DCIS: Efficient Length Extrapolation of LLMs via Divide-and-Conquer Scaling Factor Search

Lei Yang[1], **Shaoyang Xu**[2], **Jianxiang Peng**[1], **Shaolin Zhu**[1], **Deyi Xiong**[1,2*]
[1]TJUNLP Lab, College of Intelligence and Computing, Tianjin University, Tianjin, China
[2]School of New Media and Communication, Tianjin University, Tianjin, China
{yanglei_9, zhushaolin, dyxiong}@tju.edu.cn

Abstract

Large language models (LLMs) based on the Transformer architecture usually have their context length limited due to the high training cost. Recent advancements extend the context window by adjusting the scaling factors of RoPE and fine-tuning. However, suboptimal initialization of these factors results in increased fine-tuning costs and reduced performance at target length. To address these challenges, we propose a novel RoPE-based fine-tuning framework that diverges from conventional scaling factors search. Specifically, we present a **D**ivide-and-**C**onquer **I**ncremental **S**earch (DCIS) algorithm that strategically determines the better scaling factors. Further fine-tuning with the identified scaling factors effectively extends the context window of LLMs. Empirical results demonstrate that our methodology not only mitigates performance decay at extended target lengths but also allows the model to fine-tune on short contexts and generalize to long contexts, thereby reducing the cost of fine-tuning. The scaling factors obtained through DCIS can even perform effectively without fine-tuning. Further analysis of the search space reveals that DCIS achieves twice the search efficiency compared to other methods. We also examine the impact of the non-strictly increasing scaling factors utilized in DCIS and evaluate the general capabilities of LLMs across various context lengths.

1 Introduction

Transformer (Vaswani et al., 2017) has emerged as the preferred architecture for large language models due to its scaling capability (Brown et al., 2020; Achiam et al., 2023). However, the inherent quadratic complexity of self-attention necessitates limiting the context window during pre-training, exemplified by the 4096-token limit in Llama2 (Touvron et al., 2023). When models encounter sequences beyond this limit during inference, a significant loss in performance occurs (Press et al., 2022; Yang et al., 2025; Guo et al., 2023). To extend the operational context window of LLMs, previous studies have explored a wide variety of methods, such as sequence truncation (Rae et al., 2020a; Dai et al., 2019; Wu et al., 2022) and sparse sequencing (Han et al., 2023; Ding et al., 2023; Xiao et al., 2024b), though these methods often result in the loss of critical contextual information.

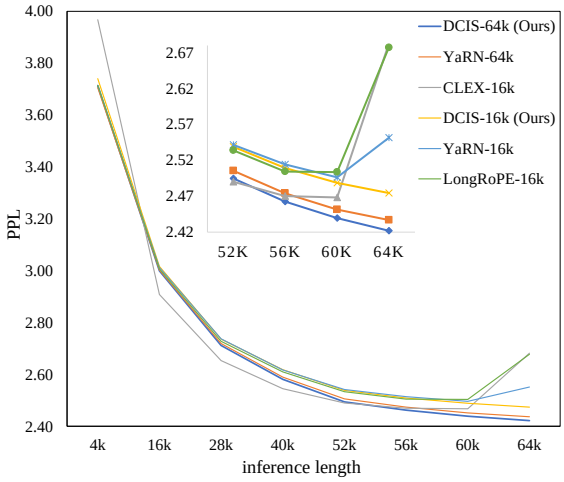


Figure 1: The Llama2-7B model is expanded to 64k context window. We test the PPL using 10 Proof-pile samples with a minimum length of 128k tokens. Fine-tuning is performed using the default method described in Section 5. "-64k/16k" indicates that fine-tuning on a 64k/16k length generalizes to a target length of 64k.

Recent advances in positional encoding techniques have facilitated length generalization capabilities in LLMs, evolving from the initial sinusoidal positional encodings of Transformers to learnable and relative positional encodings (Gehring et al., 2017; Shaw et al., 2018; Sun et al., 2024). The introduction of Rotary Position Embedding (RoPE) (Su et al., 2024) has catalyzed a new

*Corresponding author

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 15168–15182
November 4-9, 2025

wave of research that aims to increase the extrapolation length of LLMs by modifying the rotation frequency of the embedding dimensions through scaling factors (Peng et al., 2024), complemented by simultaneous fine-tuning to sustain long-context performance.

However, due to the quadratic complexity of self-attention and the increased number of intermediate activations cached during fine-tuning for long sequences (Shen et al., 2023; Pan et al., 2025), direct fine-tuning on target lengths (Peng et al., 2024) leads to significantly high memory consumption and long time as the target sequence length increases. While fine-tuning on short contexts to generalize to target lengths (Chen et al., 2024) is more efficient, traditional approaches (Chen et al., 2023) often suffer from a significant performance drop (Chen et al., 2024; Ding et al., 2024) at the target length due to their limited generalization capability. As shown in Figures 1 and 3, these existing methods fail to achieve satisfactory results in perplexity (PPL) and passkey performance metrics at target lengths.

To address these challenges, we explore better scaling factors to unlock the potential of LLMs for length generalization. We propose a novel RoPE-based fine-tuning framework that departs from traditional approaches of linearly increasing scaling factors (Peng et al., 2024; Ding et al., 2024). Instead, our framework introduces a Divide-and-Conquer Incremental Search (DCIS) algorithm, leveraging a principle of refinement from broad to specific, to efficiently determine the better scaling factors through continuous target-length inference. The non-strictly increasing nature of DCIS expands the search space considerably. The scaling factors identified via DCIS are then utilized for fine-tuning, significantly extending the model's context window to the target length.

We conduct a comprehensive evaluation of DCIS on Llama2-7B, Llama3-8B and Mistral-7B-v0.1 using PPL and Passkey. Experimental results demonstrate that DCIS effectively mitigates the performance degradation of models at target context lengths.

Our contributions can be summarized as follows:

- We propose a novel framework involving a Divide-and-Conquer Incremental Search (DCIS) algorithm for scaling factor search, followed by fine-tuning to extend the context window of LLMs.
- Extensive experiments demonstrate that DCIS overcomes the challenge of performance degradation at target lengths and exhibits strong generalization ability, leading to further reductions in fine-tuning costs. In addition, DCIS leads to substantial performance improvements even without fine-tuning.
- We conduct an in-depth analysis of DCIS, encompassing the impact of scaling factor initialization, the search space of DCIS, the role of Adaptive Scaling Factors (ASF), the effect of DCIS on general ability, sensitivity analysis of introduced hyperparameters, and scalability validation of the overall framework, demonstrating the method's effectiveness, efficiency, and robustness.

