MoBA: Mixture of Block Attention for Long-Context LLMs

Enzhe Lu[1] **Zhejun Jiang**[1] **Jingyuan Liu**[1] **Yulun Du**[1] **Tao Jiang**[1] **Chao Hong**[1]
Shaowei Liu[1] **Weiran He**[1] **Enming Yuan**[1] **Yuzhi Wang**[1] **Zhiqi Huang**[1] **Huan Yuan**[1]
Suting Xu[1] **Xinran Xu**[1] **Guokun Lai**[1] **Yanru Chen**[1] **Huabin Zheng**[1] **Junjie Yan**[1]
Jianlin Su[1] **Yuxin Wu**[1] **Yutao Zhang**[1] **Zhilin Yang**[1]
Xinyu Zhou[1] **Mingxing Zhang**[2] **Jiezhong Qiu**[3] *
[1] Moonshot AI [2] Tsinghua University [3] Hangzhou Institute of Medicine, CAS

Abstract

Scaling the effective context length is essential for advancing large language models (LLMs) toward artificial general intelligence (AGI). However, the quadratic increase in computational complexity inherent in traditional attention mechanisms presents a prohibitive overhead. Existing approaches either impose strongly biased structures, such as sink or window attention which are task-specific, or radically modify the attention mechanism into linear approximations, whose performance in complex reasoning tasks remains inadequately explored.

In this work, we propose a solution that adheres to the "less structure" principle, allowing the model to determine where to attend autonomously, rather than introducing predefined biases. We introduce Mixture of Block Attention (MoBA), an innovative approach that applies the principles of Mixture of Experts (MoE) to the attention mechanism. This novel architecture demonstrates superior performance on long-context tasks while offering a key advantage: the ability to seamlessly transition between full and sparse attention, enhancing efficiency without the risk of compromising performance. MoBA has already been deployed to handle actual production workloads with long-context requirements, demonstrating significant advancements in efficient attention computation for LLMs. Our code is available at `https://github.com/MoonshotAI/MoBA`.

1 Introduction

The pursuit of artificial general intelligence (AGI) has driven the development of large language models (LLMs) to unprecedented scales, with the promise of handling complex tasks that mimic human cognition. A pivotal capability for achieving AGI is the ability to process, understand, and generate long sequences, which is essential for many applications, from historical data analysis to complex reasoning and decision-making processes. This growing demand for extended context processing can be seen not only in the popularity of long input prompt understanding, as showcased by models like Kimi [1], Claude [2] and Gemini [3], but also in recent explorations of long chain-of-thought (CoT) output capabilities in Kimi k1.5 [4], DeepSeek-R1 [5], and OpenAI o1/o3 [6].

However, extending the sequence length in LLMs is non-trivial due to the quadratic growth in computational complexity associated with the vanilla attention [7]. This challenge has spurred a wave of research aimed at improving efficiency without sacrificing performance. One prominent direction capitalizes on the inherent sparsity of attention scores. This sparsity arises both mathematically

*Co-corresponding authors: Xinyu Zhou (`zhouxinyu@moonshot.cn`), Mingxing Zhang (`zhangmingxing@mail.tsinghua.edu.cn`) and Jiezhong Qiu (`qiujiezhong@him.cas.cn`).

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

— from the softmax operation, where various sparse attention patterns have be studied [8] — and biologically [9], where sparse connectivity is observed in brain regions related to memory storage.

Existing approaches often leverage predefined structural constraints, such as sink-based [10] or sliding window attention [11], to exploit this sparsity. While these methods can be effective, they tend to be highly task-specific, potentially hindering the model's overall generalizability. Alternatively, a range of dynamic sparse attention mechanisms, exemplified by Quest [12], Minference [8], and RetrievalAttention [13], select subsets of tokens at inference time. Although such methods can reduce computation for long sequences, they do not substantially alleviate the intensive training costs of long-context models, making it challenging to scale LLMs efficiently to contexts on the order of millions of tokens. Another promising alternative way has recently emerged in the form of linear attention models, such as Mamba [14], RWKV [15, 16], and RetNet [17]. These approaches replace canonical softmax-based attention with linear approximations, thereby reducing the computational overhead for long-sequence processing. However, due to the substantial differences between linear and conventional attention, adapting existing Transformer models typically incurs high conversion costs [18, 19, 20, 21] or requires training entirely new models from scratch [22]. More importantly, evidence of their effectiveness in complex reasoning tasks remains limited.

Consequently, a critical research question arises: How can we design a robust and adaptable attention architecture that retains the original Transformer framework while **adhering to a "less structure" principle, allowing the model to determine where to attend without relying on predefined biases?** Ideally, such an architecture would transition seamlessly between full and sparse attention modes, thus maximizing compatibility with existing pre-trained models and enabling both efficient inference and accelerated training without compromising performance.

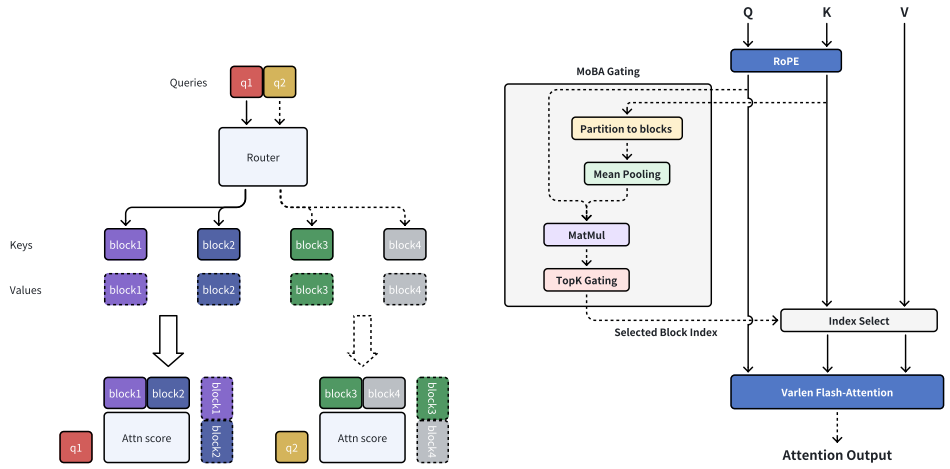
Thus, we introduce Mixture of Block Attention (MoBA), a novel architecture that builds upon the innovative principles of Mixture of Experts (MoE) [23] and applies them to the attention mechanism. MoE has been used primarily in the feedforward network (FFN) layers of Transformers [24, 25, 26], but MoBA pioneers its application to long context attention, allowing dynamic selection of historically relevant KV blocks for each query token. This approach not only enhances the efficiency of LLMs but also enables them to handle longer and more complex prompts without a proportional increase in resource consumption. MoBA addresses the computational inefficiency of traditional attention mechanisms by partitioning the context into blocks and employing a gating mechanism to selectively route query tokens to the most relevant blocks. This block sparse attention significantly reduces the computational costs, paving the way for more efficient processing of long sequences. The model's ability to dynamically select the most informative blocks of keys leads to improved performance and efficiency, particularly beneficial for tasks involving extensive contextual information.

In this paper, we detail the architecture of MoBA, firstly its block partitioning and routing strategy, and secondly its efficient implementation. We further present experimental results that demonstrate MoBA's superior performance in tasks requiring the processing of long sequences. Our work contributes a novel approach to efficient attention computation, pushing the boundaries of what is achievable with LLMs in handling complex and lengthy inputs.

