S^3PE: A Simple Scalable Sigmoid-Style Position Encoding

Zhu Zhang[1], Tianxing Yang[1], Zihan Yan[1]
[1]Department of Computer Science and Technology, Tsinghua University
{zhuzhang24,ytx24,yanzh24}@mails.tsinghua.edu.cn

Abstract

Long context continuous pretraining enables Transformer-based large language models (LLMs) to comprehend input sequences within a larger context window than pretraining stage. Common modifications to positional encoding involve interpolation methods, such as PI, NTK-aware, ABF, YaRN, and LongRoPE. While these positional encodings have proven effective, they nonetheless exhibit certain oversights. In this study, we demonstrate that these positional encodings can be expressed within a unified functional framework. Building on this insight, we propose a guiding principle for optimal positional encoding interpolation, leading to the introduction of a novel positional encoding scheme, S^3PE, designed to approximate this theoretical optimal solution. We conducted length extrapolation experiments across models of varying scales, comprehensively comparing existing mainstream positional encoding approaches. The results indicate that S^3PE consistently outperforms current mainstream positional encodings across all configurations. Our research illustrates that S^3PE provides a more robust solution for long-context modeling, demonstrating superior performance in length extrapolation scenarios.

1 Introduction

The Transformer model has become a foundational component of large language models (LLMs) and dominates the field of natural language processing. Well-known open-source models, such as the LLaMA series [1, 2, 3], are based on the Transformer architecture and are widely adopted due to their outstanding performance across a variety of tasks. Despite the tremendous success of Transformers, their quadratic computational complexity poses challenges when handling long sequences, as directly training with extended contexts is both costly and resource-intensive. To mitigate these issues, a common approach is to pre-train on shorter sequences (e.g., 4k tokens) to develop robust language modeling capabilities within a 4k context window. Subsequently, the model is continuously pre-trained on longer sequences (e.g., 32k tokens) to expand the context window to 32k. This method, known as **length extrapolation**, is feasible because the number of training tokens required for continuous pre-training is significantly lower than that in the initial pre-training phase. In simple terms, less continuous pre-training allows the model to generalize from a short context window to a long context window.

Positional encoding interpolation plays a crucial role in implementing length extrapolation. Since the introduction of Rotatory Positional Encoding (RoPE) [4], it has been widely adopted by many large models due to its excellent performance across various tasks. Typical pre-training setups often set the base frequency of RoPE to 10,000 and train on sequences of 4k tokens. However, when the input length exceeds the original context window, out-of-distribution (OOD) issues may arise, necessitating adjustments to the positional encoding. To address this limitation, various RoPE variants have been proposed, including PI [5], ABF [6], NTK [7], and YaRN [8]. Despite the differing implementations

Preprint. Under review.

of these variants, they share a common goal: to introduce mechanisms for extension to enhance performance in long contexts.

Although these RoPE variants are meticulously designed, there is no definitive evidence to suggest that any single variant consistently outperforms the others across all scenarios. In fact, existing positional encoding interpolation schemes exhibit certain oversights to varying degrees. Through analysis, we have found that all RoPE-based positional encodings can be expressed within a unified functional framework, implying that each variant is merely a specific instance within this broader functional space, as illustrated in Figure 1. The principles and shortcomings of existing positional encodings will be elaborated in Section 2.2. Within this unified functional space, we hypothesize the existence of a positional encoding that can comprehensively surpass current methods in length extrapolation scenarios. Based on this insight, we propose three guiding principles, under which we introduce a new positional encoding, **S^{3}PE**. We employ the NIAH score from RULER [9] as an evaluation metric to validate the effectiveness of S^{3}PE across multiple model scales. Experimental results indicate that S^{3}PE consistently outperforms the aforementioned mainstream positional encodings.

