Correlation-Aware Selection and Merging for Efficient Fine-Tuning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Modeling long sequences is crucial for various large-scale models, yet extending existing models to handle longer sequences poses technical and resource challenges. In this paper, we propose a more efficient fine-tuning approach based on LongLora and full-scale fine-tuning, achieving reduced GPU memory usage and shorter fine-tuning times. Specifically, we employ correlation-aware selection and merging mechanisms during fine-tuning to enable sparse attention, fine-tuning fewer parameters for more efficient extension of longer sequences. Furthermore, by introducing recursion, we achieve evaluation and extension of even longer sequences. Finally, by configuring the selection quantity and attention scope within our attention mechanism, we achieve varying degrees of fitting, indicated by controllable loss curves. Experimental results on the Llama2-7B model demonstrate the effectiveness of our proposed approach. With just 2 $\times$ A100 40G, we achieve 32K fine-tuning on Llama2-7B, and the passkey results of the 16K fine-tuned model remain at 100% for sequences up to 32K in length. Moreover, introducing recursion maintains a stable perplexity score on the 2M-length PG19 validation set.

1 Introduction

In various natural language processing (NLP) tasks, such as document-level sentiment analysis[1], long document summarization[2], and code generation[3], the ability to model long-sequence dependencies effectively is crucial. This capability allows for capturing complex relationships over sequences spanning hundreds or thousands of tokens, which is essential for tasks where contextual information is dispersed. For example, accurately summarizing a lengthy document or generating coherent code relies on understanding dependencies that go beyond adjacent tokens. Consequently, extending the context window enables LLMs to perform tasks that shorter context windows cannot handle and potentially enhances performance across a variety of NLP tasks.

However, extending the context window presents numerous challenges. Longer sequences demand significantly more memory and computational resources, leading to slower training and inference times and higher resource consumption. Moreover, capturing long-range dependencies and using large numbers of tokens in autoregressive models can result in slower convergence, potentially due to underfitting caused by the large number of tokens involved in the autoregressive process.

To address these challenges, current research often adopts a strategy of pretraining on short sequences[4][5] followed by efficient fine-tuning using positional interpolation for long sequence extension. Recent works have achieved promising results, such as LongLora[6], which combines sparse attention with improved LoRA[7] (Low-Rank Adaptation) to extend Llama2-7B[8] to 100K tokens using 8$\times$A100 80GB GPUs. However, our analysis indicates that LongLora's sparse attention does not fully exploit sparsity, and its efficient fine-tuning can be further optimized to use fewer parameters, thereby requiring less memory and computational resources for similar extensions.

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

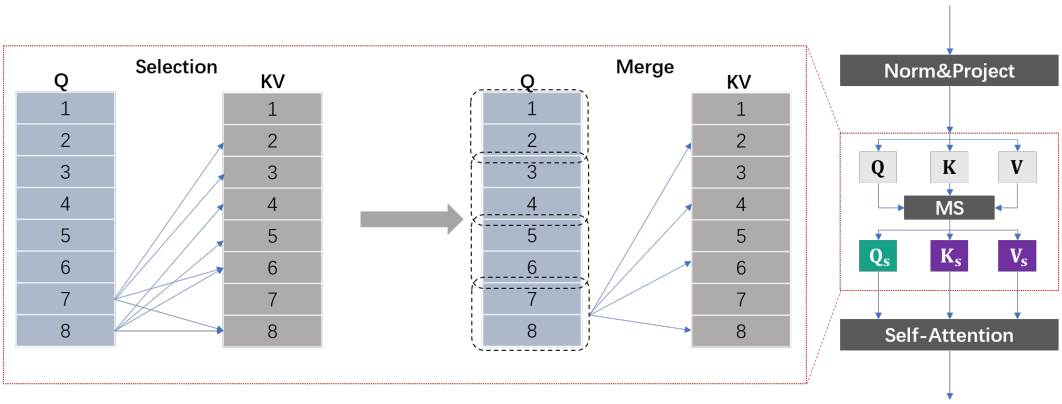

Figure 1: **Overview of Merging and Selection Attention Mechanism (MS Attention).** The MS Attention mechanism involves two main steps. In the first step, the QKV tensors are split into regions, and a single token is used to represent each region. Subsequently, the regional representatives are used to compute dot products or other similarity measures to select the most relevant KV regions for each Q region. For example, Q regions 7 and 8 select KV regions $\{kv6, kv8, kv2, kv3\}$ and $\{kv6, kv8, kv4, kv5\}$ respectively. In the second step, for each Q region, tokens are merged with their adjacent or related tokens. The union of the selected KV regions is taken, and the top-n regions are chosen. For example, combine Q regions 7 and 8, along with their selected KV regions, resulting in $\{kv6, kv8, kv2, kv4, kv3, kv5\}$. To ensure tensor consistency, we select the top k regions from the merged set. If $k = 4$, the final selection is $\{kv6, kv8, kv2, kv4\}$. Finally, each merged Q region performs self-attention with its selected relevant KV regions.

In this paper, we propose a new approach leveraging correlation selection and merging mechanisms to achieve more efficient sparse attention, along with fine-tuning strategies that use fewer parameters. This approach allows for more efficient and effective context extension. Our method demonstrates that, with only 2×A100 40GB GPUs, using DeepSpeed ZeRO-2[9], we can fine-tune Llama2-7B on sequences up to 32K tokens, achieving performance on par with LongLora's 64K token fine-tuning on the passkey task. Furthermore, by leveraging DeepSpeed ZeRO-3 and Positional Interpolation[10], we successfully fine-tuned LLaMA2-7B with a context length of 50K tokens. This fine-tuning extended the model's maximum context length to 100K tokens, demonstrating that our approach can achieve the same sequence length extension as LongLoRA[6] while utilizing significantly fewer computational resources. This result highlights the efficiency and scalability of our method in optimizing models for long-sequence processing tasks. By further enhancing the positional interpolation method, our approach enables significant extension capabilities. Specifically, fine-tuning at a length of 16K allows us to maintain a 100% pass rate on the passkey task even when the sequence length is extended to 80K. Moreover, when combined with the NTK positional encoding method, fine-tuning at 16K results in a seamless extension to 256K, achieving at least a 16-fold resource efficiency improvement compared to vanilla Attention. Additionally, we have incorporated the LongBench[11] dataset and perplexity on PG19[12] and proof-pile measurements to validate and ensure the robustness of our method.

Finally, by incorporating recursive methods[13][14], our attention variant is able to maintain stable perplexity even when processing sequences of up to 2 million tokens on the PG19[12] validation set. Additionally, our sparse attention mechanism inherently supports selecting varying numbers of tokens for autoregressive modeling, enabling a degree of control over the fitting process, which accelerates convergence and determines the final loss convergence level.

In summary, our approach offers a more resource-efficient method for extending the sequence length capabilities of LLMs, by correlation refined sparse attention and improved fine-tuning techniques. Our approach not only reduces computational and memory overhead but also enhances performance on long-sequence tasks, pushing the boundaries of what is achievable with LLMs in NLP or other applicatoin domains.