2 Related Work

Positional Embedding Scaling. Recent advancements in positional embedding scaling, particularly involving Rotary Positional Embedding (RoPE), have significantly improved the capability of LLMs to manage extended context windows. Notable methods such as PI (Chen et al., 2023), CodeLlama (Rozière et al., 2023) and YaRN (Peng et al., 2024) manually adjust scaling factors, whereas CLEX (Chen et al., 2024) optimizes rotation frequencies through training. LongRoPE (Ding et al., 2024) employs a search mechanism to fine-tune scaling factors, enhancing the model's extrapolation abilities. While these approaches demonstrate improved performance through better scaling factors utilization, they are still hindered by high fine-tuning costs and notable performance declines at target lengths, issues that our proposed method addresses more efficiently.

Sequence Compression. Models leveraging RoPE have shown intrinsic capabilities for length extrapolation even without fine-tuning. Approaches like ReRoPE (Su, 2023) and Self-Extend (Jin et al., 2024) extend sequence lengths by compressing sequence indices, although they necessitate double attention computations, raising computational demands and limiting extrapolation potential. In contrast, our approach facilitates unrestricted extension of the model's context length through standard inference process, eliminating the need for repeated attention mechanisms.

Chunking/Sparse Attention. Investigative efforts have revealed that models predominantly focus on information at the sequence extremes

(Liu et al., 2024), suggesting that removing mid-sequence data while preserving initial and proximate tokens minimally impacts overall information integrity (Han et al., 2023; Xiao et al., 2024b). Alternative strategies involve segmenting sequences into chunks corresponding to pre-training lengths and employing external memory modules for storing and recalling past contexts during current chunk inference (Rae et al., 2020a; Dai et al., 2019; Wu et al., 2022). While these methods enable some degree of length extension, they inherently sacrifice a portion of the contextual data. Our method, however, maintains the integrity of the entire text, thereby maximizing the utility of the available context information.

3 Preliminary

This section elucidates the principles underpinning our approach and delineates the problem concerning positional embedding scaling, with a focus on enhancing the Rotary Positional Embedding (RoPE) (Su et al., 2024) technique widely adopted in LLMs.

3.1 Rotary Position Embedding

RoPE has garnered significant attention in the realm of LLMs due to its robust performance and superior extrapolation effectiveness. Consider a sequence of embedding vectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_L \in \mathbb{R}^d$, where L represents the length of the input sequence and d denotes the dimensionality of the hidden states for each head. Let m indicate the position index. RoPE incorporates positional information into the embedding vectors through a rotational transformation defined as follows:

$$f_{\{Q,K\}}(\mathbf{x}_m, m, \mathbf{\Theta}) = \mathbf{R}^d_{\mathbf{\Theta},m}\mathbf{W}_{\{Q,K\}}\mathbf{x}_m, \quad (1)$$

where $\mathbf{W}_Q$ and $\mathbf{W}_K$ denote the weights matrices of theTransformer. $\mathbf{R}^d_{\mathbf{\Theta},m}$ is the rotation matrix parameterized by $\mathbf{\Theta} = \{\theta_i = 10000^{-2(i-1)/d}, i \in [1, 2, \ldots, d/2]\}$:

$$\mathbf{R}^d_{\mathbf{\Theta},m} = \left(\begin{bmatrix} \cos m\theta_i & -\sin m\theta_i \\ \sin m\theta_i & \cos m\theta_i \end{bmatrix}\right), \quad \text{for } i = 1, 2, \ldots, \lfloor d/2 \rfloor. \quad (2)$$

This transformation ensures that the relative positional information $|m - n|$ is implicitly encoded in the attention scores:

$$\begin{aligned} \mathbf{Q}_m^T\mathbf{K}_n &= (\mathbf{R}^d_{\mathbf{\Theta},m}\mathbf{W}_Q\mathbf{x}_m)^T(\mathbf{R}^d_{\mathbf{\Theta},n}\mathbf{W}_K\mathbf{x}_n) \\ &= \mathbf{x}^T\mathbf{W}_Q\mathbf{R}^d_{\mathbf{\Theta},n-m}\mathbf{W}_K\mathbf{x}_n. \end{aligned} \quad (3)$$

3.2 Frequency Scaling

Frequency Scaling. Despite RoPE's effectiveness in incorporating positional information, the model's capacity to handle sequences exceeding pre-training lengths remains limited, primarily due to inadequate training of low-frequency dimensions (LocalLLaMA, 2023) within the conventional context window size L. To address this, several scaling methods have been proposed to adjust RoPE's rotation frequency. We denote $\mathbf{R}^d_{\mathbf{\Theta},m}$ succinctly as:

$$\left[\left(\cos\left(\frac{m}{\lambda_i\beta_i}\right), \sin\left(\frac{m}{\lambda_i\beta_i}\right)\right), i \in \left[0, \frac{d-2}{2}\right]\right], \quad (4)$$

where $\beta_i = 10000^{2i/d}$ represents the base frequency, and λ_i are the scaling factors for each frequency dimension.

Scaling Factors Methods. Both NTK-aware approach (LocalLLaMA, 2023) and YaRN (Peng et al., 2024) apply theories from the Neural Tangent Kernel (NTK) and suggest varying the scaling factors λ_i based on the dimension's training need, achieving better performance with less fine-tuning data. LongRoPE (Ding et al., 2024), on the other hand, utilizes a search-based method to identify better scaling factors λ_i, employing an evolutionary search algorithm for initial short text search (128k/256k) and fine-tuning, and search again at the target length (2M). This method facilitates up to a significant increase in processing length compared to the baseline, highlighting the potential of scaling adjustments in extending model capacities.

4 Methodology

Although existing methods demonstrate generalization capabilities, they require significant fine-tuning data and suffer from marked performance degradation at extended target lengths. To address these challenges, we propose Divide-and-Conquer Incremental Search (DCIS) algorithm for length exploration. Figure 2 illustrates our framework, beginning with an exploratory search phase using the Perplexity (PPL) metric as a guide, followed by fine-tuning using the identified scaling factors to effectively extend the model's contextual modeling. In this section, we introduce our framework (Section 4.1), followed by a detailed description of the DCIS algorithm (Section 4.2).

