 10$\times$ speedups over FA-2. (3) Empirical evaluation shows that *L2A* extends the context length of Qwen models from 32K to 128K tokens, matching the performance of CLP while being 2$\times$ faster to train.

2. Preliminaries & Prior Work

Attention Mechanism. Given token embeddings $\mathbf{X} \in \mathbb{R}^{n\times d}$, queries, keys, and values are computed as:

$$\mathbf{Q} = \mathbf{X}\mathbf{W}^{\mathbf{Q}}, \quad \mathbf{K} = \mathbf{X}\mathbf{W}^{\mathbf{K}}, \quad \mathbf{V} = \mathbf{X}\mathbf{W}^{\mathbf{V}}. \tag{1}$$

Global Attention is defined as:

$$\mathrm{Attn}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}} + M\right)\mathbf{V}, \tag{2}$$

where $M_{ij} = 0$ if $j \leq i$ and $-\infty$ otherwise for causality.

The $\mathbf{Q}\mathbf{K}^\top$ term scales quadratically with sequence length, making Attention a bottleneck for long contexts. We refer to (2) as Global Attention, in contrast to Local Attention (e.g., sliding window), which restricts each token to a window of size w by setting $M_{ij} = 0$ if $0 \leq i - j \leq w$.

Sparse (Approximate) Attention Variants. Trainable sparse Attention methods reduce training cost by compressing keys (Munkhdalai et al., 2024), selecting landmarks (Mohtashami & Jaggi, 2023), or imposing fixed sparse patterns (Chen et al., 2024). Approaches such as S^2-Attn (Chen et al., 2024) and Landmark Attention (Mohtashami & Jaggi, 2023) improve efficiency through sparse or summarized access to past context, but often fail to capture exact long-range dependencies, leading to degraded performance on challenging tasks (Gao et al., 2025; Lu et al., 2024). SE-Attention (Nunez et al., 2025) improves the efficiency of Transformer architectures by replacing Global Attention with a Local Attention that compresses and retrieves blocks of tokens from the past. Similarly, Native Sparse Attention (NSA) (Yuan et al., 2025) combines compressed, selected, and local context within a sliding window. These methods

assume a fixed budget per query, attending to a predetermined set of key blocks regardless of contextual needs. As a result, sparsity is fixed and does not adapt to input or task complexity, which can lead to sub-optimal long-context performance, since the amount of relevant past information is difficult to determine a priori during training.

In contrast, training-free sparse Attention methods primarily target inference efficiency by reducing Attention cost during prefill and decoding, for example, through FlexPrefill and MInference for prefill acceleration and H2O and TokenSkip for faster decoding (Lai et al., 2025; Jiang et al., 2024a; Zhang et al., 2023; Xia et al., 2025). These approaches focus on inference-time optimization without providing training-time efficiency benefits. Please refer to Section B for further discussion on context length extrapolation, dynamic routing, and external memory-based approaches.

3. L2A: Learning When to Attend

3.1. The *L2A* Layer

As shown in Figure 1, *L2A* consists of Local Attention, a Router, and Global Attention. Local Attention is implemented via Sliding Window Attention (SWA), while Global Attention is realized using (2) over the full context. Both are initialized from the Attention layers of a pre-trained Base LLM. The Router is a linear projection followed by a sigmoid activation, with weights initialized to zero to trigger Global Attention for all tokens at initialization.

Base LLM $\rightarrow$ *L2A* LLM. We replace all Attention layers in the Base LLM with *L2A* layers, keeping Feed-Forward Network (FFN) layers unchanged. We refer to the resulting model as the *L2A* LLM, which is then trained for long-context modeling. Incorporating both Local and Global Attention introduces only a modest increase in parameter count ($\sim$10% for Qwen models) as majority of parameters reside in FFN layers.

Forward Pass for *L2A* Layer: Given an input sequence $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$, each token $\mathbf{x}_t$ first attends to a local context of size $k \ll n$ using SWA:

$$\mathbf{s}_t = \mathrm{LocalAttn}(\mathbf{x}_t, \mathbf{x}_{t-1}, \ldots, \mathbf{x}_{t-k}), \quad t = 1, \ldots, n. \tag{3}$$

This local representation, which we denote $\mathbf{s}_t$, is then passed to the Router module, which computes a score $\hat{\mathbf{d}}_t$ in $[0, 1]$ that is then thresholded to obtain the Router's decision $\mathbf{d}_t$:

$$\hat{\mathbf{d}}_t = \sigma(\mathbf{W}\mathbf{s}_t), \quad \mathbf{d}_t = \begin{cases} 1, & \text{if } \hat{\mathbf{d}}_t \geq 0.5 \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

Here, $\mathbf{W} \in \mathbb{R}^{1\times d}$ denotes a learnable linear projection for the Router and $\sigma(.)$ refers to the Sigmoid activation function. If $\mathbf{d}_t = 1$, the token is processed also via Global Attention, otherwise, it retains its local representation $\mathbf{s}_t$.

$$\mathbf{a}_t = \begin{cases} \mathrm{GlobalAttn}(\mathbf{s}_t, \mathbf{s}_{t-1}, \ldots, \mathbf{s}_1), & \text{if } \mathbf{d}_t = 1 \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

The final output is thus given as:

$$\mathbf{o}_t = \mathbf{s}_t + (\mathbf{a}_t \cdot \mathbf{d}_t) \tag{6}$$

Challenges. Implementing such a conditional computation scheme introduces two challenges: (i) sparse execution is inefficient on GPUs due to irregular memory access (Li et al., 2025; Gray et al., 2017), and (ii) training LLMs with input-conditional computation graphs can be unstable due to discontinuous routing decisions, leading to *Router collapse* (Wang et al., 2024b). We address these via custom Triton kernels (Section 3.2) and a stable training procedure described in Section 3.3.

3.2. *L2A* Kernel Design

To translate algorithmic sparsity into wall-clock speedups, we design a custom Triton kernel based on the tiled attention computation of FlashAttention-2 (FA-2). FA-2 streams tiled $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ blocks from HBM to on-chip SRAM, computes attention incrementally with causal masking, and uses an online log-sum-exp scheme for numerical stability, avoiding materialization of the full attention matrix. Building on this, our kernel supports token-level conditional sparsity by selectively computing Attention only for queries that trigger Global Attention. Active queries are first compacted into a contiguous buffer $\mathbf{Q}_c$, while their original sequence positions are recorded in an index mapping $\mathbf{Q}_{idx}$ to preserve causality. Attention is then computed over tiles of $\mathbf{Q}_c$, $\mathbf{K}$, and $\mathbf{V}$, with causality enforced via the original indices in $\mathbf{Q}_{idx}$ rather than the compacted order. This allows the kernel to skip irrelevant key–value tiles, reducing computation while maintaining correctness. Outputs are written back and scattered to their original positions using the index mapping.

This design preserves the efficiency and stability of FA-2 while enabling conditional sparsity. The forward pass is detailed in Algorithm 1. The backward pass follows the same principle, computing gradients only for active queries using the saved statistics L_c and index mapping $\mathbf{Q}_{idx}$.

3.3. *L2A* Training

Training Objective. We train *L2A* using the standard next-token prediction (NTP) loss with an additional sparsity regularizer that penalizes frequent use of Global Attention:

$$\mathcal{L} = \mathcal{L}_{\text{NTP}} + \frac{\lambda}{nL} \sum_{l=1}^{L} \sum_{t=1}^{n} \hat{\mathbf{d}}_{l,t}^2, \tag{7}$$

Here, $\hat{\mathbf{d}}_{l,t}$ denotes the router output at layer l and position t. Since the router decision $\mathbf{d}_t$ described in Equation (4) is discrete, we employ the straight-through estimator (STE) (Bengio et al., 2013) to enable backpropagation. The discrete decision is used in forward, while gradients flow through the continuous sigmoid output during backpropagation.

Training Instabilities. Naively optimizing Equation (7) results in *Router collapse*, where Global Attention is never invoked. This can be understood by analyzing the gradient of the loss with respect to the Router score:

$$\frac{\partial \mathcal{L}}{\partial \hat{\mathbf{d}}_t} = \left(\frac{\partial \mathcal{L}_{\text{NTP}}}{\partial \mathbf{o}_t}\right)^T \mathbf{a}_t + \frac{2\lambda}{nL}\hat{\mathbf{d}}_t. \tag{8}$$

Since Global Attention outputs $\mathbf{a}_t$ (Equation (5)) are computed only for query tokens that invoke it, *masked* tokens have $\mathbf{a}_t = 0$ and receive no gradient signal from the NTP loss. As a result, the regularization term dominates, driving $\mathbf{d}_t \to 0$ for all tokens and deactivating Global Attention (further discussion in Section A). To address this, we invoke Global Attention for all tokens (regardless of the Router's decision) with a small probability during training. Specifically, in each iteration, we sample a Bernoulli (with $p = 0.1$). If heads, Global Attention is applied to all tokens, and otherwise we follow Equation (5). This preserves the forward behavior for the masked tokens since the Global Attention output, $\mathbf{a}_t$, is always multiplied by $\mathbf{d}_t = 0$ (see Equation (6)), while ensuring gradients from the NTP loss reach the Router even for the masked tokens. Since Global Attention is invoked sparingly, *L2A* largely retains the efficiency benefits of conditional computation.

3.4. *L2A* Inference

Autoregressive inference consists of the prefill and decode stages. During prefill, Local Attention uses standard SWA kernels from FA-2, while Global Attention uses our sparse kernel to compute attention only for active queries. During decode, both modules use standard FA-2, with Global Attention skipped entirely for tokens where $\mathbf{d}_t = 0$.

KV Cache Bottleneck. At long context lengths, KV cache storage dominates memory (Saxena et al., 2024; Liu et al., 2024c). In *L2A*, keys and values must still be stored for all tokens, as future tokens may invoke Global Attention. Consequently, *L2A* and the Base LLM incur similar KV cache costs during decode. The additional KV storage from Local Attention is negligible ($< 3\%$ for a 4K window at 128K context). However, we leverage *L2A*'s sparse Global Attention usage to further reduce KV cache memory.

Sparsity-based Layer Pruning. We introduce a post-training pruning strategy that removes Global Attention module from layers with high sparsity (measured on held-out data), retaining only Local Attention module. Since Global Attention requires full-context KV storage while Local Attention uses a small window, this significantly reduces memory. In practice, nearly 50% of Global Attention layers can be pruned (see Figure 3), resulting in 50% KV cache reduction relative to the Base LLM with less than 1% performance drop. For detailed results and analysis, please refer to Section D.2.

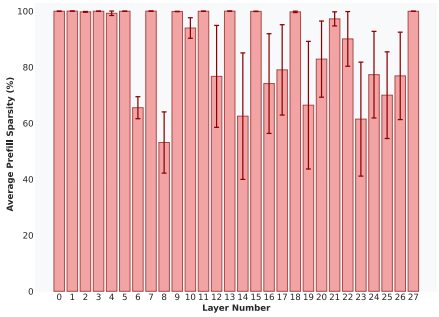

Figure 3. Sparsity levels across different layers of Qwen2.5-1.5B *L2A* model. Nearly 50% layers rely almost exclusively ($\geq 95\%$) on Local Attention.

4. Experiments

We evaluate *L2A* for long-context extension, and report long-context performance against baselines (Section 4.1), runtime speedups over FA-2 (Section 4.2), ablations and sparsity analysis (Section 4.3, Section 4.4).

Base LLMs and tasks. We use Qwen2.5 7B (Qwen et al., 2025) and Qwen3 8B as base models, and Qwen2.5 1.5B for ablations. All models have a pre-trained context length of 32K. Long-context performance is evaluated on HELMET (Yen et al., 2025), BABILong (Kuratov et al., 2024), and MRCR (Vodrahalli et al., 2024), covering diverse long-context tasks (details in Section F.2).

Baselines. We compare *L2A* with training-free methods (SelfExtend (Jin et al., 2024), DCA (An et al., 2024)), training-based sparse attention methods (S^2-Attn (Chen et al., 2024), NSA (Yuan et al., 2025)), and the inference-time method FlexPrefill (Lai et al., 2025). We use official implementations when available; for NSA, we use a widely adopted open-source version (Li, 2025) (see Section E). CLP-trained models serve as an upper bound, and the Base LLM evaluated at long context acts as a lower bound.