In summary, our contributions are as follows:

- We provide a theoretical unification framework for all existing positional interpolation methods based on RoPE.
- We propose guiding principles for optimal positional encoding interpolation, leading to the introduction of a novel positional encoding, S^{3}PE, designed to approximate this theoretical optimal solution.
- We conducted length extrapolation experiments across models of varying scales, comprehensively comparing existing mainstream positional encoding approaches. The results demonstrate that S^{3}PE consistently outperforms current mainstream positional encodings in all configurations.

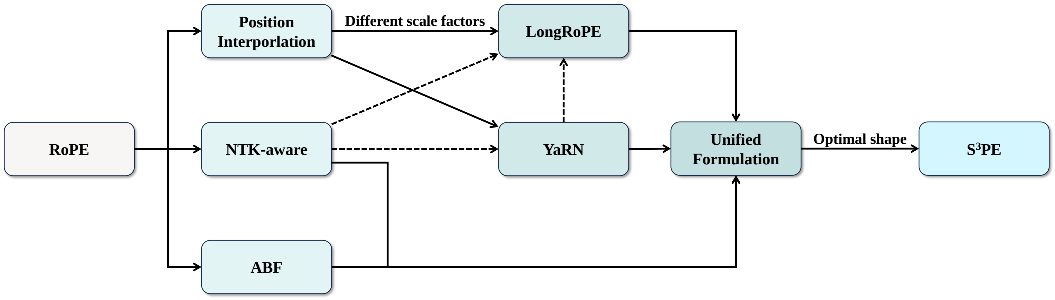

Figure 1: An introduction of S^{3}PE

2 Mainstream Position Encodings

2.1 Preliminary

RoPE [4] encodes positional information by applying phase rotation to each element of the query and key vectors before calculating the attention scores. Formally, we define a transformation $\mathbf{f}$ as follows:

$$\mathbf{f}_{\mathbf{W}}(\mathbf{x}_t, \boldsymbol{\theta}) = \mathbf{R}(\boldsymbol{\theta}, t)\mathbf{W}\mathbf{x}_t,$$

where $\mathbf{x}_t \in \mathbb{R}^d$ is the input at position t, $\mathbf{W}$ is the projection matrix, and $\boldsymbol{\theta} \in \mathbb{R}^{d/2}$ represents the frequency basis. The rotation transformation matrix $\mathbf{R}(\boldsymbol{\theta}, t)$ is defined as:

$$\mathbf{R}(\boldsymbol{\theta}, t) = \begin{pmatrix} \mathbf{R}_1(i\theta_1) & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_2(i\theta_2) & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{R}_{\frac{d}{2}}(i\theta_{\frac{d}{2}}) \end{pmatrix},$$

where

$$\mathbf{R}_k(i\theta_k) = \begin{pmatrix} \cos(i\theta_k) & -\sin(i\theta_k) \\ \sin(i\theta_k) & \cos(i\theta_k) \end{pmatrix}.$$

This matrix possesses the property:

$$\mathbf{R}(\boldsymbol{\theta}, n-m) = \mathbf{R}(\boldsymbol{\theta}, m)^\top \mathbf{R}(\boldsymbol{\theta}, n).$$

Thus, the relative positional information $n-m$ is implicitly encoded in the attention scores through the query-key product. In standard RoPE, the components of $\boldsymbol{\theta}$ are defined as $\theta_j = b^{-\frac{2j}{d}}$, where the base frequency $b = 10,000$.

$$\mathbf{f}_{\mathbf{W}}(\mathbf{x}_t, \boldsymbol{\theta}) = \mathbf{R}(\boldsymbol{\theta}, t)\mathbf{W}\mathbf{x}_t,$$

2.2 RoPE-based Position Interpolation Methods

Existing RoPE-based positional encodings can be categorized into two types based on modifications relative to RoPE: increasing the base frequency or modifying interpolation factors. Modifications to the scaling factor can also be divided into two categories: those that are related to the expansion factor of the context window (e.g., PI, NTK, YaRN, LongRoPE) and those that use a scaling factor greater than the context window without a clear numerical correlation (e.g., ABF). Interestingly, while ABF is widely adopted by the majority of current long-text large models, there is a lack of comparative studies examining ABF against other positional encodings under the same interpolation factors. Researchers seem to assume that ABF and other positional encoding schemes are independent techniques; however, they are, in fact, equivalent. We will elaborate on this in this section.