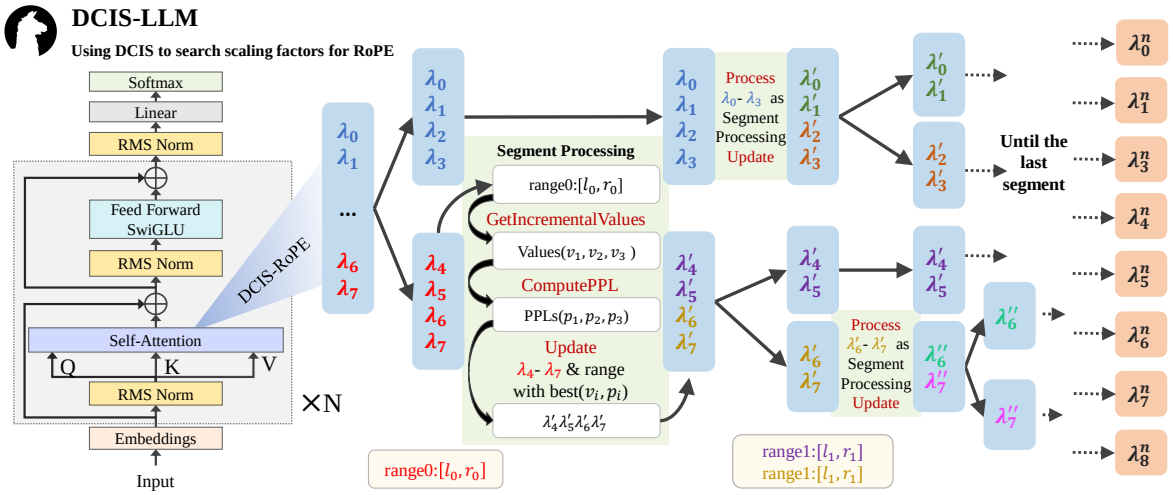


Figure 2: Diagram of the proposed DCIS framework. We illustrate the search procedure when $d = 16$, with 3 incremental values selected for each processing step. Since the scaling factors are divided into high-frequency and low-frequency parts, we will initially process them in two segments. First, DCIS searches the scaling factors (i.e., $\lambda_4 - \lambda_7$) for the last 4 positions and gets 3 incremental values (i.e., v_i) within the range $[l_0, r_0]$. It then computes the PPL (i.e., p_i) for each incremental value. Finally, it selects the best incremental value with the lowest PPL to update these 4 scaling factors. As the input sequence is divided into two segments, DCIS processes the first 4 scaling factors in the first segment in the same manner. At the second layer, DCIS processes 2 scaling factors at a time, and at the third layer, it processes 1 scaling factor at a time, so on so forth. Finally, the process ends with the obtained scaling factors from the search.

4.1 Framework

Searching Scaling Factors during Inference. Recent research (Wu et al., 2025) reveals that specific retrieval heads contain long-form textual information, suggesting the intrinsic capacity of LLMs to process extended texts. To exploit this potential within LLMs, our framework first involves searching for optimal scaling factors during the preliminary inference phase. More specifically, our approach encourages the model to autonomously select suitable scaling factors through low-cost inference, thereby enhancing its extrapolation capability with low-cost training.

Fine-Tuning with Searched Factors. Directly applying the searched scaling factors to the original LLMs leads to suboptimal performance, indicating a lack of sufficient adaptation within the LLMs (Ding et al., 2024). Following YaRN, our framework further involves a fine-tuning phase with searched scaling factors. Due to our advanced search algorithm (Section 4.2), better scaling factors are searched and initialized, thereby further reducing the number of fine-tuning steps.

4.2 Divide-and-Conquer Incremental Search

We elaborate our Divide-and-Conquer Incremental Search (DCIS) algorithm, which integrates the divide-and-conquer strategy to efficiently approximate better scaling factors. Beyond Figure 2, Algorithm 1 presents the detailed procedure of DCIS. We also demonstrate that the search space of DCIS is half that of conventional search methods, in Section 6.1.

Algorithmic Strategy. The scaling factors sequence, designated as $[\lambda_0, \lambda_1, \cdots, \lambda_{d/2-1}]$ (as formulated in Eq. 4), is methodically processed using a divide-and-conquer strategy. Specifically, we divide the sequence into a set of segments, and focus on one segment (as highlighted in red in Figure 2) at a time while maintaining the others constant. For each segment, we adopt an incremental search strategy, where a new sequence of scaling factors is generated by adding a predetermined value from a set range to the current sequence. The most effective increment is then selected based on the lowest perplexity (PPL), which is also used to narrow the range for subsequent searches in the divided subsegments. This iterative process, shown during the first iteration for $d = 16$ with three values evaluated at each segment in Figure 2, incrementally refines the scaling factors.

As shown in Algorithm 1, the scaling factors are divided into high-frequency and low-frequency components. Hence we initially process them in

two segments, with each segment handling $N = \text{head\_dim}/2$ scaling factors. In each iteration, the Segment function is first employed to obtain the current segment Seg that needs to be processed:

$$\begin{aligned} \text{Seg} &= [(\lambda_i,\ \lambda_{i+1}, \cdots,\ \lambda_{i+N-1}), \\ \text{where } i &= d - N \times j,\ j \in [1,\ d/N]]. \end{aligned} \tag{5}$$

Subsequently, the GetIncrementalValues function is used to uniformly extract C incremental values from the range of the current segment:

$$\begin{aligned} \text{Values}[k] &= [R_{j,l} + \text{step} \times k], \\ \text{where step} &= \frac{(R_{j,r} - R_{j,l})}{(C-1)},\ k \in [0,\ C-1]. \end{aligned} \tag{6}$$

The ComputePPL function is then utilized to add these incremental values to the current scaling factors, yielding new scaling factors and thereby calculating PPLs:

$$\text{PPLs}(p_k) = \text{ComputePPL}(\text{Seg} + v_k). \tag{7}$$

Finally, the PPLs are used to update the scaling factors and the range of values for the next iteration. Specifically, the incremental value with the lowest PPL is used to update the scaling factors, while the lowest $C/3$ PPL incremental values are set as the range for the next iteration:

$$\begin{aligned} \text{PPLs, Values} &= \text{sort}((p_k, v_k)) \\ \mathbf{F}_{\text{Seg}} &= \text{Seg} + v'_1, \\ R_{2\times j-1} &= R_{2\times j} = [l, r], \\ \text{where } l &= \min(v'_1, v'_2, \cdots, v'_{C/3}), \\ r &= \max(v'_1, v'_2, \cdots, v'_{C/3}). \end{aligned} \tag{8}$$

This process continues until the scaling factors of the last segment are returned as the result of this search.