2 Method

In this work, we introduce a novel architecture, termed Mixture of Block Attention (MoBA), which extends the capabilities of the Transformer model by dynamically selecting historical KV blocks for attention computation. MoBA is inspired by techniques of Mixture of Experts (MoE) and sparse attention. The former technique has been predominantly applied to the feedforward network (FFN) layers within the Transformer architecture, while the latter has been widely adopted in scaling Transformers to handle long contexts. Our method is innovative in applying the MoE principle to the attention mechanism itself, allowing for more efficient and effective processing of long sequences.

Preliminaries: Standard Attention in Transformer. We first revisit the standard Attention [7] in Transformers. For simplicity, we revisit the case where a single query token $\boldsymbol{q} \in \mathbb{R}^{1\times d}$ attends to the N key and value tokens, denoting $\boldsymbol{K}, \boldsymbol{V} \in \mathbb{R}^{N\times d}$, respectively. The standard attention is computed as $\mathrm{Attn}(\boldsymbol{q}, \boldsymbol{K}, \boldsymbol{V}) = \mathrm{Softmax}\big(\boldsymbol{q}\boldsymbol{K}^\top\big)\boldsymbol{V}$, where d denotes the dimension of a single attention head. We focus on the single-head scenario for clarity. The extension to multi-head attention involves concatenating the outputs from multiple such single-head attention operations.



(a) (b)

Figure 1: Illustration of mixture of block attention (MoBA). **(a)** A running example of MoBA; **(b)** Integration of MoBA into Flash Attention.

Algorithm 1 MoBA (Mixture of Block Attention) Implementation

Require: Query, key and value matrices $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{N \times h \times d}$; MoBA hyperparameters (block size B and top-k); h and d denote # attention heads and head dimension. Also denote $n = N/B$ to be # blocks.

$\{\tilde{\mathbf{K}}_i, \tilde{\mathbf{V}}_i\} = \text{split\_blocks}(\mathbf{K}, \mathbf{V}, B)$, where $\tilde{\mathbf{K}}_i, \tilde{\mathbf{V}}_i \in \mathbb{R}^{B \times h \times d}, i \in [n]$
$\bar{\mathbf{K}} = \text{mean\_pool}(\mathbf{K}, B) \in \mathbb{R}^{n \times h \times d}$
$\mathbf{S} = \mathbf{Q}\bar{\mathbf{K}}^\top \in \mathbb{R}^{N \times h \times n}$
$\mathbf{M} = \text{create\_causal\_mask}(N, n)$
$\mathbf{G} = \text{topk}(\mathbf{S} + \mathbf{M}, k)$
$\mathbf{Q}^s, \tilde{\mathbf{K}}^s, \tilde{\mathbf{V}}^s = \text{get\_self\_attn\_block}(\mathbf{Q}, \tilde{\mathbf{K}}, \tilde{\mathbf{V}})$
$\mathbf{Q}^m, \tilde{\mathbf{K}}^m, \tilde{\mathbf{V}}^m = \text{index\_select\_moba\_attn\_block}(\mathbf{Q}, \tilde{\mathbf{K}}, \tilde{\mathbf{V}}, \mathbf{G})$
$\mathbf{O}^s = \text{flash\_attention\_varlen}(\mathbf{Q}^s, \tilde{\mathbf{K}}^s, \tilde{\mathbf{V}}^s, \text{causal=True})$
$\mathbf{O}^m = \text{flash\_attention\_varlen}(\mathbf{Q}^m, \tilde{\mathbf{K}}^m, \tilde{\mathbf{V}}^m, \text{causal=False})$
$\mathbf{O} = \text{combine\_with\_online\_softmax}(\mathbf{O}^s, \mathbf{O}^m)$
return $\mathbf{O}$

2.1 MoBA Architecture

Different from standard attention where each query tokens attend to the entire context, MoBA enables each query token to only attend to a subset of keys and values:

$$\text{MoBA}(\boldsymbol{q}, \boldsymbol{K}, \boldsymbol{V}) = \text{Softmax}\Big(\boldsymbol{q}\boldsymbol{K}[I]^\top\Big)\boldsymbol{V}[I], \tag{1}$$

where $I \subseteq [N]$ is the set of selected keys and values. The key innovation in MoBA is the block partitioning and selection strategy. We divide the full context of length N into n blocks, where each block represents a subset of subsequent tokens. Without loss of generality, we assume that the length N is divisible by the number of blocks n. We further denote $B = \frac{N}{n}$ to be the block size and $I_i = [(i-1) \times B + 1, i \times B]$ to be the range of the i-th block. By applying the top-k gating from MoE, we enable each query to selectively focus on a subset of tokens from different blocks, rather than the entire context, namely $I = \cup_{g_i > 0} I_i$. Here the model employs a gating mechanism, denoted by g_i, to select the most relevant blocks for each query token. The MoBA gate first computes the affinity score s_i measuring the relevance between query $\boldsymbol{q}$ and the i-th block, and applies a top-k

gating among all blocks. More formally, the gate value for the i-th block g_i is computed by

$$g_i = \begin{cases} 1 & s_i \in \operatorname{Topk}\left(\{s_j | j \in [n]\}, k\right) \\ 0 & \text{otherwise} \end{cases}, \tag{2}$$

where $\operatorname{Topk}(\cdot, k)$ denotes the set containing k highest scores among the affinity scores calculated for each block. In this work, the score s_i is computed by the inner product between $\boldsymbol{q}$ and the mean pooling of $\boldsymbol{K}[I_i]$ along the sequence dimension $s_i = \langle \boldsymbol{q}, \operatorname{mean\_pool}(\boldsymbol{K}[I_i]) \rangle$.

It is important to maintain causality in autoregressive language models, as they generate text by next-token prediction based on previous tokens. This sequential generation process ensures that a token cannot influence tokens that come before it, thus preserving the causal relationship. MoBA preserves causality through two specific designs:

Causality: No Attention to Future Blocks. MoBA ensures that a query token cannot be routed to any future blocks. By limiting the attention scope to current and past blocks, MoBA adheres to the autoregressive nature of language modeling. More formally, denoting $\operatorname{pos}(\boldsymbol{q})$ as the position index of the query $\boldsymbol{q}$, we set $s_i = -\infty$ and $g_i = 0$ for any blocks i such that $\operatorname{pos}(\boldsymbol{q}) < i \times B$.

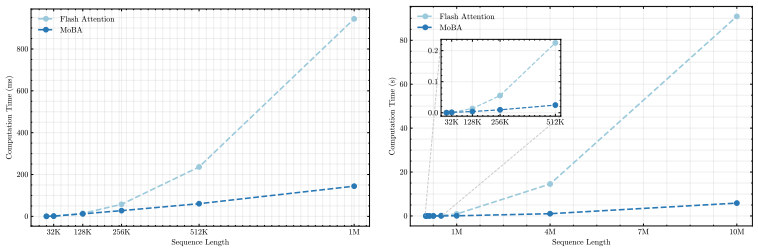
Current Block Attention and Causal Masking. We define the "current block" as the block that contains the query token itself. The routing to the current block could also violate causality, since mean pooling across the entire block can inadvertently include future information. To address this, we enforce that each token must be routed to its respective current block and apply a causal mask during the current block attention. This strategy not only avoids any leakage of information from subsequent tokens but also encourages attention to the local context. More formally, we set $g_i = 1$ for the block i where the position of the query token $\operatorname{pos}(\boldsymbol{q})$ is within the interval I_i. From the perspective of Mixture-of-Experts (MoE), the current block attention in MoBA is akin to the role of shared experts in modern MoE architectures [27, 28], where static routing rules are added when expert selection.