Implementation details. All methods are initialized from Base LLM and trained on a mixture of long documents. *L2A* is initialized as in Section 3.1, with all layers using SWA with a 4K window as Local Attention. Following Qwen (Yang et al., 2025), we increase the RoPE base frequency from 1M to 5M to train at 128K context. We freeze FFN layers and update only token-mixing layers and LayerNorm, which outperforms full-parameter training (Section E.1.3). FlexPrefill is applied to CLP models for faster prefill. Addi-

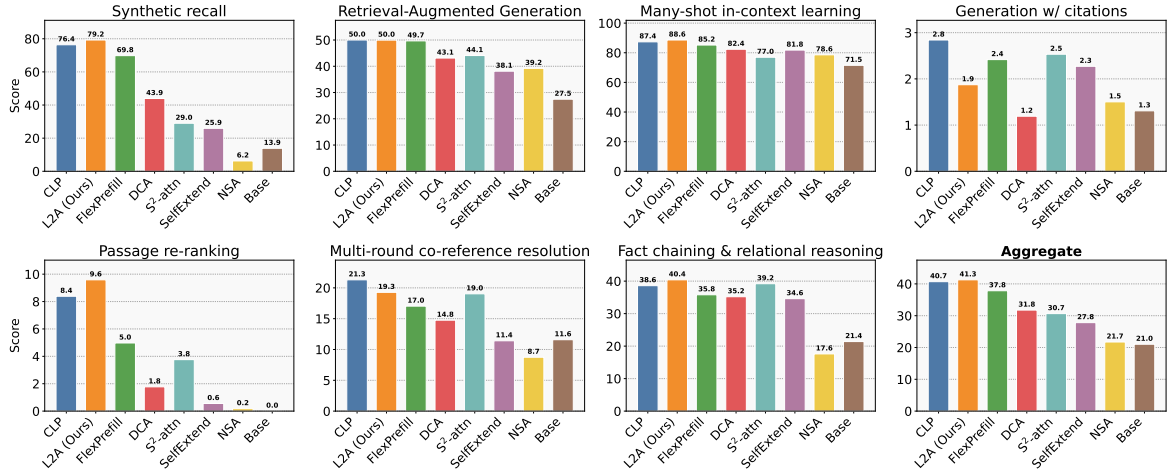

Figure 4. Task-wise performance comparison of *L2A* and baseline methods on Qwen 2.5 7B model. *L2A* remains close to CLP performance while outperforming baselines on the majority of tasks.

tional details are in Appendix F.

4.1. *L2A* Across Long & Short Context Tasks

We evaluate *L2A* on Qwen2.5 1.5B and 7B across context lengths from 8K to 128K. *L2A* achieves performance within 3% of CLP while consistently outperforming all training-based and training-free baselines (Figure 5). While most methods match Base LLM performance within the pretrained 32K context, they degrade significantly beyond it. In contrast, *L2A* maintains strong performance at longer contexts, highlighting the benefit of conditional long-term memory over fixed sparsity levels.

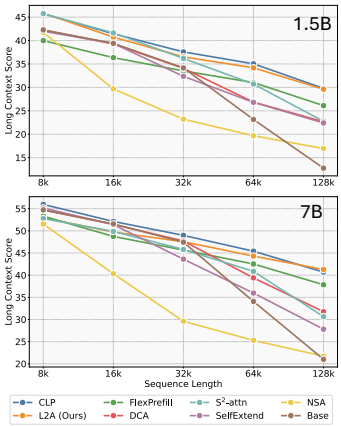

Figure 5. Aggregate long-context performance of Qwen models at 1.5B and 7B scales. *L2A* achieves comparable performance to CLP across several context lengths, while significantly outperforming prior baselines.

We report task-wise performance at 128K context, including Synthetic Recall, Retrieval-Augmented Generation, and In-Context Learning (Figure 4). *L2A* consistently outperforms all baselines and remains competitive with CLP (Tables 3, 4). Similar trends hold for Qwen3 8B, where *L2A* outperforms all baselines while remaining within 3% of CLP (see Table 5), demonstrating robustness across model families. Overall, *L2A* is the only method that matches CLP on long-context tasks while achieving higher training efficiency via sparse Global Attention.

To assess short-context performance, we also evaluate short-context benchmarks below 2K tokens. As shown in Table 1, *L2A* matches CLP across model scales, with differences within seed-level variability, indicating no degradation from long-context training.

4.2. Runtime Speedups from *L2A* Kernels

We evaluate the throughput of *L2A* kernel on NVIDIA H200 GPUs, comparing forward and backward passes against FlashAttention-2 (FA-2). As shown in Figure 8 (Section D.1), our kernel achieves 1.6-35× and 1.08-45 × speedups for forward and backward passes, respectively, across varying sparsity levels. These kernel-level gains translate to nearly 2× higher training throughput at 128K context as shown in Table 2.

During prefill, *L2A* achieves nearly 2× speedups in time-to-first-token (TTFT) compared to CLP. As shown in Figure 6, *L2A* is nearly Pareto-optimal, matching CLP performance while approaching the TTFT of FlexPrefill. Additional results across context lengths are provided in the Appendix (Figure 14, Figure 15). Notably, speedups increase with context length as Global Attention cost scales quadratically.

4.3. *L2A*'s Conditional Global Attention Patterns

We analyze sparsity induced by different long-context tasks. Sparsity reflects how often Global Attention is triggered,

Table 1. Zero-shot performance on standard short-context benchmarks for Qwen 2.5 models at different scales. *L2A* closely matches CLP, indicating that training at 128K context length does not degrade short-context performance

Scale	Method	BoolQ acc ↑	CommSenseQA acc ↑	PIQA acc_n ↑	Winogrande acc ↑	ARC-E acc_n ↑	ARC-C acc_n ↑	MMLU acc ↑	SWDE contains ↑	Avg
1.5B	CLP	**73.36**	**74.45**	75.95	**64.72**	**88.72**	**73.81**	**60.46**	86.50	**74.74**
	L2A	71.44	73.46	**76.01**	64.09	88.59	73.12	60.03	**87.31**	74.26
7B	CLP	**82.57**	**84.52**	79.43	**76.01**	96.09	**88.65**	**73.39**	91.00	**83.95**
	L2A	80.15	83.70	**80.14**	75.69	**96.21**	88.14	72.76	**92.17**	83.62

Table 2. L2A achieves higher training throughput than the Base LLM. Throughput (tokens/GPU/s) and speedup are reported for three models on NVIDIA H200 GPUs, using batch sizes of 6M tokens (Qwen2.5 7B, Qwen3 8B) and 4M tokens (Qwen2.5 1.5B).

Model	Method	Throughput	Speedup Ratio
Qwen2.5-1.5B	*L2A* (Ours)	3058.70	2.02
	CLP	1517.74	1.00
Qwen2.5-7B	*L2A* (Ours)	1383.85	2.15
	CLP	642.67	1.00
Qwen3-8B	*L2A* (Ours)	410.42	1.82
	CLP	226.04	1.00

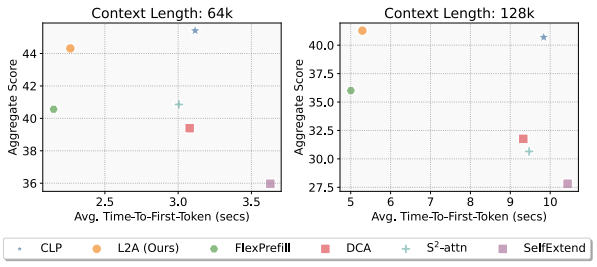

Figure 6. Time-to-first-token (TTFT) for Qwen 2.5 7B averaged over different tasks from our long-context benchmark suites. *L2A* achieves similar aggregate long-context performance as CLP, while being $\sim 2\times$ faster.

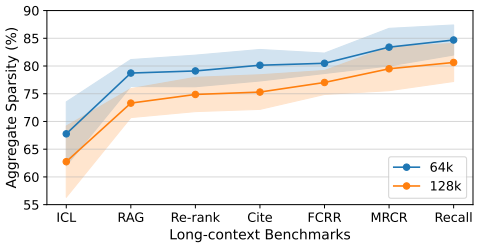

Figure 7. Average sparsity for each task at 64k and 128k context lengths, aggregated across Qwen2.5-1.5B, Qwen2.5-7B, and Qwen3-8B models. Tasks are ordered by increasing sparsity. Refer to Section F.2 for task details.

i.e., when local context is insufficient. As shown in Figure 7, sparsity varies based on how long-range dependencies are distributed across tokens and generally decreases with increasing context length. Tasks such as Recall and MRCR (Vodrahalli et al., 2024) concentrate long-range dependencies at a small number of specific positions, making local context sufficient for most tokens. Retrieval and reasoning tasks like RAG and Passage Re-ranking require global access at multiple tokens for relevance comparison and aggregation, leading to intermediate sparsity. In contrast, In-Context Learning frequently requires comparisons across demonstrations, leading to lower sparsity.

4.4. Ablation Study

Does the Router in *L2A* actually use context? We compare against a context-free baseline that independently triggers Global Attention at each token with Bernoulli probability p, which we refer to as *Context-free Sparse*. We consider $p \in \{0.1, 0.2, 0.5\}$, corresponding to 50-90% sparsity levels. Across all settings, *L2A* outperforms this baseline, showing that the Router decisions depend on local context to decide when to invoke Global Attention. For example, on Qwen2.5 1.5B, *L2A* achieves 80% average sparsity across all tasks while improving performance by $\sim$30% over the strongest baseline ($p = 0.5$, see Figure 11).

What is the effect of window size in Local Attention? We vary the SWA window size from 0 to 4K. Performance remains comparable across settings, with the Router compensating for smaller windows by reducing sparsity (Figure 16). As the short-term local context shrinks, *L2A* increasingly relies on long-term global memory. In the extreme SWA-0 case, sparsity drops to 0% (as shown in Figure 17). While accuracy is maintained, efficiency degrades, with $2\times$ higher TTFT compared to SWA-4K configuration (Figure 18). Additional ablations are presented in Section E.1.

5. Discussion

This work shows that conditional long-term memory access is an effective approach for efficient long-context modeling. A natural extension is to replace the Local Attention module with State-Space Models (SSMs) (Gu & Dao, 2024; Yang et al.; Peng et al., 2025), which offer linear-time complexity and maintain a compressed summary of past context. However, integrating SSMs into pretrained LLMs typically increases training cost and design complexity (Wang et al., 2024a). We leave this direction for future work.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Amouyal, S. J., Wolfson, T., Rubin, O., Yoran, O., Herzig, J., and Berant, J. Qampari: An open-domain question answering benchmark for questions with many answers from multiple paragraphs, 2023. URL https://arxiv.org/abs/2205.12665.

An, C., Huang, F., Zhang, J., Gong, S., Qiu, X., Zhou, C., and Kong, L. Training-free long-context scaling of large language models. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Asai, A., Wu, Z., Wang, Y., Sil, A., and Hajishirzi, H. Self-rag: Learning to retrieve, generate, and critique through self-reflection. 2024.

Bae, S., Kim, Y., Bayat, R., Kim, S., Ha, J., Schuster, T., Fisch, A., Harutyunyan, H., Ji, Z., Courville, A., and Yun, S.-Y. Mixture-of-recursions: Learning dynamic recursive depths for adaptive token-level computation, 2025. URL https://arxiv.org/abs/2507.10524.

Bajaj, P., Campos, D., Craswell, N., Deng, L., Gao, J., Liu, X., Majumder, R., McNamara, A., Mitra, B., Nguyen, T., Rosenberg, M., Song, X., Stoica, A., Tiwary, S., and Wang, T. Ms marco: A human generated machine reading comprehension dataset, 2018. URL https://arxiv.org/abs/1611.09268.

Bengio, Y., Léonard, N., and Courville, A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Bohnet, B., Tran, V. Q., Verga, P., Aharoni, R., Andor, D., Soares, L. B., Ciaramita, M., Eisenstein, J., Ganchev, K., Herzig, J., Hui, K., Kwiatkowski, T., Ma, J., Ni, J., Saralegui, L. S., Schuster, T., Cohen, W. W., Collins, M., Das, D., Metzler, D., Petrov, S., and Webster, K. Attributed question answering: Evaluation and modeling for attributed large language models, 2023. URL https://arxiv.org/abs/2212.08037.