Optimizations Incorporated. Based on empirical insights, several optimizations have been incorporated into the DCIS algorithm:

1. Initial Scaling Factors: Drawing from the success of YaRN (Peng et al., 2024), its scaling factors are used as starting points from which our search begins.

2. Priority to High-Dimensional Scaling Factors: Inspired by the NTK-aware approach (LocalLLaMA, 2023), which suggests that higher dimensions might require more extensive interpolation, our algorithm prioritizes these dimensions for updates, thereby speeding up the convergence to better scaling factors.

Algorithm 1 DCIS

Input: The target LLM, input samples $\mathbf{X}$, initial scaling factors $\mathbf{F}$, initial range R, the number of increments processed each time C, the number of dimensions for each head d.

```
N=d/2;
while N>=1 do
    for Seg=Segment(N) do
        Values=GetIncrementalValues(R);
        PPLs=ComputePPL(LLM, X, F, Seg, Values);
        F, R = update (F, R, PPLs);
    end for
    N=N/2;
end while
Return the searched scaling factors F;
```

3. Discarding Non-Guiding Increments: Increments resulting in a PPL greater than 100 are considered ineffective and are thus excluded from the search to maintain focus on potentially successful modifications.

4. Avoiding Local Optima: To prevent falling into local optima, when updating the range for the next layer of values, in addition to using the top $C/3$ PPL incremental values, we also expand the upper and lower bounds outward by one step. Here, one step is defined as the difference between adjacent incremental values.

Adaptive Scaling Factors (ASF). Unlike methods such as YaRN and LongRoPE, which prescribe a strictly increasing order for scaling factors as frequency decreases, our approach does not confine the model to predetermined scaling paths. Considering the complex and often opaque internal mechanisms of models, we posit that different dimensions within each head may require distinct treatments, some might even need interpolation despite being high-frequency components. Thus, our framework allows for flexible scaling factors adjustments, tailored to the specific needs of each dimension.

5 Experiments

We conducted extensive experiments to examine the performance of the proposed DCIS across various metrics and conditions.

5.1 Setup

Model and Evaluation Tasks. We carried out our experiments using the Llama2-7B (Touvron et al., 2023), Llama3-8B (Dubey et al., 2024) and Mistral-7B-v0.1 (Jiang et al., 2023), aiming to extend the model's context window to 64k tokens. We

Method	Fine-tuning		Inference Length								
	Length	Data	4k	16k	28k	40k	52k	56k	60k	64k	AVG
PI	32k	-	3.70	2.97	2.68	7.91	44.29	75.64	122.90	187.65 ↑	55.97
CodeLlama	16k	500B	3.95	3.11	2.79	2.67	2.61	2.58	2.56	2.55 ↓	2.85
YaRN	64k	0.8B	3.71	3.00	2.72	2.59	2.51	2.47	2.45	2.44 ↓	2.74
	16k	0.4B	3.71	3.01	2.74	2.62	2.54	2.51	2.50	2.55 ↑	2.77
CLEX	16k	2.5B	3.97	**2.91**	**2.65**	**2.55**	**2.49**	2.47	2.47	2.68 ↑	2.77
LongRoPE	16k	0.4B	3.72	3.01	2.73	2.61	2.53	2.50	2.50	2.68 ↑	2.78
DCIS (Ours)	64k	0.8B	3.71	3.00	2.71	2.58	2.49	**2.46**	**2.44**	**2.42** ↓	**2.73**
	16k	0.4B	3.74	3.02	2.74	2.62	2.54	2.51	2.49	2.47 ↓	2.77
	4k	0.8B	**3.68**	2.99	2.72	2.62	2.54	2.52	2.50	2.49 ↓	2.76

Table 1: PPL of different fine-tuning parameter configurations for various methods on Llama2-7B using the Proof-pile dataset. Blue arrows indicate a decrease in PPL at the target length, while red arrows indicate an increase.

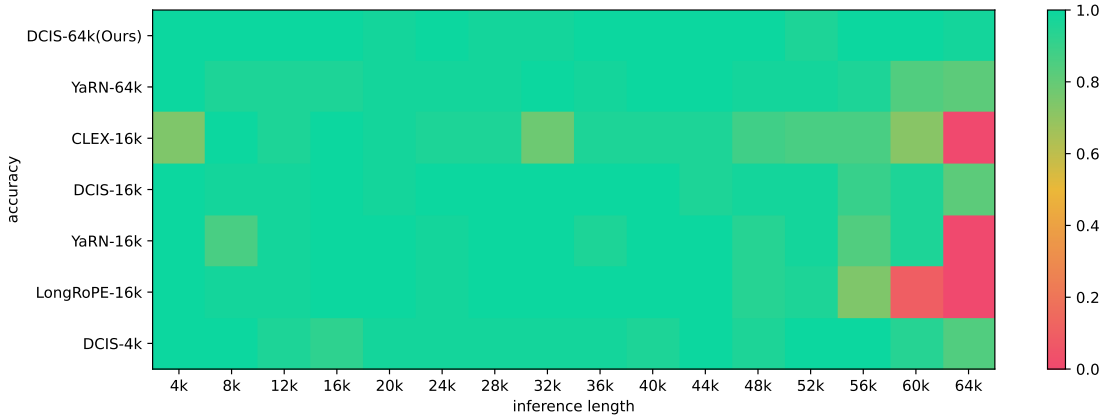


Figure 3: The recall rate of passkey with different lengths. Higher values indicate better performance.