PI

In PI[5], the RoPE function $\mathbf{f}$ is replaced with $\mathbf{f}'$, expressed as follows:

$$f^{\text{PI}}(x,t)_j = (x_{2j} + ix_{2j+1})e^{i(b^{-\frac{2j}{d}})t/s}$$

where the expansion factor of the context window is $s = \frac{L'}{L}$, $\mathbf{x}$ is the input sequence, and t is the token index. Figure 3 illustrates the positional encoding interpolation when $L = 4096$ and $s = 2$. As shown in Figure 2, PI can interpolate token indices that exceed the original context window into the well-trained periodic range during the pre-training phase, thereby avoiding OOD issues. Consequently, compared to the training costs associated with directly extrapolating from L to L', the expense of extending the context window will be significantly reduced.

In Figure 2, we observe that the periods of different dimensions in RoPE vary, with smaller dimensions having larger periods. PI treats each dimension equally and scales them by the factor corresponding to the increase in the context window. Although experiments show that PI is effective, subsequent research suggests that interpolation factors should not be treated equally.

NTK-aware

NTK-aware[10] is an improvement over linear position interpolation. It posits that linear interpolation is very sub-optimal, preventing neural networks from distinguishing the order and positions of closely spaced tokens. Consequently, the idea behind NTK-aware is to directly modify the base frequency of RoPE, thereby altering the "spinning speed" of different dimensions in RoPE. Specifically, it is defined as:

$$f^{\text{NTK}}(x,t)_j = (x_{2j} + ix_{2j+1})e^{i(s^{\frac{d}{d-2}}b)^{-\frac{2j}{d}}t}$$

The characteristic of NTK-aware is that when j takes the minimum value of 0, the corresponding angular frequency is highest, making NTK-aware equivalent to the original RoPE. Conversely, when j takes the maximum value of $\frac{d-2}{2}$, the corresponding angular frequency is lowest, making NTK-aware equivalent to PI. Thus, it can be seen as a trade-off between RoPE and PI. However, compared to PI, NTK-aware makes fewer modifications to the high-frequency dimensions, resulting in better training-free extrapolation performance. The idea of non-linear interpolation across different dimensions in NTK-aware has inspired subsequent models such as YaRN and LongRoPE.

ABF

Adjusted Base Frequency (ABF) builds on the concept of non-linear interpolation introduced by NTK but modifies the base frequency more directly, without being constrained by the expansion factor of the context window. In this sense, NTK-aware can be considered a special case of ABF. Let β be the factor by which the base frequency is increased, then:

$$f^{\text{ABF}}(x,t)_j = (x_{2j} + ix_{2j+1})e^{i(\beta b)^{-\frac{2j}{d}} t}$$

In this formulation, NTK-aware corresponds to the case where $\beta = s^{\frac{d}{d-2}}$. Interestingly, the modifications for β are often much greater than the factors used for expanding the context window. For example, when $L = 4096$ and $s = 8$, a common setting for β might be 50 or even larger. ABF posits that a larger "granularity" allows the model to better differentiate positional embedding images, thereby improving performance on downstream tasks, which is consistent with the intuition behind NTK-aware. Additionally, this work theoretically demonstrates that the granularity of ABF is greater than that of PI, even if the modification factor for ABF is several times larger than that for PI.

However, ABF does not analyze why the modification factor of $\beta = 50$ is chosen. This seems more like an experimental conclusion rather than a theoretical derivation, suggesting that using a modification factor greater than s yields better performance after fine-tuning. Despite this, the experimental conclusions of ABF have been widely adopted in other open-source large models, such as LLaMA3. ABF sets a precedent for exploring the maximum factors in positional encoding interpolation but has not seen subsequent research that follows up on this discovery. Instead, there seems to be a tendency to separate positional encoding interpolation schemes from strategies for modifying the base frequency. For instance, LLaMA3.1 simultaneously employs both ABF and YaRN. This paper will conduct ablation studies in the Experiment section on various positional encoding schemes with different interpolation factors, including ABF, to provide a more comprehensive comparative analysis.