2 Related Work

Efficient Attention Mechanisms To fully exploit the inherent sparsity and positional relationships between tokens, a significant body of research has focused on developing efficient attention mechanisms. These mechanisms reduce the computational complexity of attention operations by focusing on a subset of tokens at each step, thus processing long sequences more efficiently or reducing resource consumption. Existing methods first preserve local features and then use various strategies to attend to more distant tokens. For instance, BigBird[15] and Performer[16] use random patterns, Longformer[17] and DETR[18] use fixed patterns, while Biformer[19] and Routing Transformer[20] utilize relevance routing mechanisms. Our proposed relevance selection and merging mechanism adapts flexibly to various scenarios and is compatible with FlashAttention2[21], achieving more efficient and general sparse attention.

Positional Encoding Another research direction aimed at extending sequence length in LLMs focuses on positional encoding techniques, such as positional interpolation and extrapolation. Most pretrained models are trained on fixed-length sequences with fixed positional encodings, leading to performance degradation when extended to unknown positions. Therefore, numerous studies have analyzed the impact of positional encodings and modified them through interpolation or extrapolation to extend to longer sequences. For example, Position Interpolation[10], NTK-aware[22], Yarn[23], and LongRopE[24] mitigate the effects of pretrained positional encodings by using interpolation with different scales based on frequency importance, effectively extending sequence modeling lengths.

Efficient Fine-Tuning Efficient fine-tuning of LLMs has become a critical research direction, especially for handling long sequences. Techniques like Input-tuning[25] and LoRA[7] have shown significant promise in this area. Building on the LongLora[6] method, we further optimize by using fewer parameters for fine-tuning, aiming to reduce the computational and memory overheads associated with fine-tuning large models on extended sequences while maintaining their performance on downstream tasks.

Recursive Methods To further extend sequence length and even achieve infinite generation, many recent studies have adopted recursive methods. These methods apply attention mechanisms recursively, generating longer sequences by iteratively attending to subsets of tokens. By recursively expanding the attention range, these methods can generate sequences that exceed the model's maximum sequence length, thereby breaking the sequence generation limits in LLMs. Examples include Infini-Attention[14], Transformer-XL[13], and SSM[26][27][28][29], which utilize recursive methods during training and inference to achieve extremely long sequence modeling.

3 Preliminary

3.1 Transformer

The Llama2 model used in this paper is based on the Transformer architecture, which consists of the core modules self-attention and MLP. The computation process of self-attention[30] is as follows:

$$O = softmax(QK^T)V$$

where Q, K, and V are obtained from X using embedding weights W_q, W_k, and W_v respectively. The final output O is then passed through W_o to obtain the final output of the attention. Subsequently, the entire Transformer operation is completed through the MLP.

3.2 LoRA

Low-Rank Adaptation (LoRA)[7] is an efficient model fine-tuning method designed to reduce the computational resources and storage requirements when fine-tuning large pretrained models.

The core idea of LoRA is to decompose the weight matrix W of the pretrained model into two low-rank matrices, A and B. This decomposition is represented as $W = W_0 + \Delta W$, where W_0 is the original weight matrix and $\Delta W = AB$ is the adjustment matrix obtained through low-rank decomposition. The matrices A and B have a rank of r, which is typically much smaller than the dimensions of W. During fine-tuning, only the parameters of matrices A and B need to be adjusted, while the original weight matrix W_0 remains unchanged.

The LoRA experiments on various large language models (such as GPT-3[31] and BERT[32]) have shown that the method can significantly reduce the computational resources and storage requirements while maintaining or even improving the model's performance.

3.3 Recurrent Methods

For Transformer models, sequences can be divided into blocks and processed recursively without loss. For instance, the input X can be divided into $X_1, X_2, \ldots, X_i, \ldots, X_n$. The output of the i-th block is:

$$O_i = \sum_{j=0}^{i} \left(\frac{s_{i,j}}{s_{i,j} + ss_{i,j-1}} \mathbf{softmax}(Q_i K_j^T) V_j \right)$$

where $ss_{i,k-1} = s_{i,k-1} + ss_{i,k-2}$, $ss_{i,k-1} = \sum_{j=0}^{k-1} s_{i,j}$ and $s_{i,j} = sum(\exp(Q_i K_j^T), dim = -1)$.

Since the above computation of the Attention process requires two nested layers of loops, many studies use hidden states or compression methods to represent the previous state in the recursion. Their form is:

$$o_i = \frac{cc_{i-1}}{c_i + cc_{i-1}} \mathbf{softmax}(q_i k_{t-1}^{hT}) v_{t-1}^h + \frac{c_i}{c_i + cc_{i-1}} \mathbf{softmax}(q_i k_t^T) v_t$$

where $k_t^h = f(k_{t-1}^h, k_t)$ and $v_t^h = f(v_{t-1}^h, v_t)$. $cc_i = cc_{i-1} + c_i, c_0 = sum(exp(q_0 k_0^T))$.

By integrating these approaches, our work achieves efficient and effective extension of model capabilities for handling long sequences, demonstrating significant advancements in NLP tasks requiring extensive context modeling and sequence generation.

4 Methods

4.1 MS: Merge selection

The following is a pseudo-code and the theory in appendix A.2:

Algorithm 1 Selection and Merging Process

Input: Q,K,V tensors of shape (b, h, n, d), segment size s, top$k$, top$n$, $merges$
Output: Final indices kvm
procedure SELECTION ▷ Selection
    Partition Q tensors into regions: $(b, h, n, d) \to (b, h, n_{sq}, s_q, d)$
    Partition KV tensors into regions: $(b, h, n, d) \to (b, h, n_{sk}, s_k, d)$
    Represent each region with a semantic or averaged compressed token: $Q'_s \in (b, h, n_{sq}, d), K'_s \in (b, h, n_{sk}, d)$
    Compute relevance between Q'_s and K'_s using dot product or other similarity metrics
    Apply mask to prevent information leakage
    Obtain indices of top k most relevant K_s regions: $selectindx \in (b, h, n_{sq}, \text{topk})$
end procedure
procedure MERGING ▷ Merging
    Merge Q_s regions: $(b, h, ns, s, d) \to (b, h, n_{ms}, m \cdot s, d), m = merges$
    Split, permute, and merge $selectindx$: $mselectindx \in (b, h, n_{ms}, m, \text{topk}) \to (b, h, n_{ms}, \text{topk}, m) \to (b, h, n_{ms}, \text{topk} \cdot m)$
    Perform unique operation while preserving relevance order
    Select top n indices: $qmselectindx$ representing final set of relevant K_s regions
    Select relevant KV_s regions by $qmselectindx$: $kvm \in (b, h, n_{sq}, topn, d)$
end procedure

We propose a method for implementing more general and efficient sparse attention through correlation selection and merging mechanisms, as shown in Figure 1. This method consists of two main steps:

selection of relevant regions and merging of these regions. The proposed method consists of two main steps: selection and merging.

Step 1: Selection. First, the Q, K, and V tensors with shape (b, h, n, d) are segmented into regions, resulting in tensors Q_s with shape (b, h, n_{sq}, s_q, d), K_s, and V_s with shape (b, h, n_{sk}, s_k, d), where s_q and s_k denote the segment size. Each region is represented by a semantic token or an average compressed token, yielding Q'_s and K'_s. A dot product or another similarity metric is then applied between Q'_s and K'_s to analyze the relevance between Q_s and K_s regions. To prevent information leakage, we apply a mask and control the number of selected tokens. This process results in the indices of the top-k most relevant K_s regions for each Q_s region, denoted as $selectindx$ with shape $(b, h, n_{sq}, topk)$, where n_s is the number of Q_s regions and $topk$ is the number of selected K_s and V_s regions.