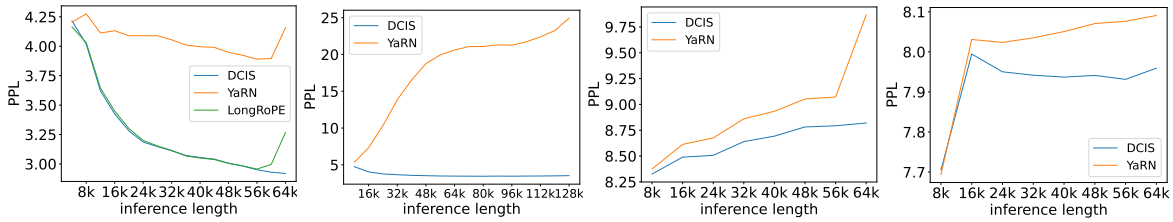
adopted YaRN's methodology for perplexity (PPL) evaluation, utilizing ten samples from the Proof-pile dataset (Rae et al., 2020a) with lengths of at least 128k tokens for assessment. Additionally, we assessed model performance using 50 passkey (Mohtashami and Jaggi, 2023) tests at each length.

Fine-tuning Parameters. We used Llama2-7B as the base model. Initially, our DCIS algorithm was employed to identify scaling factors at the target length of 64k, setting the initial range between $[-5, 5]$ with $C = 10$ incremental values per segment. Following this, we segmented the PG19 dataset (Gao et al., 2021) into 4k, 16k, and 64k context lengths, and then performed fine-tuning on each segment. The fine-tuning process closely mirrors YaRN's protocol (Peng et al., 2024) with a learning rate of 2×10^{-5}. For context lengths of {4k, 16k, 64k}, we employed total batch sizes of {512, 64, 32}, and all models were fine-tuned for 400 steps.

Baseline. By fine-tuning with the aforementioned hyperparameters, we obtained YaRN-{16k, 64k} (Peng et al., 2024), LongRoPE-16k (Ding et al., 2024), and our proposed DCIS-{4k, 16k, 64k} models. The CLEX-16k (Chen et al., 2024) used the original model. Additionally, to compare with other types of methods, we employed the InfLLM (Xiao et al., 2024a), which claims to support infinitely long context windows by chunking and storing text sequences and retrieving the top-k most relevant chunks during inference.

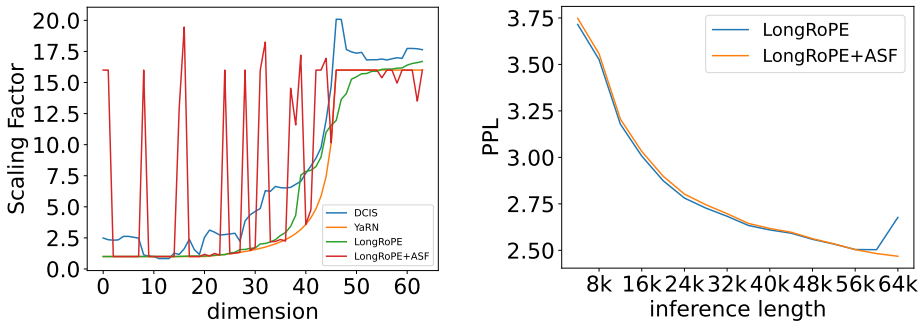
5.2 Main Results

We assessed the PPL of our proposed model and baseline models on the Proof-pile dataset, with results shown in Figure 1 and Table 1. Figure 3 depicts the performance on the passkey evaluation. Notably, models from other methods that were fine-tuned on 16k-length and subsequently generalized to a 64k context windows experienced a marked increase in PPL at the target length (64k). Furthermore, these models entirely failed the passkey test



(a) Llama2/64k/Proof-pile (b) Llama2/128k/Proof-pile (c) Llama3/64k/PG-19 (d) Mistral/64k/PG-19

Figure 4: PPL across different models, lengths, and datasets without fine-tuning.



(a) Scaling factor distributions. (b) PPL under two strategies.

Figure 5: Exploration of Adaptive Scaling Factor (ASF).

at the target length. In stark contrast, our approach exhibited consistent performance across various context lengths, including the target length, and even outperformed other methods at shorter fine-tuning lengths (4k). Additionally, when fine-tuned on a 64k-length context, DCIS consistently outperforms YaRN on both PPL and passkey metrics.

All results underscore the significance of scaling factors. DCIS identify superior scaling factors, leading to improved performance across various sequence lengths and demonstrating strong generalization capabilities, thereby reducing the memory, data, and time associated with model fine-tuning. The superior initial scaling factors contribute to further reduction in the number of fine-tuning steps. For example, in Appendix A.1, we report our observations that our method requires fewer fine-tuning steps to achieve comparable performance to YaRN.

5.3 DCIS without Fine-Tuning

Figure 4 illustrates the outcomes of our experiments wherein inference was conducted solely by adjusting scaling factors without fine-tuning. A comprehensive search for scaling factors was performed at target lengths of 64k and 128k, followed by direct PPL evaluation. Figures 4a and 4b, clearly indicate a consistent decline in our model's PPL values at both target lengths, in stark contrast to the upward trends observed in other methods. Furthermore, we extended our experiments to the newly released LLama3-8B and the different architecture, Mistral, employing the DCIS algorithm for scaling factors search, and evaluated the resulting models on the PG19 (Rae et al., 2020b) test set. Figures 4c and 4d, demonstrate that our approach consistently achieves the lowest PPL across all settings, further validating its broad applicability and effectiveness.

In Appendix A.2, we compare the PPL of scaling factors for various methods at shorter lengths, without fine-tuning.

6 Analysis

In this section, we delved into a comprehensive analysis of DCIS. We began by contrasting its search space with other search methods. Subsequently, we examined the implications of non-strictly increasing scaling factors. Furthermore, to evaluate the model's general ability on real-world tasks, we employed LongBench (Bai et al., 2024) and Open LLM Leaderboard from Hugging Face to assess the model's performance on long and short contexts. Finally, we conducted an in-depth investigation into the sensitivity analysis of hyperparameters introduced by the DCIS method, as well as the scalability performance of the proposed framework across different scenarios.

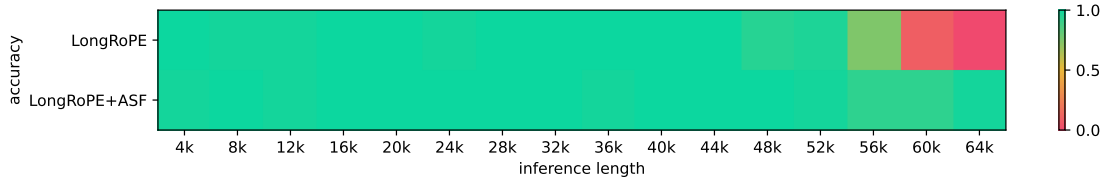


Figure 6: Recall rate under two strategies, the context length for fine-tuning is 16k.