Next, we discuss some additional key design choices of MoBA, such as its block segmentation strategy and the hybrid of MoBA and full attention.

Fine-Grained Block Segmentation. The positive impact of fine-grained expert segmentation in improving mode performance has been well-documented in the Mixture-of-Experts (MoE) literature [27, 28]. In this work, we explore the potential advantage of applying a similar fine-grained segmentation technique to MoBA. MoBA, inspired by MoE, operates segmentation along the context-length dimension rather than the FFN intermediate hidden dimension. Therefore our investigation aims to determine if MoBA can also benefit when we partition the context into blocks with a finer grain. More experimental results can be found in Section 3.1.

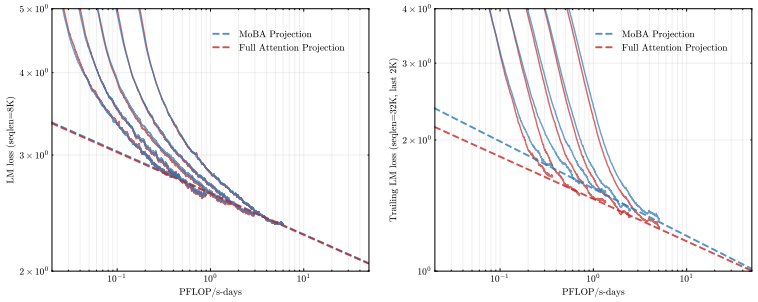
Hybrid of MoBA and Full Attention. MoBA is designed to be a substitute for full attention, maintaining the same model parameters without any addition or subtraction. This feature inspires us to conduct smooth transitions between full attention and MoBA. Specifically, at the initialization stage, each attention layer has the option to select full attention or MoBA, and this choice can be dynamically altered during training if necessary. A similar idea of transitioning full attention to sliding window attention has been studied in [29]. More results can be found in Section 3.2.

Implementation. We provide a high-performance implementation of MoBA, by incorporating optimization techniques from FlashAttention [30] and MoE [31]. Figure 2 demonstrates the high efficiency of MoBA, while we defer the detailed experiments on efficiency and scalability to Section 3.4. Our implementation consists of five major steps: (1) Determine the assignment of query tokens to KV blocks according to the gating network and causal mask; (2) Arrange the ordering of query tokens based on their assigned KV blocks; (3) Compute attention outputs for each KV block and the query tokens assigned to it. This step can be optimized by FlashAttention with varying lengths; (4) Re-arrange the attention outputs back to their original ordering; (5) Combine the corresponding attention outputs using online Softmax (i.e., tiling), as a query token may attend to its current block and multiple historical KV blocks. The detailed algorithmic workflow is formalized in Algorithm 1 and visualized in Figure 1b, illustrating how MoBA can be implemented based on MoE and FlashAttention. First, the KV matrices are partitioned into blocks (Line 1-2). Next, the gating score is computed, which measures the relevance between query tokens and KV blocks (Lines 3-7). A top-k operator is applied on the gating score (together with causal mask), resulting in a sparse query-to-KV-block mapping matrix $\boldsymbol{G}$ to represent the assignment of queries to KV blocks (Line 8). Then, query tokens are arranged based on the query-to-KV-block mapping, and block-wise attention



(a) (b)

Figure 2: Efficiency of MoBA. **(a)** Sequence length speedup evaluation: Computation time scaling of MoBA v.s. Flash Attention with increasing sequence lengths (8K-1M). **(b)** Fixed Sparsity Ratio scaling: Computation time scaling comparison between MoBA and Flash Attention across increasing sequence lengths (8K-10M), maintaining a constant sparsity ratio of 95.31% (fixed 64 MoBA blocks with variance block size and fixed top-k=3).



(a) (b)

Figure 3: Scaling Law (the fitted scaling law curve can be found in Table 3 of Appendix). **(a)** LM loss on validation set (seqlen=8K); **(b)** trailing LM loss on validation set (seqlen=32K, last 2K tokens).

outputs are computed (Line 9-12). Notably, attention to historical blocks (Line 11 and 14) and the current block attention (Line 10 and 13) are computed separately, as additional causality needs to be maintained in the current block attention. Finally, the attention outputs are rearranged back to their original ordering and combined with online softmax (Line 16) [32, 33].

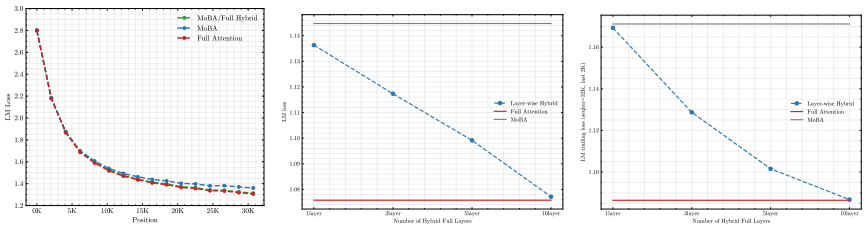
Overall, MoBA allows the model to adaptively and dynamically focus on the most informative blocks of the context. This is particularly beneficial for tasks involving long documents or sequences, where attending to the entire context may be unnecessary and computationally expensive. MoBA's ability to selectively attend to relevant blocks enables more nuanced and efficient processing of information.

3 Experiments

3.1 Scaling Law Experiments and Ablation Studies

Scalability w.r.t. LM Loss. To assess the effectiveness of MoBA, we perform scaling law experiments by comparing the validation loss of language models trained using either full attention or MoBA. Following the Chinchilla scaling law [34], we train five language models of varying sizes with a sufficient number of training tokens to ensure that each model achieves its training optimum. Detailed configurations of the scaling law experiments can be found in Table 2 of Appendix. Both MoBA and full attention models are trained with a sequence length of 8K. For MoBA models, we set the block size to 512 and select the top-3 blocks for attention, resulting in a sparse attention pattern with sparsity up to $1 - \frac{512\times 3}{8192} = 81.25\%$[2]. In particular, MoBA serves as an alternative to full attention, meaning that it does not introduce new parameters or remove existing ones. This design simplifies our

[2]Since we set top-k=3, thus each query token can attend to at most 2 history block and the current block.



(a) (b) (c)

Figure 5: Hybrid of MoBA and full attention. **(a)** position-wise LM loss for MoBA, full attention, and MoBA/full hybrid training; **(b)** SFT LM loss w.r.t # full attention layers in layer-wise hybrid; **(c)** SFT trailing LM loss (seqlen=32K, last 2K) w.r.t # full attention layers in layer-wise hybrid.

comparison process, as the only difference across all experiments lies in the attention modules, while all other hyperparameters, including the learning rate and batch size, remain constant. As shown in Figure 3a, the validation loss curves for MoBA and full attention display very similar scaling trends. Specifically, the validation loss differences between these two attention mechanisms remain consistent within a range of 1e-3. This suggests that MoBA achieves scaling performance that is comparable to full attention, despite its sparse attention pattern with sparsity up to 75%.

Long Context Scalability. However, LM loss may be skewed by the data length distribution [35], which is typically dominated by short sequences. To fully assess the long-context capability of MoBA, we assess the **LM loss of trailing tokens (trailing LM loss, in short)**, which computes the LM loss of the last few tokens in the sequence. We count this loss only for sequences that reach the maximum sequence length to avoid biases that may arise from very short sequences. A detailed discussion on the scale law regarding trailing LM loss can be found in Appendix A.2. These metrics provide insights into the model's ability to generate the final portion of a sequence, which can be particularly informative for tasks involving long context understanding. Therefore, we adopt a modified experimental setting by increasing the maximum sequence length from 8k to 32k. This adjustment leads to an even sparser attention pattern for MoBA, achieving a sparsity level of up to $1 - \frac{512 \times 3}{32768} = 95.31\%$. As shown in Figure 3b, although MoBA exhibits a marginally higher trailing LM loss compared to full attention in all five experiments, the loss gap is progressively narrowing. This experiment implies the long-context scalability of MoBA.