Step 2: Merging. In the merging step, the indices obtained from the first step are combined, effectively merging the selected regions. Q_s is merged according to a specified number of segments $merges$, resulting in Q_{ms} with shape $(b, h, n_{ms}, m \cdot s, d)$. Similarly, $selectindx$ is split, permuted, and merged into $mselectindx$ with shape $(b, h, n_{ms}, topk, m) \rightarrow (b, h, n_{ms}, topk \cdot m)$. Due to previous topk operation, the permutation can ensure that each row is sorted by relevance, with the first m indices corresponding to the most relevant K_s region for each of the merged Q_s regions, the next m indices corresponding to the second most relevant K_s region, and so on. After obtaining the merged indices mselectindx, performs unique operation while maintaining relevance order. The top-n indices, denoted as qmselectindx, are selected as the final indices corresponding to the merged Q regions and their relevant K_s regions.

The reason for this merging step is to ensure that each token can attend to enough K and V tokens, and using larger Q regions sharing more K and V tokens is more efficient than smaller regions with fewer K and V tokens. Another reason for using the merging strategy is that relevance selection within smaller regions tends to be more precise.

While sharing similarities with Biformer[19] and Routing Transformer[20], our method distinguishes itself through its flexible representation of regions, the adaptability and compressibility in selecting region sizes, and the additional versatility of the merger mechanism. This allows for a more nuanced and effective approach to handling attention mechanisms in large-scale language models, enhancing their performance and efficiency across various tasks and contexts. For example, by flexibly setting the size of the Q, K, and V partitioned regions in these two steps, the range of KV attended by Q tokens, the number of selected K and V tokens, the size of the merged regions and the number of retained K and V tokens, our attention mechanism can cover almost all scenarios.

As analyzed in Appendix A.2, our algorithm is able to cover most of the autoregressive methods using a subset of KV tokens, examples include Biformer, Landmark Attention, Routing Transformer and Swin Transformer and their variants, etc.

4.2 Reduced LongLora

We propose a method that selectively fine-tunes only the W_k and W_o weights, achieving results nearly identical to fine-tuning the entire attention mechanism. Specifically, since $QK^T = XW_qW_k^TX^T$, updating only W_k yields $W_qW_k^T$, effectively replicating the effect of simultaneously updating W_k and W_q. This approach is particularly effective when W_q is full rank. Similarly, fine-tuning W_o follows the same rationale, as the final output is a linear mapping of W_vW_o.

Additionally, since query tokens can be considered well-fitted through extensive training, learning the linear mapping for their corresponding key tokens is reasonable. Furthermore, as the attention mechanism becomes heterogeneous, updating the final classification head is a direct approach. However, considering the large number of parameters, altering the initial embedding[25] to achieve a certain degree of equivalence is also a feasible consideration.

4.3 MS recursion

To seamlessly use recursive computation for the entire sequence attention, we set each token to attend to a certain range and expand its view progressively with each layer, thereby achieving effective modeling of long sequences.

We utilize a multi-scale MS attention mechanism where each scale attends to different ranges of key tokens, and even within the same scale, different ranges can be set for the key tokens. Since the attention range is fixed within each layer and all operations before computing Softmax are linear, a compress-then-compute strategy can be employed. Specifically, the previous state is stored recursively across all fixed ranges, compressed according to the scale, ultimately achieving long sequence generation with fixed memory size.

For example, we divide the sequence into chunks of length 16,384, denoted as X_i. We then apply three scales of Attention with compression ratios of {1, 2, 4} and attention ranges of {4096, 8192, 16384}. Each of these attention ranges is compressed according to its respective ratio, resulting in dimensions {4096, 4096, 4096}, represented as X_i^{p1}, X_i^{p2}, and X_i^{p4}. These compressed representations are concatenated to form X_i^p. If the size of X_i^p exceeds the maximum attention range of 16,384, we use the uncompressed X_i directly. This concatenated or original representation is then used as the hidden state input to the next block. The computations for the next block are as follows:

$$K_i = X_i^p W_k, V_i = X_i^p W_v, Q_{i+1} = X_{i+1} W_q, K_{i+1} = X_{i+1} W_k, V_{i+1} = X_{i+1} W_v,$$

The output of the next block is then calculated by, cc_i is same as 3.3:

$$O_{i+1} = \frac{cc_i}{c_{i+1} + cc_i} \text{softmax}(Q_{i+1} K_i^T) V_i + \frac{c_{i+1}}{c_{i+1} + cc_i} \text{softmax\_withmask}(Q_{i+1} K_{i+1}^T) V_{i+1}.$$

5 Experiment

5.1 Experimental Setup

Our experiments were conducted on 2 $\times$ A100 40GB GPUs, using the Llama2-7B[8] model and Mistral-7B-v0.1 with the attention mechanism replaced by our MS Attention as described in Section 4.1. The training approach used the efficient fine-tuning method described in Section 4.2, with position interpolation applied. Finally, to extend to longer sequence lengths, we employed the recursive method described in Section 4.3.

Fine-tuning Steps and Parameters: The fine-tuning parameters included the linear transformations for key and output (W_k and W_o), as well as the embedding and normalization parameters. The training approach used an autoregressive method, where the objective was to generate the next token. The loss function used was cross-entropy, and the optimizer was AdamW[33] with parameters β_1 = 0.9, β_2 = 0.95 and $lr = 1e-4$. Both the batch size and gradient accumulation were set to 1, and the model was trained for 3000 steps.

Dataset: The dataset used for fine-tuning was the Redpajama[34] dataset, which is the same dataset used in LongLora. Evaluation was performed using the widely used long text dataset PG19[12] and proof-pile, with perplexity as the evaluation metric. Additionally, the model's performance was assessed using the passkey task[35] and LongBench[11].

5.2 Main Results

5.2.1 Long Sequence Language Modeling

For the long sequence language modeling task, we evaluated our fine-tuned model on the PG19 dataset, as show in Table 1. The experimental results show that we successfully extended the sequence length to the length used during fine-tuning. Despite the significantly smaller amount of fine-tuning data used in our approach (LongLoRA: 64,000, ours: 6,000), our model's perplexity remains competitive with LongLoRA[6]. Specifically, when fine-tuned on sequences of length 16K, our model's perplexity is approximately 0.15 higher than LongLoRA. However, when fine-tuned on sequences of length 32K, our model's perplexity is approximately 0.15 lower than LongLoRA. Furthermore, by training with a smaller number of tokens and leveraging MS Attention during evaluation, we achieve even lower perplexity. For example, on the 16K sequence, our method achieved better perplexity than others.

Table 1: **Evaluation Results on Different Context Lengths with Llama2-7B.** m128: merge 128 regions of Q. s512: select top512 regions of KV.

Model	Training Context Length w/wo setting	Evaluation Context Length					
		2048	4096	8192	16384	32768	65536
LongLora	16384	7.65	7.28	7.02	6.86	-	-
	32768	8.29	7.83	7.54	7.35	7.22	-
LongRoPE	131072	-	-	6.98	-	-	**6.59**
Ours (eval: full Attention)	16384-m128-s512	7.61	7.42	7.17	7.03	7.06	9.52
	32768-m128-s512	8.03	7.64	7.37	7.20	7.13	7.30
Ours	16384-m2-s16	7.66	**6.93**	**6.41**	**6.14**	-	-
	16384-ss512	-	-	6.21	2.57	-	-

Table 2: **Training Time and Memory Usage for Different Sequence Lengths.** m128: merge 128 regions of Q. s512: select top512 regions of KV.