YaRN

YaRN also adopts the non-linear interpolation concept from NTK-aware and employs piecewise functions to further articulate this idea. In essence, YaRN uses the wavelength along with two hyperparameters, $\tilde{\alpha}$ and $\tilde{\beta}$, to calculate the boundaries j_{low} and j_{high} for applying different interpolation strategies, resulting in the following piecewise function:

$$f^{\text{YaRN}}(x,t)_j = (x_{2j} + ix_{2j+1})e^{i(b)^{-\frac{2j}{d}} t/\alpha_j} \cdot m,$$

where m is the attention temperature coefficient, defined as:

$$\alpha_j = \begin{cases} 1, & j < j_{\text{low}} \\ s, & j \geq j_{\text{high}} \\ \left[\frac{1}{s} + \left(1 - \frac{1}{s}\right) \cdot \frac{\frac{L}{2\pi} b^{-\frac{2j}{d}} - \tilde{\alpha}}{\tilde{\beta} - \tilde{\alpha}}\right]^{-1}, & \text{else} \end{cases}$$

Since the attention temperature coefficient m is typically a small variable slightly greater than 1, or even a constant, this study temporarily fixes m at 1 to facilitate a clearer comparison of the effects of interpolation factors.

Considering the piecewise definition of α_j, it is noted that YaRN further develops the ideas of NTK-aware and ABF by minimizing modifications to the high-frequency components, directly leaving the highest-frequency dimensions unchanged. Despite its careful design, YaRN's scaling factor choices are still somewhat limited to those of NTK-aware and PI. It does not fully recognize that, in the context of fine-tuning, increasing the maximum scaling factor may yield greater benefits than a meticulously designed interpolation strategy. Consequently, the conclusions drawn from YaRN also carry certain limitations.

LongRoPE

LongRoPE is inspired by NTK-aware and YaRN, applying different interpolation factors across the various dimensions of RoPE. It is defined as:

$$f^{\text{LongRoPE}}(x,t)_j = (x_{2j} + ix_{2j+1})e^{i(b^{-\frac{2j}{d}})t/\lambda_j}$$

where λ_j is the interpolation factor for the corresponding dimension, determined through a search process based on evolutionary algorithms. This search aims to minimize perplexity (PPL) on a given dataset without fine-tuning the model. The values for λ_j are constrained by the following conditions:

$$\begin{cases} \lambda_{j+1} \geq \lambda_j \\ \lambda_j \in [1, 1.25 \times s] \end{cases}$$

where s is the factor by which the context window is expanded. The search strategy used in LongRoPE offers a new perspective on improving the training-free performance of the model. However, this search-based approach has several drawbacks: (1) the complexity of the search process; (2) the interpolation factors obtained may be model-specific and difficult to generalize; (3) the results of the search can be influenced by the initialization method. Most importantly, interpolation factors optimized for training-free perplexity reduction may not remain optimal after fine-tuning. Additionally, like the other interpolation methods mentioned (excluding ABF), LongRoPE does not explore maximum interpolation factors that are several times greater than s.

The LongRoPE paper also mentions two strategies: not applying interpolation for the first $\hat{n}$ tokens and performing recovery within the original context window. The former will not be discussed further here, as it undermines the foundational properties of RoPE. For the latter, LongRoPE essentially adopts the strategy of using one set of interpolation factors within the original context window, while applying a different set for sequences longer than the original context window. To facilitate a comparison of the effects of interpolation factor shapes, we will not examine scenarios with multiple sets of interpolation factors.