Ablation Study on Fine-Grained Block Segmentation. We carry out a series of experiments using a 1.5B parameter model with a 32K context length. The hyperparameters of block size and top-k are adjusted to maintain a consistent level of attention sparsity. Specifically, we divide the 32K context into 8, 16, 32, 64, and 128 blocks, and correspondingly select 2, 4, 8, 16, and 32 blocks, ensuring an attention sparsity of 75%. As shown in Figure 4, MoBA's performance is significantly affected by block granularity. Specifically, there is a performance difference of 1e-2 between the coarsest-grained setting (selecting 2 blocks from 8) and the settings with finer granularity (selecting 32 blocks from 128). These findings suggest that fine-grained segmentation appears to be a general technique for enhancing the performance of models within the MoE family, including MoBA.

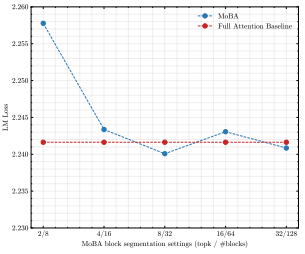


Figure 4: Fine-Grained Block Segmentation.

Pareto Frontier in terms of Efficiency and Long-context Capability. We investigate the pareto frontier by studying the relationship between attention sparsity and LM loss. We pre-train an 800M parameter model in 32K context length. We divide the 32k context length into different number of blocks (ranging from 16 to 256), and let MoBA select a fixed number of topk=3 blocks. As shown in Table 7 of Appendix, a sparsity around 3/32 turns out to be a good choice to balance efficiency (sparsity) and long-context capability.

3.2 Hybrid of MoBA and Full Attention

As discussed in Section 2, we design MoBA to be a flexible substitute for full attention, so that it can easily switch from/to full attention with minimal overhead and achieve comparable long-context performance. In this section, we first show seamless transition between full attention and MoBA can be a solution for efficient long-context pre-training. Then we discuss the layer-wise hybrid strategy, mainly for the performance of supervised fine-tuning (SFT).

MoBA/Full Hybrid Training. We train three models, each with 1.5B parameters, on 30B tokens with a context length of 32K tokens. For the hyperparameters of MoBA, the block size is set to 2048, and the top-k parameter is set to 3. The detailed training recipes are as follows: (1) MoBA/full hybrid: This model is trained using a two-stage recipe. In the first stage, MoBA is used to train on 90% of the tokens. In the second stage, the model switches to full attention for the remaining 10% of the tokens; (2) Full attention: This model is trained using full attention throughout the entire training; (3) MoBA: This model is trained exclusively using MoBA.

We evaluate their long-context performance via position-wise language model (LM) loss, which is a fine-grained metric to evaluate lm loss at each position within a sequence. Unlike the vanilla LM loss, which is computed by averaging the LM loss across all positions, the position-wise LM loss breaks down the loss for each position separately. Similar metrics have been suggested by [3, 36], who noticed that position-wise LM loss follows a power-law trend relative to context length. As shown in Figure 5a, the MoBA-only recipe results in higher position-wise losses for trailing tokens. Importantly, our MoBA/full hybrid recipe reaches a loss nearly identical to that of full attention, which highlights the effectiveness of the hybrid training recipe in balancing training efficiency with model performance. More interestingly, we have not observed significant loss spikes during the switch between MoBA and full attention, again demonstrating the flexibility and robustness of MoBA.

Layer-wise Hybrid. This flexibility of MoBA encourages us to delve into a more sophisticated strategy — the layer-wise hybrid of MoBA and full attention. We investigate this strategy with a particular focus on its application during the supervised fine-tuning (SFT). The motivation for investigating this strategy stems from our observation that MoBA sometimes results in suboptimal performance during SFT, as shown in Figure 5b. We speculate that this may be attributed to the loss masking in SFT — prompt tokens are typically excluded from the loss calculation which can pose a sparse gradient challenge for sparse attention methods like MoBA (because it may hinder the backpropagation of gradients, which are calculated only from unmasked tokens, throughout the entire context). To address this issue, we propose a hybrid approach — switching the last several Transformer layers from MoBA to full attention, while the remaining layers continue to employ MoBA. As shown in Figure 5b and Figure 5c, this strategy can significantly reduce SFT loss.

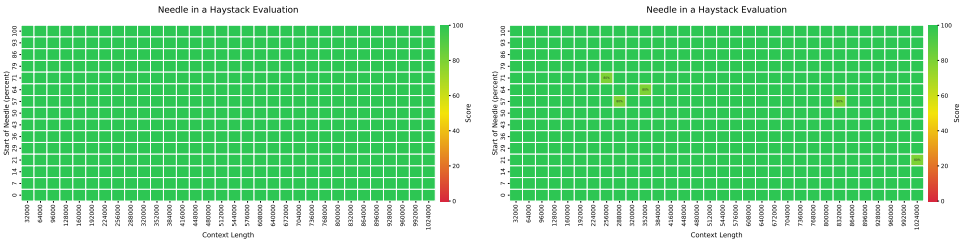
3.3 Large Language Modeling Evaluation

We assess MoBA across a variety of real-world downstream tasks, evaluating its performance in comparison to full attention models. For ease of verification, our experiments begin with the Llama 3.1 8B Base Model, which is used as the starting point for long-context pre-training. This model, termed Llama-8B-1M-MoBA, is initially trained with a context length of 128K tokens, and we gradually increase the context length to 256K, 512K, and 1M tokens during the continual pre-training. To ease this transition, we use position interpolation method [37] at the start of the 256K continual pre-training stage. This technique enables us to extend the effective context length from 128K tokens to 1M tokens. After completing the 1M continuous pre-training, MoBA is activated for 100B tokens. We set the block size to 4096 and the top-K parameter to 12, leading to an attention sparsity of up to $1 - \frac{4096\times 12}{1M} = 95.31\%$. To preserve some full attention capabilities, we adopt the layer-wise hybrid strategy — the last three layers remain as full attention, while the other 29 full attention layers are switched to MoBA. For supervised fine-tuning, we follow a similar strategy that gradually increases the context length from 32K to 1M. The baseline full attention models (termed Llama-8B-1M-Full) also follow a similar training strategy as shown in Figure 8, with the only difference being the use of full attention throughout the process. This approach allows us to directly compare the performance of MoBA with that of full attention models under equivalent training conditions.