Method	Fine-tuning Length	Categories					
		S-doc QA	M-doc QA	Sum	Few shot	Syn	Code
YaRN	16k	9.38	5.13	15.85	58.25	0.33	62.25
CLEX	16k	6.93	8.30	13.63	57.68	0.69	44.06
LongRoPE	16k	9.90	5.25	16.93	59.11	0.31	61.49
InfLLM	-	6.40	5.10	6.17	51.05	0.78	62.27
DCIS(Ours)	16k	7.58	3.51	16.11	59.10	0.42	62.05
	4k	8.17	4.09	14.55	58.07	0.25	61.67

Table 2: Evaluation of different methods on the LongBench benchmark.

6.1 Search Space Analysis

The efficiency of our DCIS algorithm is highlighted by comparing the search space. Specifically, our search space is the product of the total number of processed segments $d-2$ and the number of increments per segment C. The search space of evolutionary search utilized by LongRoPE is the product of the number of iterations T and the population size P. For example, for the Llama2-7B model with default parameters, our search space equates to $(d-2)\times C=(128-2)\times 10=1260$. In contrast, the search space of evolutionary search calculates as $T\times P=40\times 64=2560$, signifying that our algorithm's search speed is effectively double that of evolutionary search.

6.2 Non-Strictly Increasing Scaling Factor

Since LongRoPE utilizes strictly monotonically increasing scaling factors, we performed ablation studies on it to examine the effects of our proposed ASF. Specifically, we adjusted the evolutionary search algorithm in LongRoPE to a non-strictly increasing one, applied it to fine-tune the model on a 16k-length context, and subsequently generalized it to a 64k context window. Figures 5 and 6 depict the experimental outcomes. Figure 5a visualizes the scaling factor distributions employed by various methods. Notably, both DCIS and LongRoPE + ASF exhibited irregular, sawtooth-like scaling factors, while YaRN and the original LongRoPE demonstrated more stable scaling factors. Furthermore, as shown in Figures 5b and 6, our ASF can improve LongRoPE in terms of both PPL and passkey scores, suggesting that imposing fewer constraints on scaling factors may enhance model performance.

$[l,r]$	$[-3,3]$	$[-4,4]$	$[-5,5]$	$[-6,6]$	$[-7,7]$
PPL	10.2	10.2	10.2	10.2	10.2

Table 3: PPL for different initial ranges $[l,r]$ with fixed $C=10$.

C	6	8	10	12
PPL	10.4	10.2	10.2	10.2

Table 4: PPL for different C with fixed $[l,r]=[-5,5]$.

6.3 General Ability Evaluation

In addition to the previous assessments on PPL and passkey, employing LongBench and Open LLM Leaderboard, we conducted a comprehensive assessment of the models' general abilities in both long and short context scenarios. The empirical results, as depicted in Tables 2 and 7, indicate that there is no significant difference in performance among the various methods, with different models performing best on different subsets. While models utilizing full attention generally achieved slightly better results, InfLLM, which leverages a retrieval-based approach, demonstrated a notable advantage in terms of memory efficiency. These findings suggest that the optimal choice of model is contingent upon specific application requirements.

6.4 Hyperparameter Sensitivity

The DCIS algorithm introduces two critical hyperparameters: the initial search range $[l, r]$ and the number of increments per segment C.

Our experimental observations demonstrate that the DCIS algorithm exhibits the following characteristics during execution. 1) Coarse-grained iteration phase. In the initial iterations, the PPL values obtained from each sampling display a U-shaped distribution, indicating that the initial range $[l, r]$ already encompasses the optimal value region. 2) Fine-grained iteration phase. In subsequent refinement searches, PPL variations tend to stabilize, suggesting that the algorithm has converged to the vicinity of the optimal value.

Based on these observations, we can derive the following theoretical analysis. 1) Initial range $[l, r]$. As long as this range covers the region near the optimal value, its specific magnitude has a limited impact on the final results, rendering the algorithm insensitive to this parameter. 2) The number of increments per segment C. Theoretically, larger values are preferable. A greater sampling count implies higher search precision, as denser sampling can cover regions that sparser sampling might overlook. However, this simultaneously increases computational overhead.

To validate this theory, we employed Llama2-7B on 16k long texts, fixing two hyperparameters while adjusting one to compute the PPL values after applying the DCIS algorithm. As shown in Table 3, we fixed $C = 10$ and adjust the initial range $[l, r]$, and in Table 4, we fixed $[l, r] = [5, 5]$ and adjust C. The experimental results are consistent with our analysis:

1. Parameter robustness. The DCIS algorithm demonstrates excellent robustness to hyperparameter selection, where parameter variations within reasonable ranges do not significantly affect model performance.

2. Practical guidance. In practical applications, one can select moderate initial ranges (e.g., $[-5, 5]$) and the number of increments per segment (e.g., $C = 10$) to ensure both search effectiveness and computational cost control.

3. Algorithm stability. This parameter insensitive characteristic indicates that the DCIS algorithm possesses favorable stability and practicality.

Metric	Initial Value	Value After DCIS Search
PPL	19.25	10.19
LongPPL	12.99	2.64

Table 5: Comparison of different optimization objectives.

6.5 LongPPL Metric

In our framework, PPL serves primarily as the optimization objective function for searching scaling factors. This design offers excellent flexibility, allowing for substitution with other evaluation metrics according to specific requirements, such as LongPPL (Fang et al., 2025). To further validate the scalability of the proposed framework, we adjusted the optimization objective from PPL to LongPPL and conducted comparative experiments based on the Llama2-7B model on 16k text sequences.

The experimental results presented in Table 5 demonstrate that employing LongPPL as the optimization objective can similarly yield superior scaling factors, with performance essentially consistent with using PPL as the optimization objective. This finding further confirms the excellent scalability of the framework proposed in this paper under different evaluation metrics, while also validating the effectiveness and robustness of the method.