Training Setting	16K (stage2)		32K (offload optimizer)		50K (stage3)	
	Train Hours	Memory (GB)	Train Hours	Memory (GB)	Train Hours	Memory (GB)
LoRA	14.0	34.7	-	OOM	-	OOM
LongLora	11.3	34.6	-	OOM	-	OOM
Ours (m128-s512)	6.1	**33.5**	12.1	**37.5**	19.7	**39.1**
Ours (m16-s64)	**5.2**	**33.3**	**11.1**	38.0	**18.0**	39.2

5.2.2 Memory Usage and Fine-tuning Time

Our experiments using 2 $\times$ A100 40GB GPUs showed that memory usage and fine-tuning time were lower than LongLora, with the advantage becoming more apparent as sequence length increased, as show in Table 2. For example, during 32K length fine-tuning, our method could use stage2+offload optimizer for fine-tuning, while LongLora would OOM. When training with a sequence length of 16K, our memory usage is only 33.5GB, which is significantly less compared to the memory requirements of LongLoRA using 8 GPUs. Moreover, our method achieves comparable or even superior performance with significantly less training time.

5.2.3 Passkey Task

In the Retrieval-based Evaluation, we used the passkey task for evaluation. Models trained with 16K, 32K, and 50K sequences could naturally extend to double their lengths, as show in figure 3. For instance, the model trained with 50K sequences maintained 100% accuracy when evaluated on 100K sequences the limits of LongLora in Llama2-7B. Thus, with only 2 $\times$ A100 40GB GPUs, we achieved LongLora's accomplishments regarding Llama2-7B extension.

Table 3: Results on the Passkey Task with Position Interpolation, where "a/b" denotes the accuracy without changing the interpolation ratio (a) and with changing the interpolation ratio (b).

Model	2k-16k	32k	36k	40k	48k	64k	72k	80k	100k
Llama7B-MS-16K	1.0	0.9	0.4/0.8	0.2/0.8	0.2/0.7	0/0.6	0.0	0.0	0.0
Mistral7B-MS-16K	1.0	1.0	0.4/0.9	0.3/0.8	0.1/0.7	0/0.4	0.0	0.0	0.0
Llama7B-MS-32K	1.0	1.0	1.0	1.0	1.0	0.9	-/0.8	-/0.6	0.0
Llama7B-MS-50K	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.9
LongLora-32K	1.0	1.0	0/0.7	0/0.7	0/0.2	0.0	0.0	0.0	0.0

Table 4: Performance of Llama7B with Position Randomization and Position Interpolation

Model	2K-32K	40K	48K	56K	64K	80K	100K
Llama7B-MS-16K-32K Position Random	1.0	-/1.0	-/1.0	-	-	-	-
Llama7B-MS-16K-64K Position Random	1.0	-/1.0	-/1.0	-/1.0	-/1.0	-/1.0	-/0.6

Table 5: Performance of Llama7B-MS with NTK Position Encoding. Before 80K, set NTK scaling-factor to 4, 80K-160K set scaling-factor to 2, after 160K, set NTK parameter to a fixed size of 256, no longer grows with the length.

Model	2K-80K	100K	128K	160K	200K	220K	240K	256K
Llama7B-MS-16K-NTK	1.0	1.0	1.0	0.8/1.0	-/1.0	-/1.0	-/0.8	-/0.7

5.2.4 LongBench

Table 6 shows Few-shot Learning and Code Completion Evaluation on LongBench: These tasks do not require chat instruct fine-tuning. We have observed significant improvements in "trec", "triviaqa", "lcc", and "repobench-p" using our method on Llama2-7B-4k comparing with Llama2-7B-chat-4k, especially outperforming other models in the LCC task.

5.2.5 Recursive extension

To further evaluate whether our model can extend to longer sequences, we introduced the recursive method described in Section 4.3. By applying this method, our Multi-Scale MS Attention mechanism can maintain stable perplexity even for sequences up to and beyond 2 million tokens, as show in figure 3. Specifically, we trained the model on sequences of 16K length and applied the recursive method during evaluation, enabling effective inference on much longer sequences. However, this approach encounters certain issues in the passkey task. To further address the issue, it may be essential to employ our recurrent method to train on the passkey task, akin to the strategy utilized in infini-Attention[14].

5.2.6 Controlled degree of fit and convergence

The proposed method allows for controllable convergence and fitting degrees by flexibly setting parameters such as the attention range, the number of selected tokens, and the types of multi-scale compressions, as illustrated in Figure 2. This adaptability ensures that the model can be fine-tuned to achieve optimal performance across a variety of scenarios, providing robustness and efficiency in handling extended sequences.

5.2.7 ablation experiment

Firstly, in our pursuit of enhancing efficiency in micro-adjustments, we conducted fine-tuning on W_q, W_k, W_v, and W_o. The resulting Perplexity (PPL) is nearly identical to when fine-tuning is solely applied to W_k and W_o. In some instances, the latter even yields slightly higher results, as show in Table 7. This observation suggests that focusing on refining W_k and W_o alone can be as effective as fine-tuning all parameters mentioned earlier.

Table 6: Few-shot Learning and Code Completion Evaluation on LongBench

Model	TREC	TriviaQA	SAMSum	LCC	RepoBench-P
GPT-3.5-Turbo-16k	68	**91.4**	**41.7**	54.7	53.6
LongChat-v1.5-7B-32k	63.5	82.3	34.2	53	**55.3**
XGen-7B-8k	65.5	77.8	25.3	38.6	38.6
InternLM-7B-8k	52	77.8	21.2	44.1	28.8
ChatGLM2-6B-32k	62.5	78.7	36.3	55.6	49.9
Vicuna-v1.5-7B-16k	71.5	86.2	40.8	51	43.5
ChatGLM3-6B-32k	**79**	87.1	38.2	57.66	54.76
Llama2-7B-chat-4k	61.5	77.8	40.7	52.4	43.8
Llama-7B-32k-MS-PI	72.6	85.7	40.6	**61.95**	49.09

Table 7: Ablation experiment for PEFT. PEFT steps: 1000, Evaluation DataSet: PG19 validation.

PEFT	Training Context Length w/wo setting	Evaluation Context Length		
		32768	16384	8192
qkvo	16384-m16-s64	8.96	8.33	8.32
ko	16384-m16-s64	8.55	8.36	8.43

Table 8: PPL on the PG19 with Lora or not.

Model Configuration	2K	4K	8K	16K	32K
Llama-7B-MS-no_LoRA	7.06	6.96	6.78	6.56	6.93
Llama-7B-MS-LoRA	7.61	7.42	7.17	7.03	7.06

Table 9: Memory and Time w/wo DeepSpeed and LoRA

Configuration	Memory (MB) | Time (s/it)
With DeepSpeed & LoRA	36,786 | 6.37
Without DeepSpeed	31,524 | 6.19
Without LoRA	35,074 | 7.28
Without DeepSpeed & LoRA	40,132 | 6.38

6 Conclusion

In this paper, we build upon the LongLoRA framework, utilizing a mechanism of selection and merging within the Attention mechanism (referred to as MS Attention). By employing our approach on 2 $\times$ A100 40GB GPUs, we achieve the same level of length extension for Llama2-7B as LongLoRA does on 8 $\times$ A100 80GB GPUs. Specifically, we extend the context length of Llama2-7B to 100K tokens, significantly reducing the resource requirements for handling long sequences. Moreover, through the introduction of recursion, our method maintains stable perplexity even with sequences up to 2M tokens in length. Finally, the flexibility of our MS Attention mechanism allows for adjustable selection size and selection quantity, which, combined with restricting the attention range of each token, enables adaptable fitting and convergence rates.