In particular, across all evaluation tasks, we conduct two distinct inference configurations for MoBA. In the first configuration (termed as MoBA-Prefill-MoBA-Decode), we apply MoBA consistently throughout both the prefill and generation phases, maintaining the same settings as during training

Benchmark	MoBA-Prefill MoBA-Decode	MoBA-Prefill Full-Decode	Full
AGIEval [0-shot]	**0.5268**	0.5144	0.5146
BBH [3-shot]	0.6584	0.6573	**0.6589**
CEval [5-shot]	**0.6298**	0.6273	0.6165
GSM8K [5-shot]	**0.7324**	0.7278	0.7142
HellaSWAG [0-shot]	0.8262	0.8262	**0.8279**
Loogle [0-shot]	0.3333	**0.4209**	0.4016
Competition Math [0-shot]	**0.4454**	0.4254	0.4324
MBPP [3-shot]	0.5220	**0.5380**	0.5320
MBPP Sanitized [0-shot]	0.6498	**0.6926**	0.6615
MMLU [0-shot]	**0.4968**	0.4903	0.4904
MMLU Pro [5-shot][CoT]	**0.4343**	0.4295	0.4328
OpenAI HumanEval [0-shot][pass@1]	0.6890	0.6951	**0.7012**
SimpleQA [0-shot]	0.0460	0.0465	**0.0492**
TriviaQA [0-shot]	0.5610	**0.5673**	0.5667
LongBench @32K [0-shot]	0.4807	**0.4828**	0.4821
RULER @128K [0-shot]	0.7690	0.7671	**0.8031**

Table 1: Performance comparison of Llama-8B-1M-MoBA and Llamma-1M-Full. Llama-8B-1M-MoBA has two inference configurations: MoBA-Prefill-MoBA-Decode and MoBA-Prefill-Full-Decode. Details of the RULER benchmark can be found in Table 5 of Appendix.



(a) (b)

Figure 6: Performance of (a) LLama-8B-1M-MoBA-Prefill-Full-Decode and (b) LLama-8B-1M-MoBA-Prefill-MoBA-Decode on the Needle in the Haystack benchmark (upto 1M context length).

where the last three layers use full attention. For the second configuration (termed as MoBA-Prefill-Full-Decode), we employ MoBA exclusively during the prefill phase while switching to full attention during generation. As shown in Table 1, MoBA, across both its configurations, exhibits performance levels that are highly comparable to those of the Llama-8B-1M-Full model. It is particularly noteworthy that in the longest benchmark, RULER, where MoBA operates at a sparsity level of up to $1-\frac{4096\times 12}{128K}=62.5\%$, MoBA with both two inference configurations nearly matches the performance of full attention, with scores of 0.7690/0.7671 compared to 0.8031. For context lengths of up to 1M tokens, we evaluate the model using the traditional Needle in the Haystack benchmark. As shown in Figure 6, MoBA demonstrates satisfactory performance, while MoBA-Prefill-Full-Decode is slightly better than MoBA-Prefill-MoBA-Decode.

3.4 Efficiency and Scalability

The above experimental results show that MoBA achieves comparable performance not only regarding language model losses but also in real-world tasks. To further investigate its efficiency, we compare the forward pass time (prefill time) of the attention layer in two models trained in Section 3.3 — Llama-8B-1M-MoBA and Llama-8B-1M-Full. We mainly focus on the attention layer, as all other layers (e.g., FFN) have identical FLOPs in both models. As shown in Figure 2a, MoBA is more efficient than full attention across all context lengths, demonstrating a sub-quadratic computational complexity. In particular, it achieves a speedup ratio of up to 6.5x when prefilling 1M tokens. Furthermore, the end-to-end forward time of Llama-8B-1M-MoBA and Llamma-8B-1M-Full can be found in Table 6 of Appendix.

We also explore the length scalability of MoBA by gradually increasing the context length to 10 million tokens. To maintain a constant attention sparsity, we keep the top-k value and number of MoBA Block fixed while proportionally increasing the block size. To reach the 10M context length, we expanded tensor parallelism [38] toward the query head level, Specifically, we broadcast key and value tensors across distributed query heads, addressing GPU memory limitations while preserving computational efficiency. As shown in Figure 2b, MoBA demonstrates superior efficiency compared to standard Flash Attention when scaling to longer sequences. Specifically, at 10M tokens moba achieves a speedup ratio of 16x reduction in attention computation time. The inset graph in the Figure 2b, focusing on shorter sequences (32K to 512K), shows that even though both methods perform comparably at smaller scales, MoBA's computational advantage becomes increasingly evident as sequences grow longer, highlighting its particular strength in processing extremely long sequences.

Overall, the high efficiency of MoBA can be attributed to two key innovations: (1) the block sparse attention mechanism, and (2) the optimized implementation combining MoE and FlashAttention, as described in Section 2.1. These techniques effectively address the quadratic complexity limitation of full attention, reducing it to a more economical sub-quadratic scale.

4 Related Work

The development of efficient attention [39] mechanisms has been a critical area of research in the field of natural language processing, particularly with the rise of Large Language Models (LLMs). We review related work in this section:

Static Sparse Patterns: Significant efforts [40, 41, 42, 11, 43, 44, 45, 46, 47] have been dedicated to the design of static attention patterns in LLMs. Their choices of static attention patterns can encompass strided and fixed attention, window attention, global token attention, random attention, dilated attention, block sparse attention, or any combinations of them. In the realm of multimodal models, static sparse attention mechanisms have also been developed, such as axial attention [48] for 2D images and spatial-temporal attention [49] for 3D videos.

Dynamic Sparse Patterns: Different from static patterns, dynamic sparse attention techniques adaptively determine which tokens to attend. Reformer [50] and Routing Transformer [51] respectively employ locality-sensitive hashing (LSH) and K-means to cluster tokens, and attend to clusters rather than the full context. Memorizing Transformers [52] and Unlimiformer [53] dynamically attend to tokens selected by the k-nearest-neighbor (kNN) algorithms. CoLT5 [54] designs a routing modules to select the most important queries and keys. Sparse Sinkhorn Attention [55] learns to permute blocks from the input sequence, allowing dynamic block sparse attention computation. Notably, DeepSeek NSA [56] is a most recent work that studies dynamic sparse attention, whose selected attention branch is similar to MoBA's block selection mechanism. We believe that the emergence of dynamic sparse attention underscores its significance for the future of long-context LLMs

Training-free Sparse Attention. There are also strategies designed to incorporate sparse attention to model inference. During model inference, the complete prompt can be utilized for attention profiling, which allows for the exploration of more intricate sparse attention patterns. For instance, MoA [57], Minference [8], and SeerAttention [58] have investigated sparse attention configurations such as A-shape, vertical-slash, and dynamic block sparsity. Besides, considerable work has been dedicated to compressing and pruning the KV-cache to achieve a balance between the quality and speed. Notable efforts in this area include [59, 10, 60, 61, 12, 62].

Beyond Traditional Attention Architecture: Another line of research investigates novel architectures that deviate from the conventional attention mechanism. Studies in this domain have explored architectures that are inspired by Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), State Space Models (SSMs), or Linear Attention, such as [63, 64, 65, 66, 15, 67, 17, 68].

5 Conclusion

In this paper, we introduce Mixture of Block Attention (MoBA), a novel attention architecture inspired by the principles of Mixture of Experts (MoE) that aims to enhance the efficiency and scalability of large language models (LLMs) for long-context tasks. MoBA addresses the computational challenges

associated with traditional attention mechanisms by partitioning the context into blocks and employing a dynamic gating mechanism to selectively route query tokens to the most relevant KV blocks. This approach not only reduces computational complexity but also maintains model performance. Moreover, it allows for seamless transitions between full and sparse attention. Through extensive experiments, we demonstrated that MoBA achieves performance comparable to full attention while significantly improving computational efficiency. Our results show that MoBA can scale effectively to long contexts, maintaining low LM losses and high performance on various benchmarks. Additionally, MoBA's flexibility allows it to be integrated with existing models without substantial training cost, making it a practical continual pre-training solution for enhancing long-context capabilities in LLMs. In summary, MoBA represents a significant advancement in efficient attention, offering a balanced approach between performance and efficiency.