7 Conclusion

In this paper, we have presented DCIS, a novel and efficient algorithm for identifying effective RoPE scaling factors to significantly enhance LLM length extrapolation. Our framework demonstrably mitigates performance degradation at extended target lengths. Crucially, DCIS enables models fine-tuned on shorter contexts to generalize effectively to considerably longer sequences, substantially reducing fine-tuning costs. Furthermore, the scaling factors identified by DCIS yield improved extrapolation performance even without any fine-tuning. Our investigation into non-strictly increasing scaling factors revealed their benefits for model performance. Comparative analyses underscore the effectiveness, efficiency, and robustness of DCIS, offering a practical solution for extending the operational context window of LLMs.

Acknowledgments

The present research was supported by the National Key Research and Development Program of China (Grant No. 2023YFE0116400). We would like to thank the anonymous reviewers for their insightful comments.

Limitations

We conducted a search for a scaling factors of 128k on a small model, Phi-3-mini-4k-instruct (Abdin et al., 2024), with an actual context window size of 2k. We observe that the model's PPL remains around 70, with no significant decrease. Thus, such search algorithms appear to have certain requirements for the model's inherent capabilities and are unable to span too large a scaling factor at once. This also demonstrates that, relative to YaRN, which has a smaller scaling factor, the scaling factors identified through search have larger scaling factor, thereby more fully leveraging the model's extrapolative potential.

References

Marah I Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat S. Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, Allie Del Giorno, Gustavo de Rosa, Matthew Dixon, Ronen Eldan, Dan Iter, Amit Garg, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Jamie Huynh, Mojan Javaheripi, Xin Jin, Piero Kauffmann, Nikos Karampatziakis, Dongwoo Kim, Mahoud Khademi, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Chen Liang, Weishung Liu, Eric Lin, Zeqi Lin, Piyush Madan, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Xia Song, Masahiro Tanaka, Xin Wang, Rachel Ward, Guanhua Wang, Philipp A. Witte, Michael Wyatt, Can Xu, Jiahang Xu, Sonali Yadav, Fan Yang, Ziyi Yang, Donghan Yu, Chengruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. 2024. Phi-3 Technical Report: A Highly Capable Language Model Locally on Your Phone. *CoRR*, abs/2404.14219.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*.

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. 2024. LongBench: A Bilingual, Multitask Benchmark for Long Context Understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 3119–3137. Association for Computational Linguistics.

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. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Guanzheng Chen, Xin Li, Zaiqiao Meng, Shangsong Liang, and Lidong Bing. 2024. CLEX: Continuous Length Extrapolation for Large Language Models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023. Extending Context Window of Large Language Models via Positional Interpolation. *CoRR*, abs/2306.15595.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc Viet Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2978–2988. Association for Computational Linguistics.

Jiayu Ding, Shuming Ma, Li Dong, Xingxing Zhang, Shaohan Huang, Wenhui Wang, Nanning Zheng, and Furu Wei. 2023. LongNet: Scaling Transformers to 1, 000, 000, 000 Tokens. *CoRR*, abs/2307.02486.

Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. 2024. LongRoPE: Extending LLM Context Window Beyond 2 Million Tokens. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. 2024. The Llama 3 Herd of Models. *CoRR*, abs/2407.21783.

Lizhe Fang, Yifei Wang, Zhaoyang Liu, Chenheng Zhang, Stefanie Jegelka, Jinyang Gao, Bolin Ding, and Yisen Wang. 2025. What is Wrong with Perplexity for Long-context Language Modeling? In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2021. The Pile: An 800GB Dataset of Diverse Text for Language Modeling. *CoRR*, abs/2101.00027.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional Sequence to Sequence Learning. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1243–1252. PMLR.

Zishan Guo, Renren Jin, Chuang Liu, Yufei Huang, Dan Shi, Supryadi, Linhao Yu, Yan Liu, Jiaxuan Li, Bojian Xiong, and Deyi Xiong. 2023. Evaluating Large Language Models: A Comprehensive Survey. *CoRR*, abs/2310.19736.

Chi Han, Qifan Wang, Wenhan Xiong, Yu Chen, Heng Ji, and Sinong Wang. 2023. LM-Infinite: Simple On-the-Fly Length Generalization for Large Language Models. *CoRR*, abs/2308.16137.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. *CoRR*, abs/2310.06825.

Hongye Jin, Xiaotian Han, Jingfeng Yang, Zhimeng Jiang, Zirui Liu, Chia-Yuan Chang, Huiyuan Chen, and Xia Hu. 2024. LLM Maybe LongLM: Self-Extend LLM Context Window Without Tuning. *CoRR*, abs/2401.01325.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the Middle: How Language Models Use Long Contexts. *Trans. Assoc. Comput. Linguistics*, 12:157–173.

LocalLLaMA. 2023. NTK-Aware Scaled RoPE allows LLaMA models to have extended (8k+) context size without any fine-tuning and minimal perplexity degradation.

Amirkeivan Mohtashami and Martin Jaggi. 2023. Landmark Attention: Random-Access Infinite Context Length for Transformers. *CoRR*, abs/2305.16300.

Leiyu Pan, Bojian Xiong, Lei Yang, Renren Jin, Shaowei Zhang, Yue Chen, Ling Shi, Jiang Zhou, Junru Wu, Zhen Wang, Jianxiang Peng, Juesi Xiao, Tianyu Dong, Zhuowen Han, Zhuo Chen, Yuqi Ren, and Deyi Xiong. 2025. Advancing Large Language Models for Tibetan with Curated Data and Continual Pre-Training. *CoRR*, abs/2507.09205.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2024. YaRN: Efficient Context Window Extension of Large Language Models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Ofir Press, Noah A. Smith, and Mike Lewis. 2022. Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. 2020a. Compressive Transformers for Long-Range Sequence Modelling. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. 2020b. Compressive Transformers for Long-Range Sequence Modelling. In *8th International Conference on*

Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, 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. 2023. Code Llama: Open Foundation Models for Code. *CoRR*, abs/2308.12950.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-Attention with Relative Position Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 464–468. Association for Computational Linguistics.

Tianhao Shen, Renren Jin, Yufei Huang, Chuang Liu, Weilong Dong, Zishan Guo, Xinwei Wu, Yan Liu, and Deyi Xiong. 2023. Large Language Model Alignment: A Survey. *CoRR*, abs/2309.15025.

Jianlin Su. 2023. Rectified Rotary Position Embeddings. `https://github.com/bojone/rerope`.