Borgeaud, S., Mensch, A., Hoffmann, J., Cai, T., Rutherford, E., Millican, K., van den Driessche, G., Lespiau, J., Damoc, B., Clark, A., de Las Casas, D., Guy, A., Menick, J., Ring, R., Hennigan, T., Huang, S., Maggiore, L., Jones, C., Cassirer, A., Brock, A., Paganini, M., Irving, G., Vinyals, O., Osindero, S., Simonyan, K., Rae, J. W., Elsen, E., and Sifre, L. Improving language models by retrieving from trillions of tokens. *CoRR*, abs/2112.04426, 2021. URL https://arxiv.org/abs/2112.04426.

Chen, S., Wong, S., Chen, L., and Tian, Y. Extending context window of large language models via positional interpolation, 2023. URL https://arxiv.org/abs/2306.15595.

Chen, Y., Qian, S., Tang, H., Lai, X., Liu, Z., Han, S., and Jia, J. Longlora: Efficient fine-tuning of long-context large language models. In *The International Conference on Learning Representations (ICLR)*, 2024.

Chhikara, P., Khant, D., Aryan, S., Singh, T., and Yadav, D. Mem0: Building production-ready ai agents with scalable long-term memory, 2025. URL https://arxiv.org/abs/2504.19413.

Craswell, N., Mitra, B., Yilmaz, E., Campos, D., Lin, J., Voorhees, E. M., and Soboroff, I. Overview of the trec 2022 deep learning track, 2025. URL https://arxiv.org/abs/2507.10865.

Dao, T. Flashattention-2: Faster attention with better parallelism and work partitioning, 2023. URL https://arxiv.org/abs/2307.08691.

Ding, Y., Zhang, L. L., Zhang, C., Xu, Y., Shang, N., Xu, J., Yang, F., and Yang, M. Longrope: extending llm context window beyond 2 million tokens. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Gao, T., Yen, H., Yu, J., and Chen, D. Enabling large language models to generate text with citations, 2023. URL https://arxiv.org/abs/2305.14627.

Gao, T., Wettig, A., Yen, H., and Chen, D. How to train long-context language models (effectively). In *ACL*, 2025.

Geva, M., Schuster, R., Berant, J., and Levy, O. Transformer feed-forward layers are key-value memories, 2021. URL https://arxiv.org/abs/2012.14913.

Gray, S., Radford, A., and Kingma, D. P. Gpu kernels for block-sparse weights. *arXiv preprint arXiv:1711.09224*, 3(2):2, 2017.

Gu, A. and Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=tEYskw1VY2.

Han, C., Wang, Q., Peng, H., Xiong, W., Chen, Y., Ji, H., and Wang, S. LM-infinite: Zero-shot extreme length generalization for large language models. In Duh, K., Gomez,

H., and Bethard, S. (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3991–4008, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.222. URL https://aclanthology.org/2024.naacl-long.222/.

Heakl, A., Gubri, M., Khan, S., Yun, S., and Oh, S. J. Dr.llm: Dynamic layer routing in llms, 2025. URL https://arxiv.org/abs/2510.12773.

Jeong, S., Baek, J., Cho, S., Hwang, S. J., and Park, J. C. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity, 2024. URL https://arxiv.org/abs/2403.14403.

Jiang, H., Li, Y., Zhang, C., Wu, Q., Luo, X., Ahn, S., Han, Z., Abdi, A. H., Li, D., Lin, C.-Y., Yang, Y., and Qiu, L. Minference 1.0: Accelerating pre-filling for long-context llms via dynamic sparse attention, 2024a. URL https://arxiv.org/abs/2407.02490.

Jiang, Y., Wang, H., Xie, L., Zhao, H., Zhang, C., Qian, H., and Lui, J. C. D-llm: A token adaptive computing resource allocation strategy for large language models. In Globerson, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J., and Zhang, C. (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 1725–1749. Curran Associates, Inc., 2024b. doi: 10.52202/079017-0055.

Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., and Narasimhan, K. Swe-bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*, 2023.

Jin, H., Han, X., Yang, J., Jiang, Z., Liu, Z., Chang, C.-Y., Chen, H., and Hu, X. Llm maybe longlm: Selfextend llm context window without tuning. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Kamradt, G. Needle in a haystack - pressure testing llms. 2024. URL https://github.com/gkamradt/LLMTest_NeedleInAHaystack.

Kuratov, Y., Bulatov, A., Anokhin, P., Rodkin, I., Sorokin, D., Sorokin, A., and Burtsev, M. Babilong: Testing the limits of llms with long context reasoning-in-a-haystack. In Globerson, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J., and Zhang, C. (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 106519–106554. Curran Associates, Inc., 2024.

Lai, X., Lu, J., Luo, Y., Ma, Y., and Zhou, X. Flexprefill: A context-aware sparse attention mechanism for efficient long-sequence inference. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=OfjIlbelrT.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., and Kiela, D. Retrieval-augmented generation for knowledge-intensive nlp tasks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 9459–9474. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf.

Li, S., Min, Y., Yie, H., Kim, H., Ahn, S., Sim, J., Lee, C.-H., and Kim, J. Accelerating sparse matrix-matrix multiplication on gpus with processing near hbms. *arXiv preprint arXiv:2512.12036*, 2025.

Li, X. Open-source nsa implementation, 2025. URL https://github.com/XunhaoLai/native-sparse-attention-triton.

Li, Z., Li, C., Zhang, M., Mei, Q., and Bendersky, M. Retrieval augmented generation or long-context LLMs? a comprehensive study and hybrid approach. In Dernoncourt, F., Preoţiuc-Pietro, D., and Shimorina, A. (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 881–893, Miami, Florida, US, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-industry.66. URL https://aclanthology.org/2024.emnlp-industry.66/.

Liu, H., Zaharia, M., and Abbeel, P. Ringattention with blockwise transformers for near-infinite context. In *International Conference on Learning Representations*, volume 2024, pp. 3992–4008, 2024a.

Liu, X., Yan, H., Zhang, S., An, C., Qiu, X., and Lin, D. Scaling laws of rope-based extrapolation, 2024b. URL https://arxiv.org/abs/2310.05209.

Liu, Z., Yuan, J., Jin, H., Zhong, S., Xu, Z., Braverman, V., Chen, B., and Hu, X. Kivi: A tuning-free asymmetric 2bit quantization for kv cache. *arXiv preprint arXiv:2402.02750*, 2024c.

Lu, Y., Yan, J. N., Yang, S., Chiu, J. T., Ren, S., Yuan, F., Zhao, W., Wu, Z., and Rush, A. M. A controlled study on long context extension and generalization in llms, 2024. URL https://arxiv.org/abs/2409.12181.

Maharana, A., Lee, D.-H., Tulyakov, S., Bansal, M., Barbieri, F., and Fang, Y. Evaluating very long-term conversational memory of llm agents. *arXiv preprint arXiv:2402.17753*, 2024.

Mohtashami, A. and Jaggi, M. Random-access infinite context length for transformers. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Munkhdalai, T., Faruqui, M., and Gopal, S. Leave no context behind: Efficient infinite context transformers with infini-attention. *CoRR*, 2024.

Nunez, E., Zancato, L., Bowman, B., Golatkar, A., Xia, W., and Soatto, S. Expansion span: Combining fading memory and retrieval in hybrid state space models, 2025. URL https://arxiv.org/abs/2412.13328.

Peng, B., Quesnelle, J., Fan, H., and Shippole, E. Yarn: Efficient context window extension of large language models, 2023. URL https://arxiv.org/abs/2309.00071.

Peng, L., Chattopadhyay, A., Zancato, L., Nunez, E., Xia, W., and Soatto, S. Gated kalmanet: A fading memory layer through test-time ridge regression. *arXiv preprint arXiv:2511.21016*, 2025.

Qwen, Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., Lin, H., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Lin, J., Dang, K., Lu, K., Bao, K., Yang, K., Yu, L., Li, M., Xue, M., Zhang, P., Zhu, Q., Men, R., Lin, R., Li, T., Tang, T., Xia, T., Ren, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Wan, Y., Liu, Y., Cui, Z., Zhang, Z., and Qiu, Z. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Raposo, D., Ritter, S., Richards, B., Lillicrap, T., Humphreys, P. C., and Santoro, A. Mixture-of-depths: Dynamically allocating compute in transformer-based language models, 2024. URL https://arxiv.org/abs/2404.02258.

Saxena, U., Saha, G., Choudhary, S., and Roy, K. Eigen attention: Attention in low-rank space for kv cache compression. *arXiv preprint arXiv:2408.05646*, 2024.

Sharma, A., Najafi, S., Farinneya, P., Jamialahmadi, B., Tahaei, M. S., Fan, Y., Rezagholizadeh, M., Chen, B., and Jafari, A. Dtrnet: Dynamic token routing network to reduce quadratic costs in transformers, 2025. URL https://arxiv.org/abs/2509.00925.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q. V., Hinton, G. E., and Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *CoRR*, abs/1701.06538, 2017. URL http://arxiv.org/abs/1701.06538.

Stelmakh, I., Luan, Y., Dhingra, B., and Chang, M.-W. Asqa: Factoid questions meet long-form answers, 2023. URL https://arxiv.org/abs/2204.06092.

Stewart, L., Trager, M., Gonugondla, S. K., and Soatto, S. The n-grammys: Accelerating autoregressive inference with learning-free batched speculation. *arXiv preprint arXiv:2411.03786*, 2024.

Su, W., Tang, Y., Ai, Q., Wu, Z., and Liu, Y. DRAGIN: Dynamic retrieval augmented generation based on the real-time information needs of large language models. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12991–13013, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.702. URL https://aclanthology.org/2024.acl-long.702/.

Vodrahalli, K., Ontanon, S., Tripuraneni, N., Xu, K., Jain, S., Shivanna, R., Hui, J., Dikkala, N., Kazemi, M., Fatemi, B., Anil, R., Dyer, E., Shakeri, S., Vij, R., Mehta, H., Ramasesh, V., Le, Q., Chi, E., Lu, Y., Firat, O., Lazaridou, A., Lespiau, J.-B., Attaluri, N., and Olszewska, K. Michelangelo: Long context evaluations beyond haystacks via latent structure queries, 2024. URL https://arxiv.org/abs/2409.12640.

Wang, J., Paliotta, D., May, A., Rush, A., and Dao, T. The mamba in the llama: Distilling and accelerating hybrid models. *Advances in Neural Information Processing Systems*, 37:62432–62457, 2024a.

Wang, L., Gao, H., Zhao, C., Sun, X., and Dai, D. Auxiliary-loss-free load balancing strategy for mixture-of-experts. *arXiv preprint arXiv:2408.15664*, 2024b.

Xia, H., Li, Y., Leong, C. T., Wang, W., and Li, W. Tokenskip: Controllable chain-of-thought compression in llms, 2025. URL https://arxiv.org/abs/2502.12067.

Xiao, C., Zhang, P., Han, X., Xiao, G., Lin, Y., Zhang, Z., Liu, Z., and Sun, M. InfLLM: Training-free long-context extrapolation for LLMs with an efficient context memory. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024a. URL https://openreview.net/forum?id=bTHFrqhASY.

Xiao, G., Tian, Y., Chen, B., Han, S., and Lewis, M. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview.net/forum?id=NG7sS51zVF.

Yan, S.-Q., Gu, J.-C., Zhu, Y., and Ling, Z.-H. Corrective retrieval augmented generation, 2024. URL https://arxiv.org/abs/2401.15884.

Yang, A., Yu, B., Li, C., Liu, D., Huang, F., Huang, H., Jiang, J., Tu, J., Zhang, J., Zhou, J., et al. Qwen2. 5-1m technical report. *arXiv preprint arXiv:2501.15383*, 2025.

Yang, S., Kautz, J., and Hatamizadeh, A. Gated delta networks: Improving mamba2 with delta rule. In *The Thirteenth International Conference on Learning Representations*.