References

[1] Salima Behdenna, Fatiha Barigou, and Ghalem Belalem. Document level sentiment analysis: a survey. *EAI endorsed transactions on context-aware systems and applications*, 4(13):e2–e2, 2018.

[2] Huan Yee Koh, Jiaxin Ju, Ming Liu, and Shirui Pan. An empirical survey on long document summarization: Datasets, models, and metrics. *ACM Comput. Surv.*, 55(8), dec 2022.

[3] Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. Code llama: Open foundation models for code, 2024.

[4] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

[5] Hongye Jin, Xiaotian Han, Jingfeng Yang, Zhimeng Jiang, Chia-Yuan Chang, and Xia Hu. Growlength: Accelerating llms pretraining by progressively growing training length, 2023.

[6] Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. Longlora: Efficient fine-tuning of long-context large language models, 2024.

[7] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.

[8] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

[9] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 3505–3506, New York, NY, USA, 2020. Association for Computing Machinery.

[10] Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. Extending context window of large language models via positional interpolation, 2023.

[11] Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. Longbench: A bilingual, multitask benchmark for long context understanding, 2024.

[12] Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, Chloe Hillier, and Timothy P Lillicrap. Compressive transformers for long-range sequence modelling. *arXiv preprint*, 2019.

[13] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context, 2019.

[14] Tsendsuren Munkhdalai, Manaal Faruqui, and Siddharth Gopal. Leave no context behind: Efficient infinite context transformers with infini-attention, 2024.

[15] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 17283–17297. Curran Associates, Inc., 2020.

[16] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers, 2022.

[17] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer, 2020.

[18] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection, 2021.

[19] Lei Zhu, Xinjiang Wang, Zhanghan Ke, Wayne Zhang, and Rynson W.H. Lau. Biformer: Vision transformer with bi-level routing attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10323–10333, June 2023.

[20] 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.

[21] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness, 2022.

[22] Ntk-aware scaled rope, 2023.

[23] Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. Yarn: Efficient context window extension of large language models, 2023.

[24] Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. Longrope: Extending llm context window beyond 2 million tokens, 2024.

[25] Shengnan An, Yifei Li, Zeqi Lin, Qian Liu, Bei Chen, Qiang Fu, Weizhu Chen, Nanning Zheng, and Jian-Guang Lou. Input-tuning: Adapting unfamiliar inputs to frozen pretrained models, 2022.

[26] Jimmy T. H. Smith, Andrew Warrington, and Scott W. Linderman. Simplified state space layers for sequence modeling, 2023.

[27] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces, 2022.

[28] 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, 2023.

[29] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2023.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[31] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[32] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2018.

[33] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019.

[34] Together Computer. Redpajama: an open dataset for training large language models, 2023.

[35] Amirkeivan Mohtashami and Martin Jaggi. Landmark attention: Random-access infinite context length for transformers, 2023.

A Appendix / supplemental material

Analysis of Attention with Correlation-Aware Selection and Merging is presented below:

A.1 Interpreting Attention from the Perspective of Interpolation

In this section, we reinterpret the theoretical formulation of the Attention mechanism by recasting it as a special interpolation formula. This perspective elucidates the role of different components within Attention and demonstrates its clustering effect. The detailed analysis is as follows:

A.1.1 Bilinear and Cubic Spline Interpolation

Consider the following interpolation formula:

$$f(x,y) = \sum_{i=0}^{h}\sum_{j=0}^{w} W_{ij} \cdot f(x+i, y+j)$$

$$o(x,y) = \sum_{i=0}^{h}\sum_{j=0}^{w} g(d_{ij}) \cdot v(x+i, y+j) \quad \textbf{or} \quad \sum_{i=0}^{h}\sum_{j=0}^{w} g(d_{ij}) \cdot v(x_i, y_j)$$

In the above equations, $h = 2$ corresponds to bilinear interpolation, while $h = w = 3$ corresponds to cubic spline interpolation. Here, (x, y) represents the target interpolation location, and d_{ij} denotes the distance metric between (x_i, y_j) and (x, y). The distance metric d_{ij} can be computed using various methods, such as the n-norm or inner product. The function $g(d_{ij})$ is a weighting function based on this distance metric. For bilinear interpolation, $g(d_{ij}) = \frac{x_i - x}{x_i - x_j}$, while for cubic spline interpolation, $g(d_{ij}) = \frac{x_i - x}{x_i - x_j}$.

By comparing these interpolation formulas with the Attention mechanism, we can approximate the global interpolation formula as follows: Let $g(d_{ij}) = \textbf{softmax}(q_i k_j^T)$ and $v(x_i, y_j) = v$. This demonstrates that the fundamental principles underlying both interpolation and Attention are similar. Both methods combine vectors with similar features, leading to interpolated results that emphasize these shared features, making them more useful for the task at hand. Although Attention does not strictly satisfy the conditions for interpolation—since it does not require the value at position (x_i, y_j) to be $v(x_i, y_j)$—it can be considered a generalized form of interpolation or regression. This analogy may explain why training in language models is often referred to as autoregression.

A.1.2 Interpolation Formula on Riemannian Manifolds

We can further extend this analogy by considering Attention as a special form of interpolation mapping or exponential mapping on a manifold. For instance, on a Riemannian manifold, there exists a matrix exponential mapping given by:

$$\textbf{Exp}_{(P,\lambda)}(T) = P^{1/2} \exp\left(\lambda P^{-1/2} T P^{-1/2}\right) P^{1/2}$$

where T is a tangent vector matrix, typically a symmetric matrix that can be decomposed into ww^T, and P is the metric matrix corresponding to a point on the manifold.

By generalizing the variables in the Attention formula, we can map ww^T to $w_Q w_K^T$ and the metric matrices $P^{1/2}$ and $P^{-1/2}$ to the X matrices in Attention. Thus, the interpolation formula becomes:

$$\textbf{Exp}_{(P,\lambda)}(T) = \exp\left(X w_Q w_K^T X\right) X^{-1}$$

This transformation completes the interpolation and maps it to the V space.

In summary, this theoretical perspective allows us to view the Attention mechanism as a form of interpolation, enabling us to leverage many well-established principles and techniques from interpolation to improve Attention. For example, giving higher weights to points closer in distance is a well-known and effective method in interpolation. This approach is employed in several successful algorithms, such as the Routing Transformer, Landmark Attention, and Biformer. Our MS Attention mechanism, under certain fixed parameter settings, can be nearly equivalent to these methods, fully covering their application scenarios. It can also reduce to other efficient Attention mechanisms, such as Swin Transformer. Detailed analysis of these equivalences is provided in the next section.