Limitations. Firstly, the router of MoBA is parameter-free. This design choice may limit its ability to capture more nuanced long-context patterns. Future work could explore the development of learnable router, which allows more sophisticated and adaptive block selection mechanisms. Secondly, this work focuses on text inputs. Extending MoBA to other modalities, such as images, audio, or multimodal inputs, could significantly broaden its applicability. Finally, investigating MoBA's potential for improving generalization in complex reasoning tasks, such as long chain-of-thought (CoT) reasoning, could provide further insights into its capabilities.

Broader Impact. The development of MoBA has the potential to significantly enhance the efficiency and scalability of large language models (LLMs), enabling more powerful and resource-efficient processing of long-context tasks. This advancement could positively impact various applications, such as natural language understanding, complex reasoning, and decision-making processes, leading to more capable AI systems. However, it is important to consider potential negative societal impacts, such as the misuse of enhanced language models for generating disinformation or the unintended consequences of deploying models with biases. To mitigate these risks, future work should focus on developing robust safeguards and ensuring transparency in model deployment and usage.

Acknowledgments and Disclosure of Funding

Mingxing Zhang is supported by Beijing Natural Science Foundation (L252014). Jiezhong Qiu is supported by NSFC 62306290.

References

[1] MoonshotAI. Kimi chat. `https://kimi.moonshot.cn/`, 2023.

[2] Anthropic. Introducing 100k context windows. `https://www.anthropic.com/news/100k-context-windows`, 2023.

[3] Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

[4] Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025.

[5] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[6] Melody Y Guan, Manas Joglekar, Eric Wallace, Saachi Jain, Boaz Barak, Alec Heylar, Rachel Dias, Andrea Vallone, Hongyu Ren, Jason Wei, et al. Deliberative alignment: Reasoning enables safer language models. *arXiv preprint arXiv:2412.16339*, 2024.

[7] A Waswani, N Shazeer, N Parmar, J Uszkoreit, L Jones, A Gomez, L Kaiser, and I Polosukhin. Attention is all you need. In *NIPS*, 2017.

[8] Huiqiang Jiang, Yucheng Li, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir H Abdi, Dongsheng Li, Chin-Yew Lin, et al. Minference 1.0: Accelerating pre-filling for long-context llms via dynamic sparse attention. *arXiv preprint arXiv:2407.02490*, 2024.

[9] Jake F Watson, Victor Vargas-Barroso, Rebecca J Morse-Mora, Andrea Navas-Olive, Mojtaba R Tavakoli, Johann G Danzl, Matthias Tomschik, Karl Rössler, and Peter Jonas. Human hippocampal ca3 uses specific functional connectivity rules for efficient associative memory. *Cell*, 188(2):501–514, 2025.

[10] Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.

[11] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[12] Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. Quest: Query-aware sparsity for efficient long-context llm inference. *arXiv preprint arXiv:2406.10774*, 2024.

[13] Di Liu, Meng Chen, Baotong Lu, Huiqiang Jiang, Zhenhua Han, Qianxi Zhang, Qi Chen, Chengruidong Zhang, Bailu Ding, Kai Zhang, et al. Retrievalattention: Accelerating long-context llm inference via vector retrieval. *arXiv preprint arXiv:2409.10516*, 2024.

[14] Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. *arXiv preprint arXiv:2405.21060*, 2024.

[15] Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, et al. Rwkv: Reinventing rnns for the transformer era. *arXiv preprint arXiv:2305.13048*, 2023.

[16] Bo Peng, Daniel Goldstein, Quentin Anthony, Alon Albalak, Eric Alcaide, Stella Biderman, Eugene Cheah, Xingjian Du, Teddy Ferdinan, Haowen Hou, et al. Eagle and finch: Rwkv with matrix-valued states and dynamic recurrence. *arXiv preprint arXiv:2404.05892*, 2024.

[17] Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*, 2023.

[18] Jean Mercat, Igor Vasiljevic, Sedrick Keh, Kushal Arora, Achal Dave, Adrien Gaidon, and Thomas Kollar. Linearizing large language models. *arXiv preprint arXiv:2405.06640*, 2024.

[19] Junxiong Wang, Daniele Paliotta, Avner May, Alexander M Rush, and Tri Dao. The mamba in the llama: Distilling and accelerating hybrid models. *arXiv preprint arXiv:2408.15237*, 2024.

[20] Aviv Bick, Kevin Li, Eric Xing, J Zico Kolter, and Albert Gu. Transformers to ssms: Distilling quadratic knowledge to subquadratic models. *Advances in Neural Information Processing Systems*, 37:31788–31812, 2025.

[21] Michael Zhang, Simran Arora, Rahul Chalamala, Alan Wu, Benjamin Spector, Aaryan Singhal, Krithik Ramesh, and Christopher Ré. Lolcats: On low-rank linearizing of large language models. *arXiv preprint arXiv:2410.10254*, 2024.

[22] Aonian Li, Bangwei Gong, Bo Yang, Boji Shan, Chang Liu, Cheng Zhu, Chunhao Zhang, Congchao Guo, Da Chen, Dong Li, et al. Minimax-01: Scaling foundation models with lightning attention. *arXiv preprint arXiv:2501.08313*, 2025.

[23] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[24] Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*, 2020.

[25] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022.

[26] Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. St-moe: Designing stable and transferable sparse expert models. *arXiv preprint arXiv:2202.08906*, 2022.

[27] Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*, 2024.

[28] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[29] Xuan Zhang, Cunxiao Du, Chao Du, Tianyu Pang, Wei Gao, and Min Lin. Simlayerkv: A simple framework for layer-level kv cache reduction. *arXiv preprint arXiv:2410.13846*, 2024.

[30] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

[31] Samyam Rajbhandari, Conglong Li, Zhewei Yao, Minjia Zhang, Reza Yazdani Aminabadi, Ammar Ahmad Awan, Jeff Rasley, and Yuxiong He. Deepspeed-moe: Advancing mixture-of-experts inference and training to power next-generation ai scale. In *International conference on machine learning*, pages 18332–18346. PMLR, 2022.

[32] Maxim Milakov and Natalia Gimelshein. Online normalizer calculation for softmax. *arXiv preprint arXiv:1805.02867*, 2018.

[33] Hao Liu and Pieter Abbeel. Blockwise parallel transformer for large context models. *arXiv preprint arXiv:2305.19370*, 2023.

[34] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[35] Chenxin An, Jun Zhang, Ming Zhong, Lei Li, Shansan Gong, Yao Luo, Jingjing Xu, and Lingpeng Kong. Why does the effective context length of llms fall short? *arXiv preprint arXiv:2410.18745*, 2024.

[36] Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, Madian Khabsa, Han Fang, Yashar Mehdad, Sharan Narang, Kshitiz Malik, Angela Fan, Shruti Bhosale, Sergey Edunov, Mike Lewis, Sinong Wang, and Hao Ma. Effective long-context scaling of foundation models. *arXiv preprint arXiv:2309.16039*, 2023.

[37] Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. Extending context window of large language models via positional interpolation. *arXiv preprint arXiv:2401.06066*, 2023.

[38] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

[39] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. arxiv. *arXiv preprint arXiv:2009.06732*, 2020.

[40] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

[41] Qipeng Guo, Xipeng Qiu, Pengfei Liu, Yunfan Shao, Xiangyang Xue, and Zheng Zhang. Star-transformer. *arXiv preprint arXiv:1902.09113*, 2019.

[42] Jiezhong Qiu, Hao Ma, Omer Levy, Scott Wen-tau Yih, Sinong Wang, and Jie Tang. Blockwise self-attention for long document understanding. *arXiv preprint arXiv:1911.02972*, 2019.