3 Methods

Let $\mathbb{S}_N = \{w_t\}_{t=1}^N$ denote a sequence of N input tokens, where w_t represents the t-th token. The corresponding word embeddings for $\mathbb{S}_N$ are denoted as $\mathbb{E}_N = \{\boldsymbol{x}_t\}_{t=1}^N$, where $\boldsymbol{x}_t \in \mathbb{R}^d$ is the d-dimensional word embedding vector for the token w_t. For $j \in [0, \frac{d}{2})$, the RoPE positional encoding can be expressed as:

$$f^{\text{RoPE}}(x,t)_j = (x_{2j} + ix_{2j+1})e^{i(b^{-\frac{2j}{d}})t} \tag{1}$$

In this section, we will discuss several existing positional encodings and provide visualizations from a unified perspective.

3.1 Unified Form

Let $\mathbf{S}_N = \{w_t\}_{t=1}^N$ denote a sequence containing N input tokens, where w_t represents the t-th token. The corresponding hidden states are denoted as $\mathbf{X}_N = \{\boldsymbol{x}_t\}_{t=1}^N$, with $\boldsymbol{x}_t \in \mathbb{R}^d$ as a d-dimensional vector. For $j \in [0, \frac{d}{2})$, the RoPE positional encoding can be expressed as:

$$f^{\text{RoPE}}(\boldsymbol{x},t)_j = (x_{2j} + ix_{2j+1})e^{i(b^{-\frac{2j}{d}})t}$$

It is easy to observe that modifying the base frequency is equivalent to modifying the interpolation factors, specifically:

$$e^{i(\beta b)^{-\frac{2j}{d}}t} = e^{ib^{-\frac{2j}{d}}\cdot(t/\beta^{\frac{2j}{d}})}$$

Thus, we can unify the various positional encoding interpolation schemes mentioned in Section 2.2, including ABF and NTK-aware:

$$f(\boldsymbol{x},t)_{j,m} = (x_{2j} + ix_{2j+1})e^{ib^{-\frac{2j}{d}}(t/\alpha_j)} \cdot m$$

where α_j represents the different interpolation factors, and m denotes the attention temperature coefficient. To focus on the influence of the shape and maximum scaling of α_j, we set $m = 1$. Clearly, the various positional encodings mentioned in Section 2.2 are all special cases of this unified form. In this way, modifications to positional encoding, whether applied to token position t or base frequency b, are equivalent to modifying α_j.

By expanding the positional encoding with Euler's formula, we obtain:

$$e^{i(b^{-\frac{2j}{d}})t/\alpha_j} = \cos\left(\frac{b^{-\frac{2j}{d}}t}{\alpha_j}\right) + i\sin\left(\frac{b^{-\frac{2j}{d}}t}{\alpha_j}\right)$$

indicating that the exponential term corresponds to the angular frequency of the trigonometric functions. Within this unified form, we can plot the α_j corresponding to all positional encoding schemes mentioned in Section 2.2 to facilitate visual comparison. For ABF, we fix the hyperparameter

β to align with other positional encodings for comparison. For YaRN, we follow the settings in the original paper, with $\tilde{\alpha} = 1$ and $\tilde{\beta} = 32$. For LongRoPE, we use the official codebase to search for a set of factors based on PPL reduction (as prescribed, setting the maximum interpolation factor to $s = 8 \times 1.25 = 10$). In the case where $d = 64$ and $s = 8$, the visualization of different positional encodings is shown in Figure 2 (a)-(e).