A.2 Approximation and Convergence of Our MS Attention to Various Efficient Attention Methods

Our proposed Select-Merge Attention (MS Attention) can approximate and converge to the optimal solutions of many efficient Attention mechanisms that utilize a subset of key-value (KV) pairs. Below, we describe the computation process using the notations introduced in the algorithm description:

- Selection:

$$A_s = Q'_s K_s'^T, \quad \textbf{Idx} = \textbf{topkIndex}(A_s)$$

- Merge:

$$Q_s = \textbf{merge}_q(Q'_s), \quad \textbf{Idx} = \textbf{filter}(\textbf{merge}_q(\textbf{Idx})), \quad KV_s = \textbf{Select}(KV, \textbf{Idx})$$

- Final Attention:

$$O = \textbf{Attention}(Q_s, K_s, V_s)$$

A.2.1 Landmark Attention

Landmark Attention adjusts the attention scores by incorporating landmark tokens, with the output expressed as:

$$O = (\textbf{softmax}(QK^T) \cdot \textbf{repeat}(\textbf{softmax}(QG^T), \textbf{blocksize}, \textbf{dim} = -1))V$$

When setting the split size of Q to 1 and the split size of K to blocksize, our MS Attention also calculates $\textbf{softmax}(QK_s^T)$ and $\textbf{softmax}(QK_s'^T)$, which fully enables the adjustment of attention scores using semantic tokens. This operation can be approximated under other settings as well.

Alternatively, if the score adjustment is performed without directly multiplying $\textbf{softmax}(QK_s'^T)$, treating semantic tokens as regular KV tokens during Attention computation, the output can be expressed as:

$$\frac{\textbf{sumexp}(QK_s^T) \times \textbf{softmax}(QK_s^T)V_s}{\textbf{sumexp}(QK_s^T) + \textbf{sumexp}(QK_s'^T)} + \frac{\textbf{sumexp}(QK_s'^T) \times \textbf{softmax}(QK_s'^T)V'_s}{\textbf{sumexp}(QK_s^T) + \textbf{sumexp}(QK_s'^T)}$$

where V'_s is a linear combination of V_s or V. This approach can also directly adjust the attention coefficient of V_s in the output, allowing convergence to the optimal solution based on the task.

A.2.2 BiFormer

The BiFormer computation process is as follows:

$$A_r = Q_r(K_r)^T, \quad I_r = \textbf{topkIndex}(A_r), \quad KV_g = \textbf{Select}(KV, I_r)$$
$$O = \textbf{Attention}(Q, K_g, V_g)$$

When the Merge size is set to 1, our method fully degenerates to BiFormer. To achieve lower complexity, BiFormer requires setting the initial region Q_r, K_r relatively large and then compressing it as a region representative. This approach reduces the accuracy of selection.

When merge size is greater than 1, our method performs more fine-grained partitioning and selection. If BiFormer's selection is optimal, our algorithm can also converge to this optimal selection. Overall, compared to BiFormer, our method has a larger convergence space, and under the same fine partitioning and selection, our method has relatively lower computational complexity.

A.2.3 Routing Attention Methods

For the Routing Transformer, the following update rule is applied:

$$\mu \leftarrow \lambda\mu + \frac{(1-\lambda)}{2} \sum_{i:\mu(Q_i)=\mu} Q_i + \frac{(1-\lambda)}{2} \sum_{j:\mu(K_j)=\mu} K_j$$

This can be rewritten as:

$$\mu \leftarrow \lambda\mu + \frac{(1-\lambda)}{2}\textbf{argmax}_{qu}(\mu Q^T)Q + \frac{(1-\lambda)}{2}\textbf{argmax}_{qu}(\mu K^T)K$$

where

$$\textbf{argmax}_{qu}(\cdot) = \begin{cases} 1 & \textbf{argmax}(\cdot, \textbf{dim} = -2) \\ 0 & \textbf{otherwise} \end{cases}$$

$$\textbf{Idx}_Q = \textbf{topkIndex}(\mu Q^T), \quad \textbf{Idx}_K = \textbf{topkIndex}(\mu K^T)$$

Using the triangle inequality of the metric, the above approximation can select Q-related KV tokens with:

$$\textbf{Idx} = \textbf{topkIndex}(Q\mu^T\mu K^T)$$

Our semantic tokens in each Attention step also cluster similarly:

$$m_s = \textbf{Softmax}(m_s K_s^T)K_s$$

or

$$m_s = m_s + \textbf{softmax}(Q_s K_s^T)V_s$$

Our selection process can be written as:

$$\textbf{Idx} = \textbf{topkIndex}(m_s W_Q W_K^T m_s^T)$$

where m_s is the regional semantic token of X_s (e.g., average or $m_s = \textbf{Softmax}(m_s K_s^T)K_s$), equivalent to further clustering.

Therefore, during selection, the cluster center can be used to represent the tokens within the relevant cluster, similar to the Routing Transformer. Our method, by only using the cluster center for selection, introduces minimal quantization error. However, because the selection quantity is sufficient, the loss is negligible. In contrast, the Routing Transformer loses some related Q and K tokens to ensure regular shape, introducing non-negligible error.

A.2.4 Swin Transformer

The Swin Transformer utilizes local Attention and shifted local Attention. Our selection mechanism can completely converge to Swin Transformer when it is optimal.

- Local Attention:

$$\textbf{softmax}(Q_{i:j} K_{i:j}^T)V_{i:j}$$

When this is the optimal solution, our selection mechanism will automatically select tokens within the local region.

- Shifted Local Attention:

$$\textbf{softmax}(Q_{i:j} K_{i+r:j+r}^T)V_{i+r:j+r}$$

When this is optimal, our selection mechanism will automatically select tokens within the corresponding region ($KV_{i+r:j+r}$), where r is the cyclic shift offset.

In a similar manner, our method can approximate or cover many variants of the above Transformers.

To compare our method with these efficient Transformers, we conducted experiments on the PG19 and ImageNet datasets, as shown in the tables 10 and 11.

On the PG19 dataset, our method outperforms Landmark Attention in terms of both performance and efficiency, improving PPL by 3.6. The efficiency advantage becomes more pronounced as the input length increases.

Table 10: Results on PG19: Perplexity (PPL) and Training Memory Consumption (MB). 20186/22496 is using flashatten or not

Model	PPL (PG19)	1 × 8192	1 × 4096	1 × 2048
MS Attention-110M	15.9	-	-	-
MS Attention-300M	10.9	20186/22496	13152/14300	9976/10346
Landmark Attention-300M	14.55	>40960 (OOM)	>40960 (OOM)	17938

Table 11: Results on ImageNet: Top-1 Accuracy, FLOPs, and Parameters

Model	FLOPs (G)	Params (M)	Top-1 Acc. (%)
Swin-T	4.5	29	81.3
BiFormer-T	2.2	13.1	81.4
MS Attention (Ours)	2.0	13.1	**82.0**

A.3 Detailed Parameter Settings

Our algorithm can encompass the majority of Q-attention to KV-subset methods by adjusting parameters such as the QKV segment size, the number of selected top-K high-relevance KV regions, and the number of merged Q regions. Below is a detailed discussion on the selection of these hyperparameters:

A.3.1 QKV Segment Size Selection:

The QKV segment size is crucial and typically ranges from 8 to 128. This parameter must be chosen by balancing computational complexity and performance. In future work, we plan to incorporate Triton operators in the QK routing step to achieve linear spatial complexity, thereby mitigating the current limitations:

- (1) During the selection and routing process, a semantic token represents each region, and the relevance between Q and K semantic tokens is measured. KV tokens related to the Q region are then selected based on this relevance. A larger QK region size results in a coarser semantic representation, leading to less precise KV token selection. To minimize this loss, more KV tokens must be selected in the second step, thereby reducing information loss due to coarse semantics. Thus, smaller segment sizes are preferred, though incorporating Triton operators in the QK routing step is anticipated to provide linear spatial complexity.