Jianlin Su, Murtadha H. M. Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. RoFormer: Enhanced transformer with Rotary Position Embedding. *Neurocomputing*, 568:127063.

Haoran Sun, Renren Jin, Shaoyang Xu, Leiyu Pan, Supryadi, Menglong Cui, Jiangcun Du, Yikun Lei, Lei Yang, Ling Shi, Juesi Xiao, Shaolin Zhu, and Deyi Xiong. 2024. FuxiTranyu: A Multilingual Large Language Model Trained with Balanced Data. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: EMNLP 2024 - Industry Track, Miami, Florida, USA, November 12-16, 2024*, pages 1499–1522. Association for Computational Linguistics.

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, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. *CoRR*, abs/2307.09288.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. 2025. Retrieval Head Mechanistically Explains Long-Context Factuality. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net.

Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. 2022. Memorizing Transformers. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Chaojun Xiao, Pengle Zhang, Xu Han, Guangxuan Xiao, Yankai Lin, Zhengyan Zhang, Zhiyuan Liu, Song Han, and Maosong Sun. 2024a. InfLLM: Unveiling the Intrinsic Capacity of LLMs for Understanding Extremely Long Sequences with Training-Free Memory. *CoRR*, abs/2402.04617.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2024b. Efficient Streaming Language Models with Attention Sinks. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Lei Yang, Renren Jin, Ling Shi, Jianxiang Peng, Yue Chen, and Deyi Xiong. 2025. ProBench: Benchmarking Large Language Models in Competitive Programming. *CoRR*, abs/2502.20868.

A Additional Results

We present supplementary results obtained during our experimental process in this section.

A.1 Fewer Fine-Tuning Steps

During the fine-tuning process of YaRN and DCIS, we observed that our method, DCIS, achieved PPL and passkey test results consistent with, or even superior to, those of YaRN after only 100 steps of fine-tuning, as depicted in Table 6 and Figure 7. It is demonstrated that an initially superior scaling factors requires fewer fine-tuning steps.

A.2 Shorter Length of PPL without Fine-Tuning

Even at shorter lengths of 16k and 32k, our scaling factors consistently achieved the lowest PPL, as illustrated in Figure 8.

A.3 Benchmarks

As shown in Table 7 of the Open LLM Leaderboard, there is no clear superiority among different methods. Furthermore, the models with extended context windows do not exhibit significant performance degradation on short texts compared to their original counterparts.

Method	Fine-tuning Steps	Evaluation Context Window Size								
		4k	8k	16k	24k	32k	40k	48k	56k	64k
YaRN	100	3.71	3.53	3.01	2.78	2.68	2.60	2.54	2.48	2.45
	200	3.71	3.53	3.01	2.78	2.68	2.60	2.54	2.48	2.44
	400	3.71	3.52	3.00	2.78	2.67	2.59	2.53	2.47	2.44
DCIS(Ours)	100	3.72	3.53	3.00	2.77	2.67	2.59	2.53	2.47	2.43
	200	3.72	3.53	3.01	2.77	2.67	2.59	2.53	2.47	2.43
	400	3.71	3.52	3.00	2.77	2.66	2.58	2.52	2.46	2.42

Table 6: Comparison of the PPL results at different numbers of steps of fine-tuning on the 64k length.

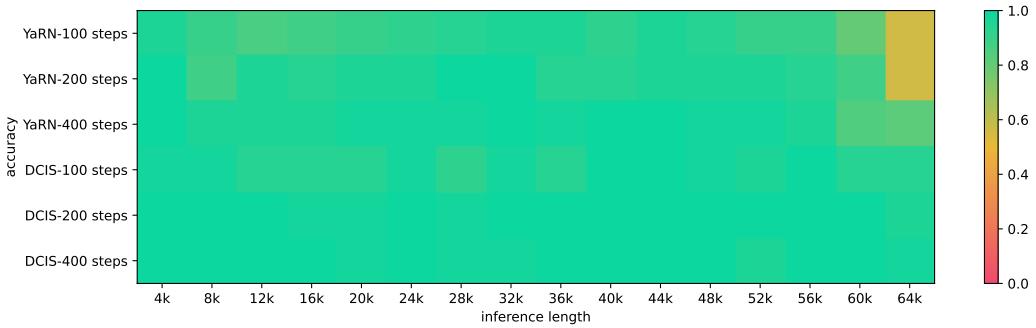


Figure 7: Comparison of the passkey results at different numbers of steps of fine-tuning on the 64k length.

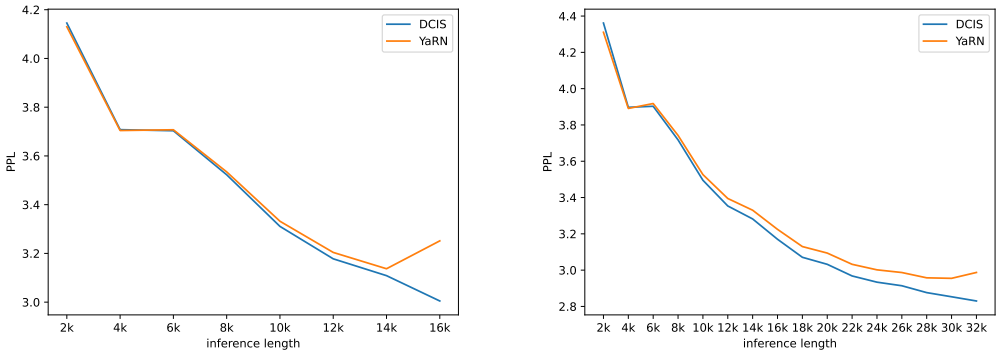


Figure 8: Left: 16k-length. Right: 32k-length.

Method	Fine-tuning Length	ARC-c	Hellaswag	MMLU	TruthfulQA
Original	-	52.6	79.0	46.4	39.0
YaRN	64k	52.8	78.8	42.1	39.0
	16k	52.6	78.4	42.4	38.4
CLEX	16k	52.3	78.3	42.1	41.3
DCIS(Ours)	64k	52.3	78.4	41.8	39.2
	16k	53.0	78.2	41.4	38.8
	4k	52.6	78.4	43.7	38.0

Table 7: Evaluation of different methods on the benchmarks from the Hugging Face Open LLM Leaderboard