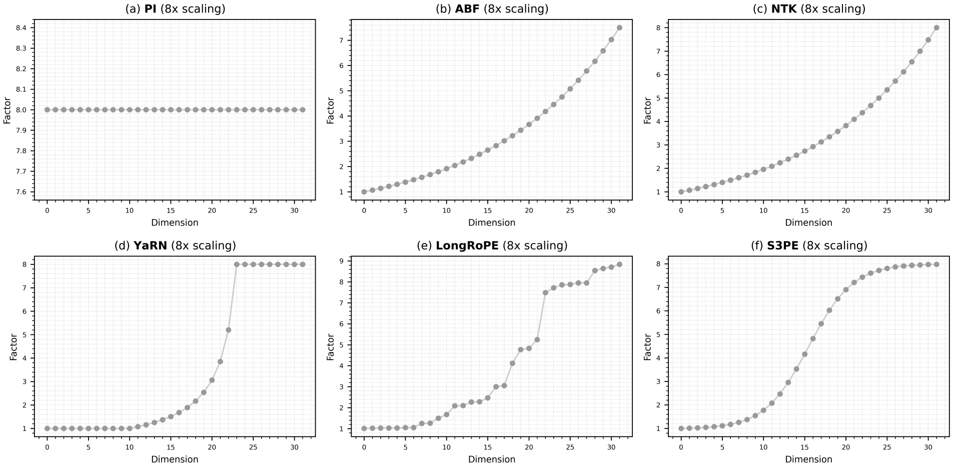

Figure 2: Visualized scaling factors of all position encodings when extension ratio is 8.

3.2 S^{3}PE

The positional encodings discussed in Section 2.2 either overlook the fact that the maximum interpolation factor can be multiple times greater than the context window expansion factor s (e.g., PI, NTK-aware, YaRN, LongRoPE), or they ignore the diversity of interpolation factor shapes (e.g., ABF). By leveraging the unified formulation presented in Section 3.1, it becomes evident that all of these positional encodings are mathematically special cases, each with varying degrees of limitations. In conducting ablation studies on positional encoding interpolation schemes, each shape of interpolation factor maps one-to-one to a set of fine-tuning evaluation scores, serving as a measure of the optimality of the positional encoding. We hypothesize that the encodings mentioned in Section 2.2 are likely far from optimal and that an interpolation scheme exists which can outperform these positional encodings.

To this end, we conducted comprehensive experiments to compare the impact of the maximum interpolation factor across various positional encoding shapes. Through comparative testing of multiple shapes, we identified the optimal interpolation factor shape, derived from transformations of a Sigmoid function through translation and scaling. For this shape, we further experimented with different symmetry centers and curve slopes, ultimately identifying the pattern illustrated in Figure (f), which we denote as S^{3}PE. The shape of S^{3}PE is both elegant and simple, yet it demonstrates superior performance over other positional encoding schemes. Additionally, S^{3}PE is compatible with computation acceleration techniques, such as flash attention, positioning it as a promising candidate for positional encoding.

As with prior work, the form of S^{3}PE is rigorously derived from theoretical analysis. However, for the optimal shape of the interpolation factor function α_j, we make the following observations, which align closely with the shape of S^{3}PE:

- $\alpha_{j+1} \geq \alpha_j \geq 1$;
- When j is closer to 0, α_j approaches 1, with a slower growth rate;
- When j is closer to the maximum value $\frac{d}{2} - 1$, α_j remains at a high level, satisfying $\alpha_j > s$.

For (1), this nonlinearity has already been validated by a substantial number of experiments involving the positional encodings discussed in Section 2.2. Regarding (2), insights can be drawn from the ABF

Table 1: Training hyper parameter settings of different model sizes

Model	Model Size	Batch Tokens	Warmup Steps	Learning Rate	Training Steps
s2	0.031B	128k	200	2e-4	3000
s3	0.106B	256k	200	2e-4	3000
s4	0.251B	512k	200	2e-4	3000
s5	0.486B	1M	200	2e-4	3000
s6	0.849B	2M	250	2e-4	2500

paper's theoretical analysis on "granularity," suggesting that larger modifications to high-frequency components reduce the model's ability to discern minimal token index differences. Indeed, as long as conditions (1) and (2) are satisfied, the model tends to achieve better performance in extended context windows after fine-tuning. Condition (3) represents a key distinction between S^{3}PE and ABF, which may explain why S^{3}PE consistently outperforms ABF across various settings for α_j's maximum values ($s' = 8$ or $s' = 50$), as evidenced in Section 4. We hypothesize that this is due to the fact that when j is large, the rotational period corresponding to RoPE's angular frequency far exceeds the pre-training length of 4k, while most pre-training data is shorter than 4k tokens. This suggests that for lower-frequency dimensions of j, the model achieves optimal recognition within a range potentially smaller than the 4k pre-training length, say $[0, L_0]$. Hence, the selected maximum factor would be $s' = \frac{L}{L_0} \times s$. For $s' = 50$, $s = 8$, and $L = 4096$, this corresponds to an approximate L_0 of 655, which is plausible. Verification of these findings is presented in Section 4.