- (2) Alternatively, a larger segment size can be set, and more KV tokens can be selected in the second step. This approach may increase algorithmic complexity as more KV tokens are likely to be used for interpolation within the same Q region. However, this issue can be alleviated through the third step of merging, where the selected KV tokens can be further merged and selected according to relevance, filtering out irrelevant information.

- (3) Additionally, the segment size of Q regions is independent of the segment size of KV regions. A relationship between them should only be established when specific tasks and complexity constraints require it.

A.3.2 Top-K High-Relevance KV Region Selection:

The number of selected top-K high-relevance KV regions is generally determined by factors such as the model's pre-training length and the amount of task-specific data. It may also need to be adjusted based on the granularity of the segmentation from the first step. Current large model training methods suggest that selecting 1K-8K KV tokens for interpolation is robust:

- (1) The above represents a general approach, while scenario-specific selection yields higher performance and efficiency. For example, if the task's data volume is small, a relatively small number of high-relevance regions can be selected to maximize overfitting and memorize the critical data. Conversely, as the data volume increases, more KV tokens may be required for

autoregressive prediction of the next token, necessitating the selection of more high-relevance regions.

- (2) In addition to task-based selection, the granularity of segmentation from the first step must also be considered. If segmentation precision is insufficient, more KV tokens should be selected to prevent the loss of critical information. Subsequent merging can then be used to jointly select high-relevance information, improving space and time complexity by sharing KV tokens across Q regions.

- (3) For length fine-tuning, the model's parameters have adapted to the interpolation degree of the pre-training length, which uses a specific number of KV tokens. Hence, the same magnitude of KV tokens must be selected during fine-tuning to avoid overfitting.

This parameter is highly flexible and can be adapted to various scenarios, often requiring some experience for optimal settings. Future work may involve incorporating a loss function to control the selection space, allowing the algorithm to automatically select the appropriate size based on the loss.

A.3.3 Merging Q Regions:

The number of merged Q regions generally depends on the segmentation precision and space complexity. In scenarios where space complexity is not a major concern, sizes ranging from $\frac{topk}{4C_Q}$ to $\frac{topk}{C_Q}$ can be chosen. The number of merged KV tokens is typically selected from a range of topk to 2topk KV tokens. Generally, sharing high-relevance KV regions across multiple Q regions results in lower space complexity. Moreover, during the merging process, high-relevance KV regions are selected with flexibility, enhancing performance through methods such as:

- a. Simple unique and sort operations for selection.

- b. Unique operations followed by a secondary segmentation based on high-relevance scores, with allocation and selection according to region clustering.

A.3.4 Algorithm Complexity Analysis

(1) Let the segmentation sizes for the Q and KV regions be C_Q and C_K, respectively. The time and space complexity of routing Q and K using dot products is $O\left(\frac{N^2}{C_Q C_K}\right)$.

(2) In the second step, the selection of high-relevance regions, denoted by C_S, results in space complexity of $O\left(\frac{N}{C_Q}C_S\right)$ due to the storage of necessary indices.

(3) The merging step, with a merge size of C_M, involves combining the selected indices. Filtering algorithms can be introduced during merging; in this work, we employ unique and high-score selection. The space complexity for storing KV tokens after selection and merging is $O\left(\frac{N^2}{C_Q C_M}C_S\right)$.

(4) The final attention operation has a time complexity of $O(2\frac{N}{C_Q C_M}C_Q C_M C_S) = O(NC_S)$. Hence, only the segmentation size and selection quantity affect the algorithm's complexity.

Without merging, the selected KV tokens have a complexity of $O\left(\frac{N^2}{C_Q}C_S\right)$, whereas with merging, the complexity is $O\left(\frac{N^2}{C_Q C_M}C_S\right)$. The reduction in complexity due to merging is because the selection process is only carried out after merging, storing only the selected indices before that.

A.4 Positional Awareness and Breaking Translation Invariance

A.4.1 Positional Awareness:

Our algorithm leverages semantic token-based routing and selection mechanisms to capture boundary effects, thus disrupting the translation invariance inherent in global Attention mechanisms.

Semantic Token Routing and Selection: The routing process in our algorithm employs semantic tokens, which represent the semantics of a region, carrying contextual information similar to a sliding window in convolutional neural networks (CNNs). Due to the contextual variance surrounding each token, the derived semantic tokens differ, leading to the routing of different KV tokens. This results in the selection of distinct interpolation variables and, consequently, varying outputs. This mechanism allows our approach to effectively capture boundary effects, unlike traditional global Attention which maintains translation invariance.

A.4.2 Extrapolation through Finetuning and Position Interpolation:

Our method can be fine-tuned during training and later applied to full Attention during inference, enabling significantly higher extrapolation factors. Specifically, our MS Attention mechanism can achieve up to twice the extrapolation compared to traditional methods, with potential reasons outlined as follows:

- Improved Relative Positional Awareness: The primary reason behind this extrapolation capability is our selection mechanism, which better extends the model's awareness of relative positions. In full Attention, tokens near the boundaries rely on all preceding tokens for regression, potentially leading the model to integrate all previous positional information. In contrast, our selection mechanism integrates a subset of positions into the final positional information, leading to a more accurate representation of relative positions. This method allows the model to assert the correct positional information even with different combinations of preceding positions.

- Extrapolation Limits: The reason for successful extrapolation up to twice the training length, but not beyond, lies in the finetuning process itself. During finetuning with a length l, the model learns the correct positional information and its integration within this range. Therefore, the model can correctly extrapolate up to $2l$, but further extrapolation may fail as it exceeds the recognized range.

For instance, when finetuning LLaMA2-7B with a length of $l = 16K$, the positional range for 'Position_ids' is set from 1 to 16K. By increasing the initial position range, e.g., choosing starting positions within the 1-64K range ($l_1 = 64K$), and setting the interpolation ratio for position interpolation as $80K/4K = 20$ or higher, we successfully achieve 100% accuracy on the 80K-length passkey task. Similarly, by exposing the model to even longer positions and using our adaptive positional interpolation encoding, we set $l_1 = 128K$ and successfully extrapolate to 144K. This analysis suggests that our method, even with simple position interpolation, has the potential to achieve near-infinite length extrapolation.

A.4.3 Extrapolation through Finetuning and NTK-Based Positional Encoding:

Combining our MS Attention with NTK-based positional encoding enables extrapolation by factors exceeding tenfold. This approach is explained using the radix theory proposed by the authors of RoPE:

- Learning Relative Magnitudes: Our MS Attention accurately learns the relative magnitudes of positional encodings. By incrementally increasing the radix base size, the model learns to represent the relative magnitudes across different radices. Additionally, by varying the starting positions, the model further refines its understanding of relative magnitudes within the same radix.

- Future Work: Future extensions will incorporate relative shifts between Q tokens and K tokens, allowing the model to sense deviations between different positions of Q and K.

This combination of MS Attention and NTK-based encoding showcases a significant potential for enhancing extrapolation capabilities, ultimately pushing the boundaries of positional understanding in large language models.