[43] Ankit Gupta and Jonathan Berant. Gmat: Global memory augmentation for transformers. *arXiv preprint arXiv:2006.03274*, 2020.

[44] Joshua Ainslie, Santiago Ontanon, Chris Alberti, Vaclav Cvicek, Zachary Fisher, Philip Pham, Anirudh Ravula, Sumit Sanghai, Qifan Wang, and Li Yang. Etc: Encoding long and structured inputs in transformers. *arXiv preprint arXiv:2004.08483*, 2020.

[45] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297, 2020.

[46] Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. Longt5: Efficient text-to-text transformer for long sequences. *arXiv preprint arXiv:2112.07916*, 2021.

[47] Jiayu Ding, Shuming Ma, Li Dong, Xingxing Zhang, Shaohan Huang, Wenhui Wang, Nanning Zheng, and Furu Wei. Longnet: Scaling transformers to 1,000,000,000 tokens. *arXiv preprint arXiv:2307.02486*, 2023.

[48] Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers. *arXiv preprint arXiv:1912.12180*, 2019.

[49] Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. Open-sora: Democratizing efficient video production for all, March 2024.

[50] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.

[51] Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. Efficient content-based sparse attention with routing transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68, 2021.

[52] Yuhuai Wu, Markus N Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. *arXiv preprint arXiv:2203.08913*, 2022.

[53] Amanda Bertsch, Uri Alon, Graham Neubig, and Matthew Gormley. Unlimiformer: Long-range transformers with unlimited length input. *Advances in Neural Information Processing Systems*, 36, 2024.

[54] Joshua Ainslie, Tao Lei, Michiel de Jong, Santiago Ontañón, Siddhartha Brahma, Yury Zemlyanskiy, David Uthus, Mandy Guo, James Lee-Thorp, Yi Tay, et al. Colt5: Faster long-range transformers with conditional computation. *arXiv preprint arXiv:2303.09752*, 2023.

[55] Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. Sparse sinkhorn attention. In *International Conference on Machine Learning*, pages 9438–9447. PMLR, 2020.

[56] Jingyang Yuan, Huazuo Gao, Damai Dai, Junyu Luo, Liang Zhao, Zhengyan Zhang, Zhenda Xie, YX Wei, Lean Wang, Zhiping Xiao, et al. Native sparse attention: Hardware-aligned and natively trainable sparse attention. *arXiv preprint arXiv:2502.11089*, 2025.

[57] Tianyu Fu, Haofeng Huang, Xuefei Ning, Genghan Zhang, Boju Chen, Tianqi Wu, Hongyi Wang, Zixiao Huang, Shiyao Li, Shengen Yan, et al. Moa: Mixture of sparse attention for automatic large language model compression. *arXiv preprint arXiv:2406.14909*, 2024.

[58] Yizhao Gao, Zhichen Zeng, Dayou Du, Shijie Cao, Hayden Kwok-Hay So, Ting Cao, Fan Yang, and Mao Yang. Seerattention: Learning intrinsic sparse attention in your llms. *arXiv preprint arXiv:2410.13276*, 2024.

[59] Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems*, 36, 2024.

[60] Matanel Oren, Michael Hassid, Yossi Adi, and Roy Schwartz. Transformers are multi-state rnns. *arXiv preprint arXiv:2401.06104*, 2024.

[61] Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. Model tells you what to discard: Adaptive kv cache compression for llms. *arXiv preprint arXiv:2310.01801*, 2023.

[62] Yi Lu, Xin Zhou, Wei He, Jun Zhao, Tao Ji, Tao Gui, Qi Zhang, and Xuanjing Huang. Longheads: Multi-head attention is secretly a long context processor. *arXiv preprint arXiv:2402.10685*, 2024.

[63] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pages 5156–5165. PMLR, 2020.

[64] Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. Hyena hierarchy: Towards larger convolutional language models. In *International Conference on Machine Learning*, pages 28043–28078. PMLR, 2023.

[65] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

[66] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

[67] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

[68] Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. *arXiv preprint arXiv:2406.06484*, 2024.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We show Mixture of Block Attention (MoBA), an innovative approach that applies the principles of Mixture of Experts (MoE) to the attention mechanism, achieves significant advancements in efficient attention computation for LLMs.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the main limitations in Section 5.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: This paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We list the pseudocode of MoBA in Algorithm 1 and include the code in the Supplementary Material. Implementation details and experimental settings are described in Section 2.1 and Section 3 respectively.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

Justification: We include our code in Supplementary Material.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide the training and test details in Section 3 and Appendix Section A.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Our experiments follow the standard protocol in long-context LLM tasks, and thus did not report statistical significance.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We confirm that all necessary computational resource information is provided in Section A.1 and Section A.3, including the GPU types, cluster network architecture, and training durations for our various experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We confirm the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the broader impacts of our work in Section 5

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks, because the main contribution of this work is novel architecture, therefore we do not release data or models.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We properly cite the original paper that produced the code packages or datasets that are used in this paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Our new assets include the code of this work, which will be released under The MIT License (MIT).

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: This research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Technical Appendices and Supplementary Material

A.1 Scaling Law

For our validation experiments in the Section 3.1 and Section 3.2, we utilized a distributed computing infrastructure consisting of 8 server nodes, each equipped with 8 NVIDIA H800 GPUs (64 GPUs in total). Within each node, the GPUs were interconnected via NVLink for high-bandwidth intra-node communication, while Remote Direct Memory Access (RDMA) was employed for efficient inter-node communication across the cluster. The learning rate schedule followed the optimal decay pattern established during the training phase. Each experiment was terminated upon completion of the expected number of training tokens.

No-Emb Model Param	Head	Layer	Hidden	Training Token	Block size	TopK
545M	14	14	1792	10.8B	512	3
822M	16	16	2048	15.3B	512	3
1.1B	18	18	2304	20.6B	512	3
1.5B	20	20	2560	27.4B	512	3
2.1B	22	22	2816	36.9B	512	3

Table 2: Configuration of Scaling Law Experiments

L(C)	MoBA	Full
LM loss (seqlen=8K)	$2.625 \times C^{-0.063}$	$2.622 \times C^{-0.063}$
Trailing LM loss (seqlen=32K, last 2K)	$1.546 \times C^{-0.108}$	$1.464 \times C^{-0.097}$

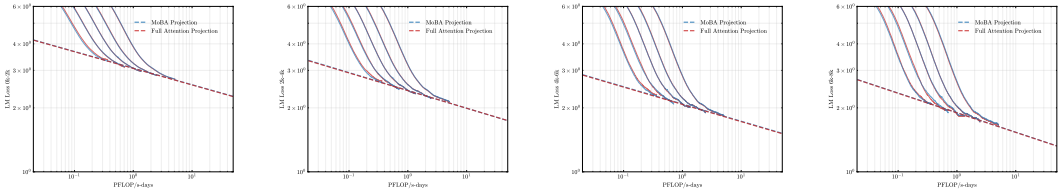
Table 3: Fitted scaling law curve.