Yen, H., Gao, T., Hou, M., Ding, K., Fleischer, D., Izsak, P., Wasserblat, M., and Chen, D. Helmet: How to evaluate long-context language models effectively and thoroughly. In *International Conference on Learning Representations (ICLR)*, 2025.

Yuan, J., Gao, H., Dai, D., Luo, J., Zhao, L., Zhang, Z., Xie, Z., Wei, Y., Wang, L., Xiao, Z., Wang, Y., Ruan, C., Zhang, M., Liang, W., and Zeng, W. Native sparse attention: Hardware-aligned and natively trainable sparse attention. In Che, W., Nabende, J., Shutova, E., and Pilehvar, M. T. (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23078–23097, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1126. URL https://aclanthology.org/2025.acl-long.1126/.

Zancato, L., Seshadri, A., Dukler, Y., Golatkar, A., Shen, Y., Bowman, B., Trager, M., Achille, A., and Soatto, S. B'mojo: Hybrid state space realizations of foundation models with eidetic and fading memory. In Globerson, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J., and Zhang, C. (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 130433–130462. Curran Associates, Inc., 2024.

Zhang, Z., Sheng, Y., Zhou, T., Chen, T., Zheng, L., Cai, R., Song, Z., Tian, Y., Ré, C., Barrett, C., et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36:34661–34710, 2023.

Zhou, Y., Liu, H., Chen, Z., Tian, Y., and Chen, B. Gsm-infinite: How do your llms behave over infinitely increasing context length and reasoning complexity?, 2025. URL https://arxiv.org/abs/2502.05252.

A. Gradient Analysis of the Routing Mechanism

We repeat the equations presented in Section 3.3 here before diving into understanding the routing decision behavior during training. Formally, given an input sequence $(\mathbf{x}_1, \ldots, \mathbf{x}_t)$:

$$\mathbf{s}_t = \text{LocalAttn}(\mathbf{x}_t, \mathbf{x}_{t-1}, \ldots, \mathbf{x}_{t-k}) \;\; \forall \;\; t = 1, \ldots, n \tag{9}$$

$$\hat{\mathbf{d}}_t = \sigma(\mathbf{W}\mathbf{s}_t), \quad \mathbf{d}_t = \begin{cases} 1, & \text{if } \hat{\mathbf{d}}_t \geq 0.5, \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

$$\mathbf{a}_t = \text{GlobalAttn}(\mathbf{s}_t, \mathbf{s}_{t-1}, \ldots, \mathbf{s}_1), \tag{11}$$

The contribution of global attention to the residual stream can be written as

$$\mathbf{o}_t = \mathbf{d}_t . \mathbf{a}_t + \mathbf{s}_t, \tag{12}$$

where this term is added to the residual connection following standard Transformer conventions. Thus, when $\mathbf{d}_t = 0$, the token retains its local representation $\mathbf{s}_t$.

The overall training objective is defined as:

$$\mathcal{L} = \mathcal{L}_{\text{NTP}} + \frac{\lambda}{nL} \sum_{l=1}^{L} \sum_{t=1}^{n} \hat{\mathbf{d}}_{l,t}^2. \tag{13}$$

For clarity, gradients for a single layer are derived, and the layer index l is omitted in what follows. The gradient $\frac{\partial \mathcal{L}}{\partial \mathbf{W}}$ determines how the routing function learns to activate global attention during training. Aggregating contributions across all tokens, we have

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}} = \sum_{t=1}^{n} \frac{\partial \mathcal{L}}{\partial \hat{\mathbf{d}}_t} \frac{\partial \hat{\mathbf{d}}_t}{\partial \mathbf{W}} \tag{14}$$

$$\frac{\partial \hat{\mathbf{d}}_t}{\partial \mathbf{W}} = \hat{\mathbf{d}}_t (1 - \hat{\mathbf{d}}_t) \mathbf{s}_t^\top \tag{15}$$

The gradient of the loss with respect to $\hat{\mathbf{d}}_t$ is

$$\frac{\partial \mathcal{L}}{\partial \hat{\mathbf{d}}_t} = \frac{\partial \mathcal{L}_{\text{NTP}}}{\partial \hat{\mathbf{d}}_t} + \frac{\partial}{\partial \hat{\mathbf{d}}_t} \left(\frac{\lambda}{L} \sum_{\tau=1}^{n} \hat{\mathbf{d}}_\tau^2 \right) \approx \left(\frac{\partial \mathcal{L}_{\text{NTP}}}{\partial \mathbf{o}_t} \right)^T \frac{\partial \mathbf{o}_t}{\partial \mathbf{d}_t} + \frac{2\lambda}{L} \hat{\mathbf{d}}_t = \left(\frac{\partial \mathcal{L}_{\text{NTP}}}{\partial \mathbf{o}_t} \right)^T . \mathbf{a}_t + \frac{2\lambda}{L} \hat{\mathbf{d}}_t. \tag{16}$$

The approximation ($\approx$) in the above equation reflects the use of Straight Through Estimator (STE) for backpropagation through discrete routing decisions as discussed in Section 3.3. Specifically, $\mathbf{d}_t$ is a discrete routing decision in the forward pass, while the gradients are computed with respect to the continuous routing score $\hat{\mathbf{d}}_t$ during training.

Combining equations 15 and 16:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}} = \sum_{t=1}^{n} \frac{\partial \mathcal{L}}{\partial \hat{\mathbf{d}}_t} \frac{\partial \hat{\mathbf{d}}_t}{\partial \mathbf{W}} = \sum_{t=1}^{n} \left[\frac{\partial \mathcal{L}_{\text{NTP}}}{\partial \mathbf{o}_t} . \mathbf{a}_t \hat{\mathbf{d}}_t (1 - \hat{\mathbf{d}}_t) \, \mathbf{s}_t^\top + \frac{2\lambda}{L} \hat{\mathbf{d}}_t^2 (1 - \hat{\mathbf{d}}_t) \, \mathbf{s}_t^\top \right] \tag{17}$$

To interpret the effect of the gradient update on the routing decision, consider the gate logit $\mathbf{z}_t := \mathbf{W}\mathbf{s}_t$, which directly determines the routing score $\hat{\mathbf{d}}_t = \sigma(\mathbf{z}_t)$. Under gradient descent, $\mathbf{W} \leftarrow \mathbf{W} - \eta \frac{\partial \mathcal{L}}{\partial \mathbf{W}}$, the induced change in $\mathbf{z}_t$ is

$$\Delta \mathbf{z}_t = -\eta \left(\frac{\partial \mathcal{L}}{\partial \mathbf{W}} \right) \mathbf{s}_t = -\eta \left[\left(\frac{\partial \mathcal{L}_{\text{NTP}}}{\partial \mathbf{o}_t} . \mathbf{a}_t \hat{\mathbf{d}}_t (1 - \hat{\mathbf{d}}_t) \right) \|\mathbf{s}_t\|^2 + \left(\frac{2\lambda}{L} \hat{\mathbf{d}}_t^2 (1 - \hat{\mathbf{d}}_t) \right) \|\mathbf{s}_t\|^2 \right] \tag{18}$$

The first term arises from the next token prediction loss (NTP) and provides a context-dependent learning signal that can increase or decrease $\mathbf{z}_t$, enabling the model to selectively activate global attention only when it improves the language modeling objective. The second term arises from the sparsity regularization and is elementwise non-negative, thus driving $\mathbf{z}_t$ toward smaller values, encouraging $\hat{\mathbf{d}}_t \to 0$ and promoting sparse activation of global attention. For masked tokens (i.e. $\mathbf{a}_t = 0$), the NTP term in Equation (18) vanishes, and the regularization term alone favors solutions with $\mathbf{d}_t \to 0$, causing the model to never invoke Global Attention.We refer to this behavior as router collapse and discuss it in Section 3.3, along with our strategy for invoking Global Attention across all tokens with Bernoulli sampling (probability ~ 0.1).

B. Extended Related Works

B.1. Context Length Extrapolation

To mitigate the computational and memory cost of training LLMs on long sequences, several context length extrapolation methods have been proposed (An et al., 2024; Xiao et al., 2024a; Peng et al., 2023; Ding et al., 2024). These approaches extend a model's native context with minimal training by rescaling or reusing positional indices learned during pretraining (Peng et al., 2023; Chen et al., 2023; Liu et al., 2024b), and can be applied directly to pretrained models or used in conjunction with long-context training. For example, Position Interpolation (Chen et al., 2023) linearly scales rotary position embedding (RoPE) indices, while YaRN (Peng et al., 2023) applies uneven frequency interpolation to better preserve high-frequency components.

Another line of work compresses the input using windowed or chunk-based attention (Xiao et al., 2024b; Han et al., 2024), which restricts access to distant context. While hierarchical schemes like Dual-Chunk Attention (DCA) partially reintroduce global information (An et al., 2024; Jin et al., 2024), downstream performance often remains limited, motivating training-based approaches for improved long-context modeling.

B.2. Dynamic Routing in LLMs

Mixture-of-Experts (MoE) architectures introduced conditional computation as a scalable approach for improving the efficiency of LLM training and inference (Shazeer et al., 2017). MoE consists of multiple expert networks, typically lightweight MLPs, selectively activated for each input through a trainable gating mechanism. While effective at scale, this paradigm generally requires training the entire model from scratch. Several recent works propose training lightweight routing modules on top of frozen backbones, enabling coarse-grained conditional execution such as skipping entire layers based on input difficulty (Heakl et al., 2025).

To achieve more adaptive and fine-grained conditional computation, token-level routing has emerged as a viable alternative (Raposo et al., 2024; Jiang et al., 2024b; Sharma et al., 2025). By operating at the token level, these approaches aim to focus computation on informative tokens while allowing simpler tokens to follow cheaper execution paths. Mixture-of-Depths (MoD) (Raposo et al., 2024) selects a fixed top-k subset of tokens to process at each decoder block using a learned, layer-wise router, while Mixture-of-Recursions (MoR) (Bae et al., 2025) extends this idea to recursive transformers by dynamically determining the number of recursion steps per token. However, both approaches rely on static per-layer compute budgets that route a fixed number of tokens regardless of input complexity or sequence length. In addition, their top-k routing decisions are inherently non-causal. Although auxiliary routers or losses are used to predict routing behavior during inference, they introduce a mismatch between training and inference dynamics, leading to degraded performance. D-LLM (Jiang et al., 2024b) further improves upon deterministic top-k routing by employing Gumbel-Softmax–based routing modules for each layer. Despite more flexible routing, D-LLM, similar to prior token-routing approaches, routes tokens around entire decoder blocks, causing skipped tokens to bypass both attention and MLP layers. While this design improves computational efficiency, it disrupts token interactions and limits representation capacity. These limitations are expected to become more pronounced in long-context settings, where preserving fine-grained token–token interactions over extended sequences is crucial, suggesting that block-level routing is a suboptimal design choice.