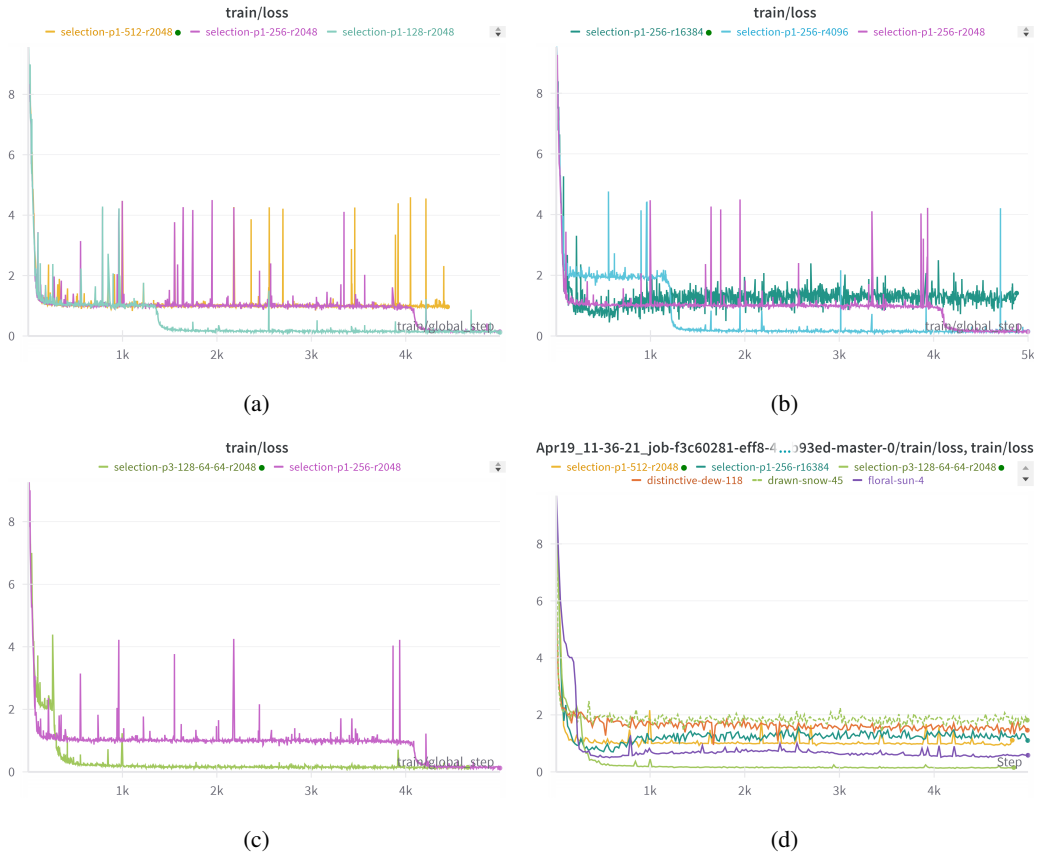

Figure 2: **Loss Curves for Llama2-7B Using Restricted Scope Multi-Scale Selected Attention.** The model gradually increases the receptive field through the outputs of each layer. p: number of compression scales, e.g., p1 for one scale, p3 for three scales. r: maximum scope each token attends to, e.g., r2048 indicates attending to the first 2048 tokens. (a) Fixed range of 2048, comparing loss curves for selecting 128, 256, and 512 tokens. Fewer selected tokens lead to overfitting. (b) Fixed selection number of 256, varying the attention range. Smaller attention ranges lead to overfitting. (c) Multi-scale selection with three compression scales, which converges more effectively and mitigates overfitting. (d) Flexible parameter settings to achieve various convergence and fitting degrees, demonstrating a loss range of 0.1 to 2.

A.5 Controlled degree of fit and convergence

Table 12: Lossfor Llama2-7B Using Restricted Scope Multi-Scale Selected Attention.

Parameters	loss
selection-p1-512-r2048	1.28
selection-p1-256-r16384	1.44
selection-p3-128-64-64-r2048	0.31
selection-p1-2048-rno	1.73
selection-p1-2048-Recurrent	1.97
selection-p3-128-64-64	0.76

A.6 The result of the recursive extension

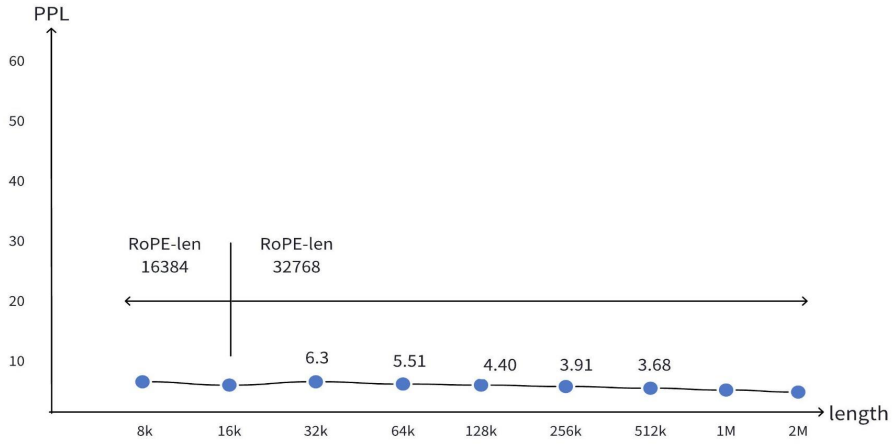

Figure 3: **Sequence Length Extension Using Recursive Methods.** By combining multi-scale MS Attention with recursive methods, we extend the sequence length. The model is fine-tuned on 16K length sequences, and during evaluation, lengths less than 16K are interpolated to 16K positions, while lengths greater than 16K use 32K position interpolation.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: The papers not including the checklist will be desk rejected. The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- **You should answer [Yes] , [No] , or [NA] .**
- **[NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.**
- **Please provide a short (1–2 sentence) justification right after your answer (even for NA).**

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS paper checklist",**
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers.**

1. **Claims**

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

   Answer: [Yes]

   Justification: In the abstract, we summarized the alogrithm we proposed, the Attention with Selection and Merging, and the results obtained from our experiments.

   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 reveal the recursive extension encounters certain issues in the passkey task. And we give a way to further address the issue that akin to the strategy utilized in infini-Attention.

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: [Yes]

   Justification: We give specific computational procedures and parameter settings to cover the Biformer, routing Transformer, Landmark Attention, and Swin Transformer methods in the Appendix.

   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 show the parameter settings for the experiment in the experiment table or in the header, including the split size, the number of merges and the number of selections. Regarding the positional coding approach, we point out how to set the scaling to get the desired result

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: [NA]

Justification: We didn't commit the code, but we can always do so, regardless of the outcome, and we'll open source the code on github soon after the results are in!

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 show the parameter settings for the experiment in the experiment table, in the header or the section of the Experiment.

   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: We did not conduct experiments that required reporting 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: In section 5.1, we provided details about the computer resources used for the 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: he research conducted in the paper complies with the NeurIPS ethical guidelines in all aspects.

   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: [No]

    Justification: We're not having a relevant discussion.

    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: [No]

    Justification: Our current work extends the length of the pre-existing model, while our pre-trained model is trained only on the pre-existing dataset

    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: he data, code, and models used in this paper have all been properly cited.

    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: We provide the theory and details of the algorithm in section 4 and appendix.

    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: the 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: the 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.**