A.2 Long Context Scalability

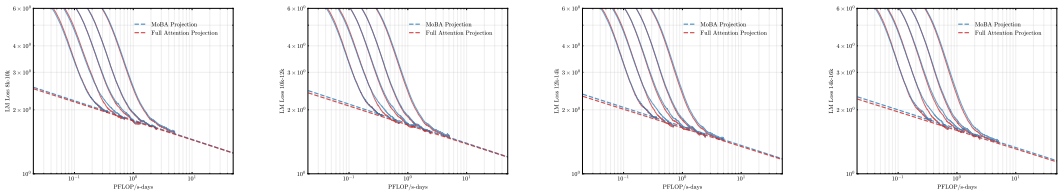
To address the bias in natural data distribution that favors short contexts, we strategically segmented the overall sequences into discrete segments based on their actual positions. For example, the segment spanning positions 30K-32K exclusively reflects losses associated with documents exceeding 30K context lengths and also masks the positions from 30K to 32K. This approach ensures a more balanced and representative evaluation across different context lengths. In our exploration of long-context scalability, we made a pivotal discovery: the trailing tokens account for the majority of the performance discrepancy between the full context baseline and the newly proposed sparse attention architectures. Consequently, we streamlined the long-context scaling process by focusing on trailing token scaling. This not only simplifies the computational requirements but also significantly enhances the efficiency and effectiveness of investigating long-context scenarios. This finding holds substantial implications for the development of more efficient and scalable attention mechanisms in the future.

A.3 Large Language Modeling Evaluation

For our validation experiments in the Section 3.3, we utilized a distributed computing infrastructure consisting of 128 GPU server nodes, each equipped with 8 NVIDIA H800 GPUs (1024 GPUs in total). Within each node, the GPUs were interconnected via NVLink for intra-node communication, while Remote Direct Memory Access (RDMA) was employed for inter-node communication. We trained our model following the recipe outlined in Figure 8, requiring over 30 days to complete training recipe of the Full model with a context length of 1M tokens. The MoBA variant achieved a significantly reduced training time by leveraging the same pretrained checkpoint established prior to the 1M context length training phase.

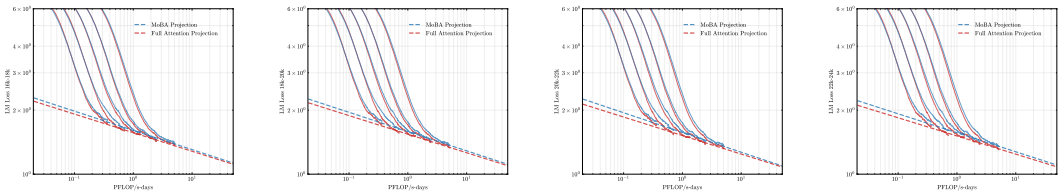


(a) Scaling law (0-2k) (b) Scaling law (2-4k) (c) Scaling law (4-6k) (d) Scaling law (6-8k)

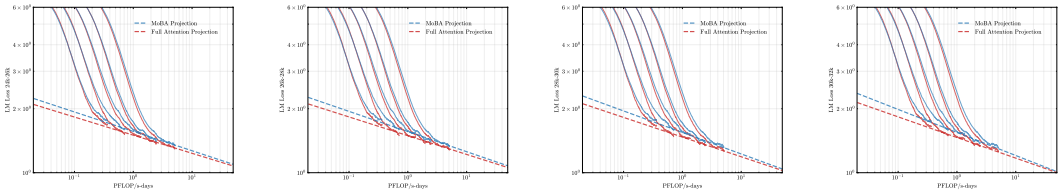


(e) Scaling law (8-10k) (f) Scaling law (10-12k) (g) Scaling law (12-14k) (h) Scaling law (14-16k)

Figure 7: Scaling laws for positions 0-16k



(i) Scaling law (16-18k) (j) Scaling law (18-20k) (k) Scaling law (20-22k) (l) Scaling law (22-24k)



(m) Scaling law (24-26k) (n) Scaling law (26-28k) (o) Scaling law (28-30k) (p) Scaling law (30-32k)

Figure 7: Scaling laws for positions 16-32k

LM Loss Position Range	MoBA	Full
0K - 2K	$3.075 \times C^{-0.078}$	$3.068 \times C^{-0.078}$
2K - 4K	$2.415 \times C^{-0.084}$	$2.411 \times C^{-0.083}$
4K - 6K	$2.085 \times C^{-0.081}$	$2.077 \times C^{-0.081}$
6K - 8K	$1.899 \times C^{-0.092}$	$1.894 \times C^{-0.092}$
8K - 10K	$1.789 \times C^{-0.091}$	$1.774 \times C^{-0.089}$
10K - 12K	$1.721 \times C^{-0.092}$	$1.697 \times C^{-0.087}$
12K - 14K	$1.670 \times C^{-0.089}$	$1.645 \times C^{-0.088}$
14K - 16K	$1.630 \times C^{-0.089}$	$1.600 \times C^{-0.087}$
16K - 18K	$1.607 \times C^{-0.090}$	$1.567 \times C^{-0.087}$
18K - 20K	$1.586 \times C^{-0.091}$	$1.542 \times C^{-0.087}$
20K - 22K	$1.571 \times C^{-0.093}$	$1.519 \times C^{-0.086}$
22K - 24K	$1.566 \times C^{-0.089}$	$1.513 \times C^{-0.085}$
24K - 26K	$1.565 \times C^{-0.091}$	$1.502 \times C^{-0.085}$
26K - 28K	$1.562 \times C^{-0.095}$	$1.493 \times C^{-0.088}$
28K - 30K	$1.547 \times C^{-0.097}$	$1.471 \times C^{-0.091}$
30K - 32K	$1.546 \times C^{-0.108}$	$1.464 \times C^{-0.097}$

Table 4: Loss scaling with different positions

Ruler Tasks	MoBA-Prefill MoBA-Decode	MoBA-Prefill Full-Decode	Full
Avg.	0.769	0.767	0.803
niah_multikey_1	0.982	0.982	1.000
niah_multikey_2	0.762	0.754	0.992
niah_multikey_3	0.944	0.948	0.994
niah_multiquery	0.819	0.812	0.765
niah_multivalue	0.558	0.559	0.571
niah_single_1	1.000	1.000	1.000
niah_single_2	0.992	0.984	1.000
niah_single_3	0.966	0.958	0.982
ruler_cwe	0.009	0.010	0.010
ruler_fwe	0.882	0.881	0.905
ruler_qa_hotpot	0.510	0.518	0.542
ruler_qa_squad	0.604	0.602	0.753
ruler_vt	0.968	0.965	0.926

Table 5: Ruler Test Result @128K

Context Length	Full Prefill Time (sec)	MoBA Prefill Time (sec)	Speed-up
256 K	5.67	3.55	1.60×
512 K	19.59	8.40	2.33×
1 M	73.97	22.15	3.34×

Table 6: End-to-end Efficiency of Llama-8B-1M-MoBA and Llamma-1M-Full (Note that Llama-8B-1M-MoBA is a hybrid model with the last 3 layers to be full attention).

#Blocks	4	8	32	128	512
LM loss	2.48	2.48	2.47	2.53	2.60

Table 7: The relationship between attention sparsity and LM loss.

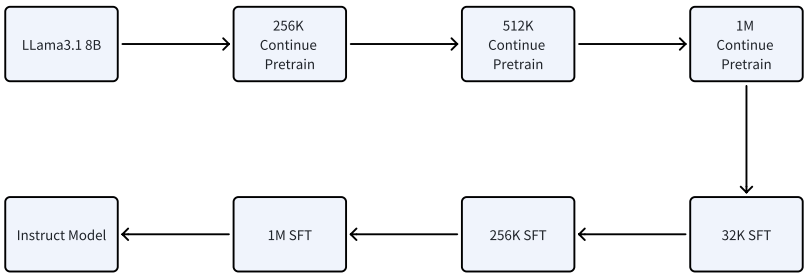


Figure 8: The continual pre-training and SFT recipes.