B.3. External Memory-Based Approaches

To address the limitations of fixed context windows, several approaches augment pretrained LLMs with external memory mechanisms to retrieve and incorporate relevant information into the input (Lewis et al., 2020; Chhikara et al., 2025; Borgeaud et al., 2021; Su et al., 2024). Early work in this direction focuses on Retrieval-Augmented Generation (RAG), where the model input is augmented with passages retrieved from an external knowledge corpus (Lewis et al., 2020;

Algorithm 1 *L2A* Kernel Forward Pass

Require: Matrices $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{n \times d}$ and non-zero query indices $\mathbf{Q}_{idx} \in \mathbb{R}^{n}$

Consolidate active (non-zero) queries into $\mathbf{Q}_c \in \mathbb{R}^{n_c \times d}$ using indices $\mathbf{Q}_{idx}$.
$\mathbf{Q}_c$ tiled into $T_q = n_c/B_q$ blocks ($\mathbf{Q}_{c,1}, \ldots, \mathbf{Q}_{c,T_q}$ of size $B_q \times d$), $\mathbf{K}, \mathbf{V}$ into $T_k = n/B_k$ blocks ($\mathbf{K}_1, \ldots, \mathbf{K}_{T_k}$, $\mathbf{V}_1, \ldots, \mathbf{V}_{T_k}$, of size $B_k \times d$), and $\mathbf{Q}_{idx}$ into $T_q = n/B_q$ blocks ($\mathbf{Q}_{idx,1}, \ldots, \mathbf{Q}_{idx,T_q}$ of size B_q)
Initialize $\mathbf{O} \leftarrow \mathbf{0}_{n \times d}$. Partition $\mathbf{O}_c \in \mathbb{R}^{n_c \times d}$ into T_q tiles ($\mathbf{O}_{c,i}, \ldots, \mathbf{O}_{c,T_q}$ of size $B_q \times d$), and divide the logsumexp L_c into T_q blocks ($L_{c,i}, \ldots, L_{c,T_q}$ of size B_q).
for $1 \le i \le T_q$ **do**
  Load $\mathbf{Q}_{c,i}$ and $\mathbf{Q}_{idx,i}$ tiles from HBM to on-chip SRAM.
  $max\_k\_blocks = max(\mathbf{Q}_{idx,i})/B_k$
  Initialize $\mathbf{O}_{c,i}^{(0)} = (0)_{B_q \times d} \in \mathbb{R}^{B_q \times d}, \ell_{c,i}^{(0)} = (0)_{B_q} \in \mathbb{R}^{B_q}, m_{c,i}^{(0)} = (-\infty)_{B_q} \in \mathbb{R}^{B_q}$.
  for $1 \le j \le max\_k\_blocks$ **do**
    Load $\mathbf{K}_j, \mathbf{V}_j$ tiles from HBM to on-chip SRAM with ids $\mathbf{K}_{idx}$.
    $\mathbf{S}_i^{(j)} = (\mathbf{Q}_{c,i}\mathbf{K}_j^T)/\sqrt{d} \in \mathbb{R}^{B_q \times B_k}$ with element-wise causal mask $\mathbf{Q}_{idx,i} >= \mathbf{K}_{idx}$.
    $m_{c,i}^{(j)} = \max(m_{c,i}^{(j-1)}, \text{rowmax}(\mathbf{S}_i^{(j)})) \in \mathbb{R}^{B_q}$
    $\tilde{\mathbf{P}}_i^{(j)} = \exp(\mathbf{S}_i^{(j)} - m_{c,i}^{(j)}) \in \mathbb{R}^{B_q \times B_k}$
    $\ell_{c,i}^{(j)} = e^{m_{c,i}^{(j-1)} - m_{c,i}^{(j)}} \ell_{c,i}^{(j-1)} + \text{rowsum}(\tilde{\mathbf{P}}_i^{(j)}) \in \mathbb{R}^{B_q}$.
    $\mathbf{O}_{c,i}^{(j)} = \text{diag}(e^{m_{c,i}^{(j-1)} - m_{c,i}^{(j)}})^{-1} \mathbf{O}_{c,i}^{(j-1)} + \tilde{\mathbf{P}}_i^{(j)} \mathbf{V}_j$.
  end for
  $\mathbf{O}_{c,i} = \text{diag}(\ell_{c,i}^{(T_q)})^{-1} \mathbf{O}_{c,i}^{(T_q)}$.
  $L_{c,i} = m_{c,i}^{(T_q)} + \log(\ell_{c,i}^{(T_q)})$.
  Write $\mathbf{O}_{c,i}$ and $L_{c,i}$ to HBM as the i-th tiles of $\mathbf{O}_c$ and L_c, respectively.
end for
Save $\mathbf{Q}_c, \mathbf{K}, \mathbf{V}, \mathbf{O}_c, L_c, \mathbf{Q}_{idx}$ for backward.
Scatter $\mathbf{O}_c$ to $\mathbf{O}$ using $\mathbf{Q}_{idx}$ (i.e., $\mathbf{O}[\mathbf{Q}_{idx}] \leftarrow \mathbf{O}_c$).
return $\mathbf{O}$ and L_c

Borgeaud et al., 2021). RAG provides an efficient complement to long-context inference, but its effectiveness is contingent upon the quality and relevance of the retrieved information (Asai et al., 2024; Yan et al., 2024). This can be mitigated through nearest-neighbor based retrieval (Borgeaud et al., 2021), self-corrective and adaptive retrieval strategies (Yan et al., 2024; Jeong et al., 2024) and context-based graphical external memory representations (Chhikara et al., 2025). Despite these advances, external memory-based approaches often lag behind models that directly operate over long contexts (Li et al., 2024; Zhou et al., 2025). This gap is commonly attributed to the limited ability of such approaches to discern complex and implicit dependencies and to attend to relevant context. In contrast, our approach treats long-term memory access as a native operation within the model, enabling conditional global attention to perform context-based retrieval directly over the input sequence.

C. *L2A* Kernel Forward Pass

Algorithm 1 outlines the forward pass of the *L2A* kernel, which extends FlashAttention-2 to support token-level conditional Global Attention. Active queries are first compacted into a contiguous buffer $\mathbf{Q}_c$ using the index mapping $\mathbf{Q}_{idx}$, enabling efficient tiled computation. The kernel processes $\mathbf{Q}_c$, $\mathbf{K}$, and $\mathbf{V}$ in blocks streamed from high-bandwidth memory into on-chip SRAM, and for each query tile iterates only over key–value tiles that lie within its causal range as determined by $\mathbf{Q}_{idx}$. Attention scores are accumulated using an online log-sum-exp scheme to ensure numerical stability without materializing the full attention matrix. After processing all relevant tiles, the compact outputs $\mathbf{O}_c$ and normalization statistics L_c are written back to memory and scattered to reconstruct the full output $\mathbf{O}$. This design preserves the efficiency and numerical stability of FA-2 while avoiding unnecessary computation for inactive queries.

D. Additional Results

D.1. *L2A* Kernel Speedups

We benchmark the forward and backward pass of *L2A* kernel on NVIDIA H200 GPUs and compare it against FlashAttention-2. As shown in Figure 8, our custom kernel achieves 1.6-35× and 1.08-45 × speedups for the forward and the backward pass across varying sparsity levels, where sparsity denotes the fraction of queries for which Global Attention is skipped.

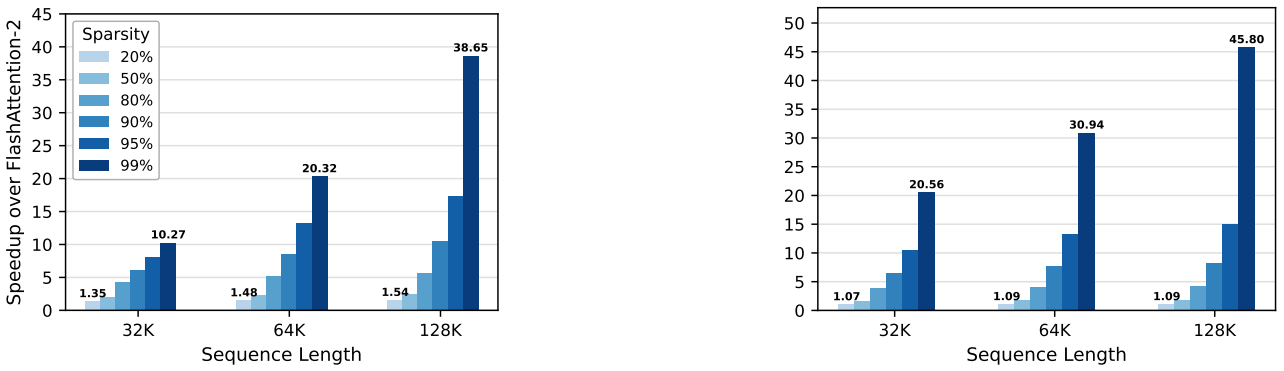

Figure 8. L2A kernel speedups over FlashAttention-2 for the forward (left) and backward (right) pass across varying sparsity levels and sequence lengths. Qwen-2.5 7B configuration is used (28 attention heads and a head dimension of 128).

D.2. Reducing KV Cache via Sparsity

We evaluate the sparsity-based layer pruning strategy introduced in Section 3.4. In Qwen 2.5 1.5B, nearly 50% of layers exhibit ≥95% sparsity (see Figure 3). Pruning Global Attention from these layers removes 15 out of 28 modules, yielding ∼50% KV cache memory reduction with negligible performance loss (Figure 9). This enables larger batch sizes and improves decoding throughput compared to CLP.

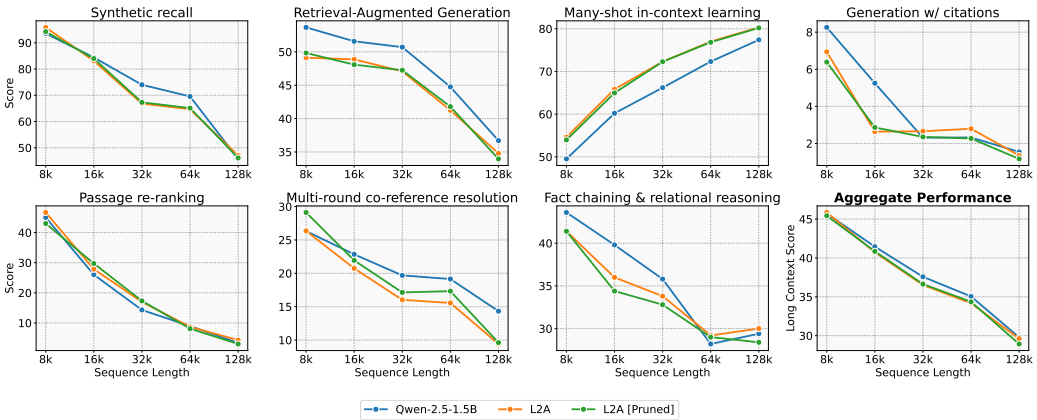

Figure 9. L2A [Pruned] is the model obtained by pruning Global Attention Modules that are more than 95% sparse—that is, layers where the Global Attention Module is not invoked for 95% of tokens. The figure shows minimal loss in long-context performance compared to L2A.

E. Additional Analysis of Native Sparse Attention (NSA)

As mentioned in Section 4.1, the original NSA work (Yuan et al., 2025) does not provide an official kernel implementation. Therefore, we rely on a publicly available open-source implementation (Li, 2025) to evaluate NSA in our experiments. To verify the correctness of this implementation, we reproduce the Needle-in-a-Haystack retrieval results reported in the original paper at a sequence length of 64K. We obtain perfect retrieval accuracy, matching the results reported in the paper,

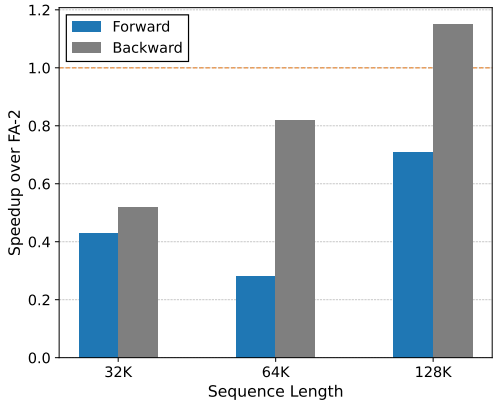

Figure 10. Native Sparse Attention (NSA) kernel (Li, 2025) speedups over FlashAttention-2 for the forward and backward pass across various sequence lengths. All experiments use the Qwen 2.5 7B configuration with 28 attention heads and a head dimension of 128.

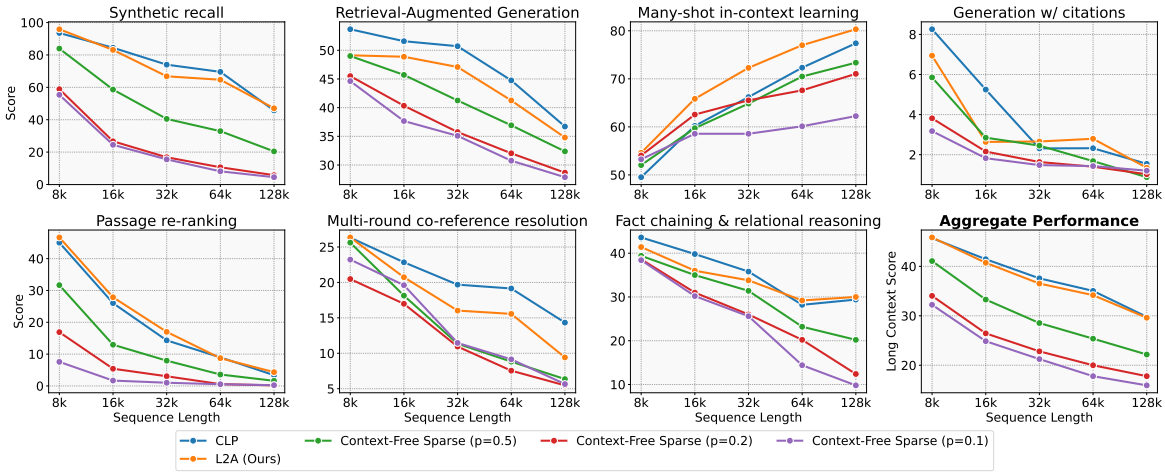

Figure 11. Context-free Sparse baselines that independently trigger Global Attention at each token with Bernoulli probability $p \in \{0.1, 0.2, 0.5\}$, corresponding to 50-90% sparsity levels. *L2A* outperforms these by a large margin, providing crucial evidence that the Router leverages local context to decide upon the invocation of Global Attention.

indicating that the implementation faithfully realizes the NSA algorithm.

We then analyze the kernel-level performance of this implementation relative to FlashAttention-2 (Dao, 2023) across varying sequence lengths. As shown in Figure 10, the evaluated NSA kernel is consistently slower than FlashAttention-2 for both the forward and backward passes. We emphasize that this observation does not contradict the claims of the original NSA paper. Specifically, the original design proposes executing the compression, top-n block selection, and Sliding Window Attention (SWA) components in parallel, whereas the available open-source kernel executes these stages sequentially. The absence of parallel execution in this implementation is likely the primary contributor to the observed performance gap.

In all experiments, we follow the hyperparameters described in the original NSA paper as closely as possible. The only deviation is the choice of selected block count n, and we find the setting in the original paper ($n = 16$) to result in degraded performance in our setup. We therefore increase n to 128 in all NSA experiments. Notably, as shown in Figure 5, even with this substantially larger n, which is more favorable to NSA than the original paper setting, *L2A* significantly outperforms NSA across all evaluated tasks. This highlights the advantage of learning when to invoke global attention on a per-token basis, as opposed to relying on a fixed long-range context budget. Although NSA learns which tokens to attend within the context, the amount of long-context capacity is predetermined and applied uniformly across tokens.

E.1. Ablation Study

E.1.1. Ablation on the Regularization Coefficient.

We next study the effect of the regularization coefficient λ in the training objective (Equation (7)) on *L2A* 's performance. As expected, increasing λ induces higher sparsity, leading to greater reliance on Local Attention and, in turn, degraded long-context performance (Figure 12). However, tuning λ is challenging because its values do not map directly to specific sparsity levels. To address this, we instead vary the sigmoid threshold in Equation (10), which enables direct control over Global Attention sparsity at test time.

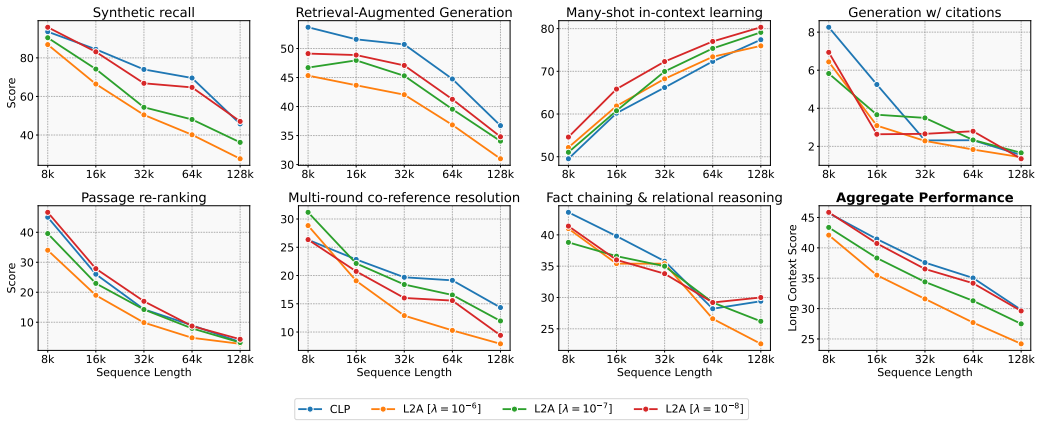

Figure 12. Ablation on the regularization coefficient λ. Increasing λ induces higher sparsity, leading to greater reliance on Local Attention and degraded long-context performance.

E.1.2. Test-time compute scaling through sigmoid threshold.

In all experiments, we apply a fixed threshold of 0.5 to the sigmoid output of the routing module to obtain discrete routing decisions, as defined in Equation 4. To examine test-time compute scaling, we vary this threshold at inference to control the sparsity of global attention. For a range of threshold values as shown in Figure 13, increasing the threshold leads to higher sparsity and lower time-to-first-token, at the cost of reduced aggregate performance, while lower thresholds recover performance at increased compute cost. This enables flexible test-time control over the compute–accuracy trade-off, allowing inference cost to be modulated without retraining.

E.1.3. Comparing full-model fine-tuning with attention and LayerNorm-only training

As described in Section 4, we restrict training to the Attention and LayerNorm layers while freezing the feed-forward (FFN) layers. This design choice is motivated by prior work showing that a large fraction of the model's pretrained parametric knowledge is encoded in FFNs (Geva et al., 2021), while attention layers primarily focus on contextual routing and composition. By preserving FFNs, we aim to retain the pretrained capabilities of the Base LLM while adapting to long-context modeling. As shown in Table 6, this strategy consistently outperforms full-model fine-tuning.

F. Implementation Details

F.1. Training Hyperparameters

All models are trained using the AdamW optimizer with an initial learning rate of 5×10^{-5}, 5% warmup steps, cosine learning rate scheduling, and gradient clipping with a norm of 1. We use a batch size of 6M tokens for the Qwen 2.5 7B-based and Qwen3 8B-based models, and 4M tokens for the Qwen 2.5 1.5B model. All experiments are conducted with a sequence length of 128K tokens using BF16 training. We train for 25B tokens the Qwen2.5 7B- and Qwen3 8B-based models, while for the smaller Qwen2.5 1.5B-based models we train on 16.7B tokens.

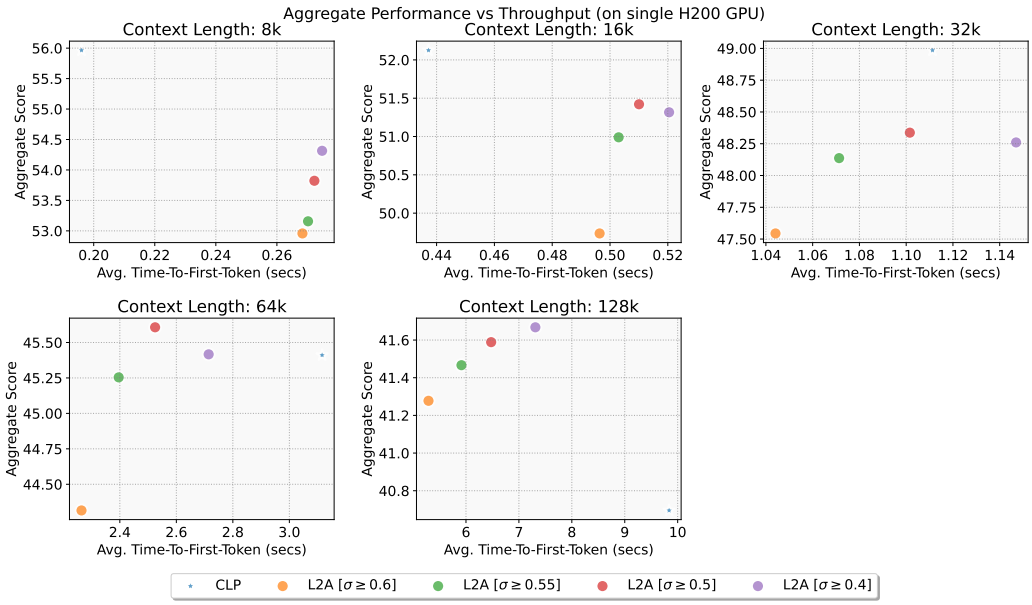

Figure 13. Scaling test time compute for *L2A* by varying the threshold of the Routing module (Equation (4)). Increasing the threshold leads to higher sparsity and lower time-to-first-token, at the cost of reduced aggregate performance.

F.2. Long-context Benchmarks

We consider a suite of long-context evaluation tasks drawn from HELMET (Yen et al., 2025), BabiLong (Kuratov et al., 2024), and MRCR (Vodrahalli et al., 2024), which we describe below.

- *Synthetic Recall*: These tasks are variants of the needle-in-a-haystack setting (Kamradt, 2024), in which the model must retrieve a critical piece of information (the "needle") from a long sequence of irrelevant or distracting tokens (the "haystack"). In addition to retrieval, these variants evaluate the model's ability to perform multi-hop tracing and information aggregation across the context. Our Recall evaluation consists of these tasks from HELMET (Yen et al., 2025): JSON Key–Value, NIAH Multi-Key (MK) Needle, NIAH Multi-Key (MK) UUID, and NIAH Multi-Value (MV).

- *Multi-round Co-reference Resolution (MRCR)*: MRCR (Vodrahalli et al., 2024) measures an LLM's ability to resolve multiple reference needles distributed across a long context. The model is presented with a multi-turn, synthetically generated conversation in which the user issues repeated requests for a piece of writing on a given topic. Embedded within the conversation are a few identical requests, and the model is ultimately prompted to return a specific instance. Task difficulty increases with both the context length and the number of needles.

- *Retrieval Augmented Generation (RAG)*: These tasks involve open-domain question answering, where the model is provided with a gold passage containing the answer, interleaved with numerous distractor passages retrieved from a large corpus. The model is required to answer the question using the provided passages. We evaluate on the following datasets from HELMET (Yen et al., 2025): Natural Questions, TriviaQA, PopQA, and HotpotQA.

- *Many-shot In-context Learning (ICL)*: ICL evaluates an LLM's ability to acquire new skills from a small number of examples provided in the prompt. In this setting, the model learns to classify inputs into different concepts based on several in-context demonstrations. We evaluate ICL performance using the following datasets from HELMET: TREC Coarse, TREC Fine, NLU, BANKING77, and CLINIC150.

- *Generation with Citations (Cite)*: LLMs are benchmarked on a realistic question-answering setting that requires generating responses with correct attributions (Bohnet et al., 2023). Given multi-faceted questions and a set of relevant passages, models are tasked with producing long-form answers while citing supporting passage identifiers (Gao et al., 2023). This evaluates the model's ability to reason over the provided context and its adherence to citation instructions.

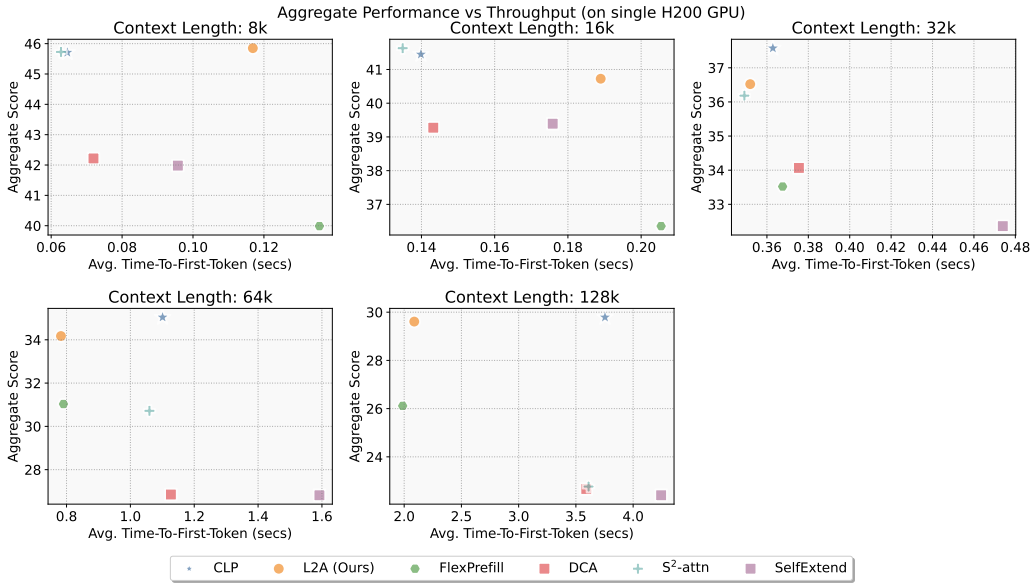

Figure 14. Time-to-first-token (TTFT) for Qwen2.5 1.5B model for various sequence lengths. *L2A* becomes progressively faster as the context length increases, highlighting the need for conditional Global Attention.

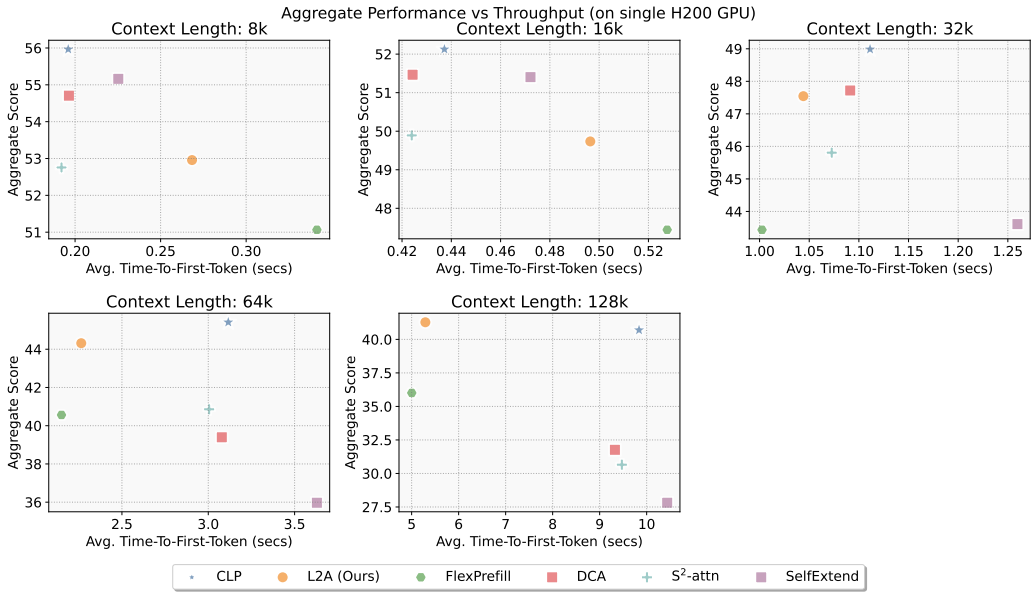

Figure 15. Time-to-first-token (TTFT) for Qwen2.5 7B model for various sequence lengths. *L2A* becomes progressively faster as the context length increases, highlighting the need for conditional Global Attention.

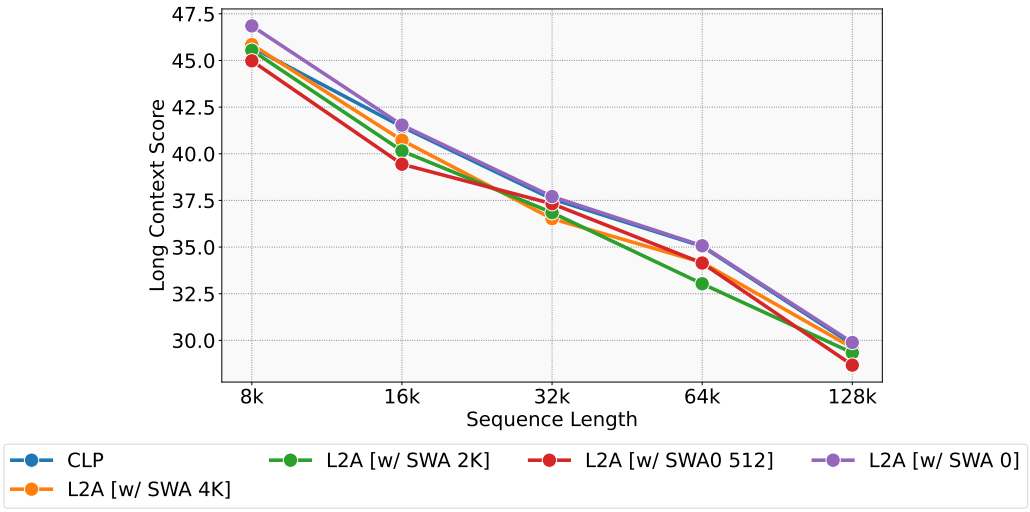

Figure 16. Aggregate long-context performance for *L2A* variants with varying SWA window sizes. The performance remains similar, while the Router module compensates for smaller SWA windows by reducing sparsity in Global Attention invocations.

We consider the ASQA (Stelmakh et al., 2023) and QAMPARI (Amouyal et al., 2023) subsets, and the outputs are evaluated for both answer correctness and citation quality.

- *Passage Re-ranking (Re-rank)*: We evaluate passage re-ranking performance using the MS MARCO benchmark (Bajaj et al., 2018) and annotated datasets from the TREC Passage Re-ranking challenge (Craswell et al., 2025). Each instance consists of a query and a set of candidate passages labeled by relevance. For a given input length L, we select k passages as context, balance label distributions, and randomize passage order to mitigate positional bias, generating three permutations per instance. Performance is measured using NDCG@10, which balances evaluation fidelity with the computational cost of inference.

- *Fact Chaining & Relational Reasoning (FCRR)*: We evaluate long-context fact chaining and relational reasoning using the QA1–QA5 tasks from the BabiLong benchmark (Kuratov et al., 2024). These tasks are designed to assess progressively more challenging reasoning capabilities, ranging from single-fact retrieval (QA1) to multi-hop fact chaining (QA2 and QA3) and relational reasoning over two- and three-argument relations (QA4 and QA5). Relevant facts are sparsely distributed across long sequences containing a large number of distractor tokens, requiring models to selectively retrieve and compose information over extended contexts. Performance on these tasks reflects a model's ability to handle long-range dependencies, multi-step reasoning, and structured relational inference under long-context noise.

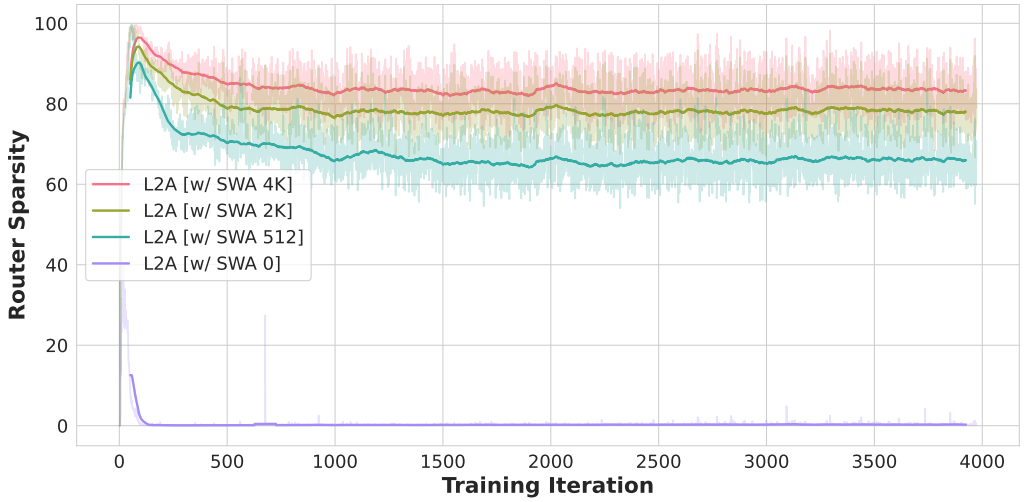

Figure 17. Sparsity levels for *L2A* variants with different SWA sizes. As the short-term local context shrinks, *L2A* increasingly relies on long-term global memory. In the extreme case, the SWA-0 configuration exhibits 0% sparsity during training.

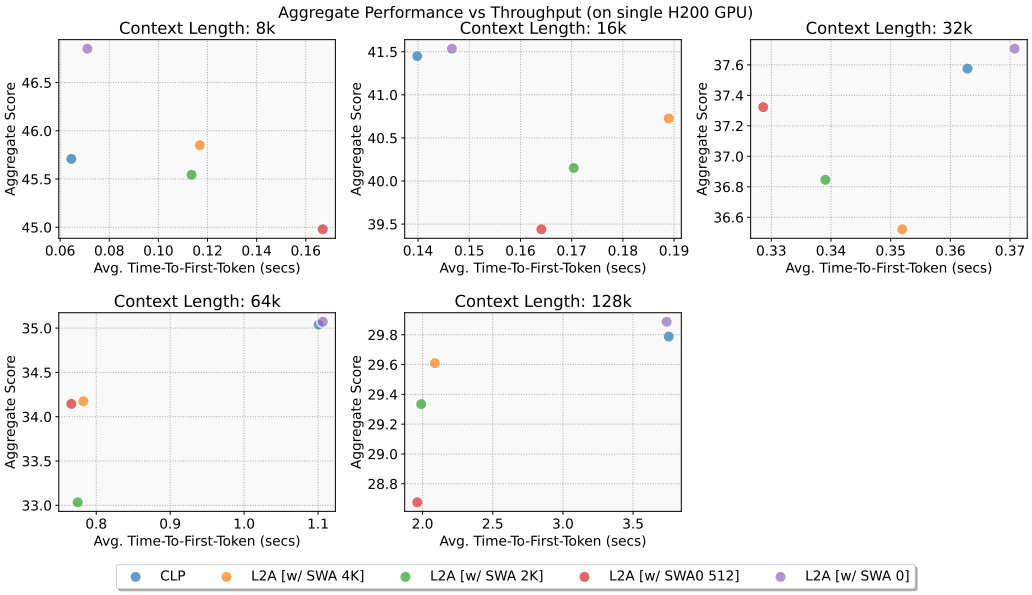

Figure 18. Time-to-first token (TTFT) for *L2A* variants with varying SWA context window from 0-4K. These results reveal a clear tradeoff: as the short-term local context shrinks, *L2A* increasingly relies on long-term global memory resulting into higher TTFT as shown.

Table 3. Task-wise performance comparison of *L2A* and baseline methods across long-context benchmarks on Qwen 2.5 7B model. *L2A* remains close to CLP performance while outperforming baselines on most tasks. All scores are reported on a 0–100 scale.

Task	Model	Context Length				
		8k	16k	32k	64k	128k
Synthetic recall	CLP	**99.1**	98.5	**98.7**	**95.4**	76.4
	L2A (Ours)	98.7	97.3	96.5	93.9	**79.2**
	FlexPrefill	98.1	96.5	91.8	88.9	69.8
	DCA	98.6	96.7	94.0	71.9	43.9
	S^2-attn	**99.1**	**98.8**	97.6	74.6	29.0
	SelfExtend	98.6	96.4	78.2	55.3	25.9
	NSA	97.9	75.0	29.1	13.8	6.2
	Base	98.6	97.1	95.9	45.1	13.9
Retrieval-Augmented Generation	CLP	**64.7**	**62.1**	**59.9**	**57.3**	**50.0**
	L2A (Ours)	62.0	60.4	57.7	55.1	**50.0**
	FlexPrefill	59.9	58.9	55.0	53.9	49.7
	DCA	62.6	60.3	53.9	48.8	43.1
	S^2-attn	63.2	60.3	56.9	53.6	44.1
	SelfExtend	62.9	60.5	52.5	46.8	38.1
	NSA	61.7	55.1	48.9	45.0	39.2
	Base	62.5	60.1	54.3	43.7	27.5
Many-shot in-context learning	CLP	73.6	79.4	81.7	84.5	87.4
	L2A (Ours)	71.4	**79.6**	**83.5**	**87.0**	**88.6**
	FlexPrefill	**74.2**	78.5	82.3	84.2	85.2
	DCA	70.0	75.6	78.7	81.2	82.4
	S^2-attn	61.8	67.0	71.5	76.7	77.0
	SelfExtend	70.2	75.0	78.0	81.2	81.8
	NSA	66.6	71.0	74.2	78.2	78.6
	Base	70.1	75.1	78.4	78.5	71.5
Generation w/ citations	CLP	**24.2**	**18.3**	8.6	**4.8**	**2.8**
	L2A (Ours)	17.2	10.5	8.5	4.1	1.9
	FlexPrefill	13.3	6.8	5.1	3.6	2.4
	DCA	21.9	15.5	**9.5**	2.1	1.2
	S^2-attn	20.9	14.9	8.4	2.9	2.5
	SelfExtend	22.5	16.3	5.9	2.5	2.3
	NSA	15.3	6.4	2.8	2.1	1.5
	Base	21.7	16.3	8.2	2.2	1.3
Passage re-ranking	CLP	48.1	32.5	**25.1**	**18.1**	8.4
	L2A (Ours)	44.0	26.8	20.1	12.5	**9.6**
	FlexPrefill	**50.5**	30.8	19.4	9.2	5.0
	DCA	46.3	**33.5**	23.2	9.6	1.8
	S^2-attn	46.7	32.7	24.2	14.0	3.8
	SelfExtend	47.3	33.2	18.1	6.8	0.6
	NSA	42.0	9.4	4.5	0.3	0.2
	Base	46.3	33.2	22.7	1.5	0.0
Multi-round co-reference resolution	CLP	34.6	30.4	28.9	**26.3**	**21.3**
	L2A (Ours)	32.9	31.5	25.8	24.2	19.3
	FlexPrefill	29.7	27.0	28.3	24.5	17.0
	DCA	32.0	30.9	29.9	22.4	14.8
	S^2-attn	32.5	**31.8**	26.9	25.9	19.0
	SelfExtend	32.4	30.7	**30.8**	21.0	11.4
	NSA	**35.0**	26.1	16.6	13.7	8.7
	Base	32.4	31.5	29.6	25.8	11.6
Fact chaining & relational reasoning	CLP	47.4	43.6	40.0	31.4	38.6
	L2A (Ours)	44.6	42.2	40.6	33.4	**40.4**
	FlexPrefill	47.6	42.6	37.8	33.4	35.8
	DCA	51.6	**47.8**	**44.8**	39.8	35.2
	S^2-attn	45.2	43.8	35.2	38.2	39.2
	SelfExtend	**52.2**	47.6	41.8	38.2	34.6
	NSA	42.2	39.2	31.2	24.0	17.6
	Base	51.6	**47.8**	43.0	**41.8**	21.4
Average across tasks	CLP	**56.0**	**52.1**	**49.0**	**45.4**	40.7
	L2A (Ours)	53.0	49.7	47.5	44.3	**41.3**
	FlexPrefill	53.3	48.7	45.7	42.5	37.8
	DCA	54.7	51.5	47.7	39.4	31.8
	S^2-attn	52.8	49.9	45.8	40.9	30.7
	SelfExtend	55.2	51.4	43.6	36.0	27.8
	NSA	51.5	40.3	29.6	25.3	21.7
	Base	54.7	51.6	47.4	34.1	21.0

Table 4. Task-wise performance comparison of *L2A* and baseline methods across long-context benchmarks on Qwen 2.5 1.5B model. *L2A* remains close to CLP performance while outperforming baselines on most tasks.All scores are reported on a 0–100 scale.

Task	Model	Context Length				
		8k	16k	32k	64k	128k
Synthetic recall	CLP	93.6	**84.4**	**74.0**	**69.6**	45.8
	L2A (Ours)	**95.9**	83.1	66.8	64.7	**47.1**
	FlexPrefill	90.4	77.4	60.2	54.9	31.9
	DCA	85.9	81.1	60.8	32.7	18.3
	S^2-attn	93.5	81.1	64.8	41.2	16.3
	SelfExtend	85.1	80.7	52.1	33.9	21.7
	NSA	76.2	29.7	14.2	9.3	4.1
	Base	85.4	80.1	64.5	19.4	3.4
Retrieval-Augmented Generation	CLP	**53.7**	**51.6**	**50.7**	**44.8**	**36.7**
	L2A (Ours)	49.1	48.9	47.1	41.2	34.8
	FlexPrefill	52.0	51.0	47.2	43.0	35.8
	DCA	49.7	46.0	42.2	36.3	28.8
	S^2-attn	51.8	50.6	48.6	39.6	27.4
	SelfExtend	49.4	45.9	39.8	33.6	27.4
	NSA	48.5	44.0	39.0	32.3	30.2
	Base	49.8	46.1	41.8	30.2	22.8
Many-shot in-context learning	CLP	49.5	60.2	66.2	72.3	77.4
	L2A (Ours)	54.6	**65.8**	**72.3**	**77.0**	**80.3**
	FlexPrefill	39.2	53.0	63.0	68.9	72.0
	DCA	54.7	64.8	69.8	74.2	76.3
	S^2-attn	54.2	62.4	68.8	73.8	72.8
	SelfExtend	54.4	65.2	70.2	73.6	74.4
	NSA	**61.5**	65.0	65.4	65.3	65.0
	Base	54.5	65.2	69.4	68.1	43.6
Generation w/ citations	CLP	8.3	5.2	2.3	2.3	1.5
	L2A (Ours)	6.9	2.6	2.7	2.8	1.3
	FlexPrefill	5.5	3.1	3.3	1.9	**1.6**
	DCA	7.9	**5.3**	3.2	1.4	0.9
	S^2-attn	**8.9**	3.8	3.5	**3.0**	1.4
	SelfExtend	7.4	5.0	2.9	1.4	1.0
	NSA	7.1	3.4	2.6	1.6	0.8
	Base	8.1	4.9	**4.4**	0.7	0.9
Passage re-ranking	CLP	45.0	26.0	14.3	**9.0**	3.4
	L2A (Ours)	**46.7**	27.9	**17.0**	8.7	**4.3**
	FlexPrefill	25.7	13.0	8.4	3.5	2.0
	DCA	33.6	20.0	9.2	1.7	0.0
	S^2-attn	41.4	**31.1**	12.3	8.0	2.2
	SelfExtend	34.4	20.3	8.9	0.4	0.0
	NSA	32.8	16.3	10.0	1.5	1.4
	Base	33.5	21.6	10.0	4.3	0.0
Multi-round co-reference resolution	CLP	26.3	22.8	**19.7**	**19.1**	**14.3**
	L2A (Ours)	26.4	20.7	16.0	15.6	9.4
	FlexPrefill	24.0	19.4	16.5	13.0	12.1
	DCA	26.7	22.8	17.6	13.9	11.0
	S^2-attn	**26.8**	22.2	19.0	17.8	12.8
	SelfExtend	26.3	**23.1**	18.6	18.7	8.9
	NSA	25.6	18.7	8.3	9.7	6.1
	Base	26.4	22.7	18.0	15.4	5.6
Fact chaining & relational reasoning	CLP	**43.6**	39.8	35.8	28.2	29.4
	L2A (Ours)	41.4	36.0	33.8	29.2	**30.0**
	FlexPrefill	43.2	37.6	36.0	**32.0**	27.4
	DCA	37.0	35.0	35.6	27.8	23.4
	S^2-attn	43.4	**40.2**	**36.2**	31.6	26.4
	SelfExtend	37.0	35.6	34.0	26.2	23.4
	NSA	40.4	30.6	23.0	18.0	11.0
	Base	38.2	35.2	31.0	24.0	13.0
Average across tasks	CLP	45.7	41.4	**37.6**	**35.0**	**29.8**
	L2A (Ours)	**45.9**	40.7	36.5	34.2	29.6
	FlexPrefill	40.0	36.4	33.5	31.0	26.1
	DCA	42.2	39.3	34.1	26.9	22.7
	S^2-attn	45.7	**41.6**	36.2	30.7	22.8
	SelfExtend	42.0	39.4	32.4	26.8	22.4
	NSA	41.7	29.7	23.2	19.7	17.0
	Base	42.3	39.4	34.2	23.2	12.8

Table 5. Task-wise performance comparison of *L2A* and baseline methods across long-context benchmarks on Qwen3 8B model. *L2A* remains close to CLP performance while outperforming baselines on most tasks.All scores are reported on a 0–100 scale.

Task	Model	Context Length				
		8k	**16k**	**32k**	**64k**	**128k**
Synthetic recall	CLP	**100.0**	**99.9**	**99.7**	**98.8**	87.9
	L2A (Ours)	99.8	99.8	99.4	98.2	87.1
	FlexPrefill	99.4	99.5	99.0	98.4	**88.6**
	DCA	99.2	99.4	95.4	52.7	40.0
	Base	99.1	99.4	97.4	40.0	0.0
Retrieval-Augmented Generation	CLP	**65.4**	**63.6**	**60.5**	**59.1**	**52.2**
	L2A (Ours)	63.2	61.2	**60.5**	58.0	51.0
	FlexPrefill	63.0	60.2	58.0	56.8	46.5
	DCA	63.5	61.2	57.3	52.8	47.9
	Base	63.5	61.5	58.0	47.9	7.0
Many-shot in-context learning	CLP	70.8	74.7	79.1	82.8	85.4
	L2A (Ours)	64.4	67.3	74.4	80.2	**85.7**
	FlexPrefill	**71.2**	**75.8**	**80.2**	**83.7**	85.1
	DCA	65.4	68.5	70.5	72.9	76.2
	Base	65.5	68.9	70.1	68.8	9.3
Generation w/ citations	CLP	32.6	26.6	**17.0**	**12.3**	**5.9**
	L2A (Ours)	30.4	27.1	15.2	5.2	2.1
	FlexPrefill	19.3	7.7	5.7	5.0	3.6
	DCA	33.3	26.8	16.4	3.9	1.6
	Base	**34.8**	**27.3**	15.2	6.4	0.7
Passage re-ranking	CLP	**50.0**	36.3	30.1	20.4	**11.2**
	L2A (Ours)	49.0	36.7	25.9	15.6	7.5
	FlexPrefill	42.3	32.4	24.3	12.1	7.0
	DCA	47.5	37.4	27.6	13.9	9.3
	Base	46.3	**37.8**	**32.8**	**20.7**	2.1
Multi-round co-reference resolution	CLP	**38.5**	33.9	31.9	33.9	**30.1**
	L2A (Ours)	36.3	34.1	31.2	**34.6**	26.7
	FlexPrefill	34.0	29.8	31.9	31.6	25.5
	DCA	35.7	**35.5**	**33.4**	29.2	17.2
	Base	35.7	34.9	32.5	29.9	8.9
Fact chaining & relational reasoning	CLP	43.4	43.6	46.6	37.8	35.6
	L2A (Ours)	47.2	**47.6**	**47.6**	38.6	**37.8**
	FlexPrefill	44.8	39.0	39.0	34.2	31.0
	DCA	**52.8**	47.0	46.0	**40.8**	30.0
	Base	**52.8**	47.2	44.0	27.6	0.4
Average across tasks	CLP	**57.2**	**54.1**	**52.1**	**49.3**	**44.0**
	L2A (Ours)	55.7	53.4	50.6	47.2	42.5
	FlexPrefill	53.4	49.2	48.3	46.0	41.0
	DCA	56.8	53.7	49.5	38.0	31.7
	Base	56.8	53.9	50.0	34.5	4.1

Table 6. Zero-shot performance on standard short-context benchmarks for Qwen 2.5 models at different scales. In these set of results, we compare CLP for Attention and LayerNorm parameters only vs. training all parameters. At both model scales, training Attn + LayerNorm parameters only is the better choice.

Scale	Method	BoolQ acc ↑	CommSenseQA acc ↑	PIQA acc_n ↑	Winogrande acc ↑	ARC-E acc_n ↑	ARC-C acc_n ↑	MMLU acc ↑	SWDE contains ↑	Avg
1.5B	CLP (Attn + Norm)	**73.36**	**74.45**	75.95	64.72	**88.72**	**73.81**	**60.46**	86.50	**74.74**
	CLP (Full Model)	63.67	72.65	**76.01**	**64.80**	88.05	72.53	59.29	**87.58**	73.07
7B	CLP (Attn + Norm)	**82.57**	**84.52**	79.43	**76.01**	**96.09**	**88.65**	**73.39**	**91.00**	**83.95**
	CLP (Full Model)	79.60	82.88	**79.76**	74.03	95.54	87.88	72.76	89.56	82.75