4 Experiments

Data Recipe. To simulate the most realistic long-text extrapolation scenario, we carefully adjusted the data composition during continuous pretraining, drawing on previous research. Specifically, we followed the stable pretraining data recipe and applied per-source upsampling, an effective strategy validated by [11]. This involved setting token proportions for each dataset as follows: 25% CommonCrawl Chinese, 25% Code Pretrain, 24% Dolma, 15% C4, 8% Pile, and 3% other sources, upsampling based on data length accordingly. Additionally, following practices from LLaMA3.1[3] and GLM4, we ensured the token count within each interval was proportional to the interval length during data sampling. For example, the token count within the [4k, 8k] range is approximately half that of the [8k, 16k] range.

Training Settings. To evaluate the performance of S^{3}PE and other positional encodings across models of various scales, we conducted sandbox experiments with a series of checkpoints at different model sizes from the pretraining phase. These checkpoints ranged in size from 0.031B to 0.849B, with architectures similar to LLaMA2, and were pretrained on texts within a 4k length limit. Using the data recipe outlined above, we continued pretraining these checkpoints on data with a 32k length mix. We applied the WSD learning rate strategy from [12], using its first two phases: linearly increasing the learning rate from zero to the peak rate and then holding it steady. Although the lack of a decay phase might mean the models do not reach their optimal performance levels, this has no impact on our conclusions, as we are only conducting a comparative analysis of different positional encodings. As model size increases, we progressively increase tokens per batch. For each experiment, the hyperparameter settings for models of different sizes are provided in Table 1, following the setting of [13].

We denote a positional encoding with a maximum interpolation factor $s' = k$ as PE$_{k\times}$. For example, when $s' = 8$, the function graph of α_j is shown in Figure 2. It is worth noting that for the typical setting of a practical context expansion factor $s = 8$, the comparable existing positional encodings include ABF$_{50\times}$, PI$_{8\times}$, NTK$_{8\times}$, YaRN$_{8\times}$, and LongRoPE$_{8\times}$. However, to ensure rigor, we conducted experiments on each positional encoding with both $s' = 8$ and $s' = 50$, providing a thorough comparison. For each model size, experiments were conducted to comprehensively verify whether S^{3}PE's performance remains robust.

Evaluation Methods. It should be noted that relying solely on PPL to evaluate model performance is insufficient, as a lower PPL does not necessarily indicate better performance, and differences in performance may exist even when PPL values are very close. Furthermore, given the presence of smaller models in our experiments, which may perform less effectively on complex tasks such

Table 2: RULER NIAH Average Scores' Comparison

Model	ABF		PI		NTK		YaRN		LongRoPE		S^{3}PE	
	8×	[50×]	[8×]	50×	[8×]	50×	[8×]	50×	[8×]	50×	8×	50×
s2	12.65	17.84	12.87	9.14	12.76	17.92	10.84	20.28	16.33	**23.59**	19.16	22.28
s3	25.09	34.25	26.17	20.64	25.09	34.61	25.51	31.06	28.97	36.02	31.34	**37.93**
s4	32.27	41.31	30.61	34.97	31.66	40.36	35.67	39.09	35.23	42.72	36.63	**43.09**
s5	36.69	40.56	40.82	41.22	36.42	41.63	39.05	41.77	39.70	43.27	40.31	**44.95**
s6	47.13	54.44	50.20	54.02	48.52	54.15	51.70	52.02	51.06	54.59	53.02	**56.75**

Table 3: RULER NIAH 16k-32k Scores' Comparison

Model	ABF		PI		NTK		YaRN		LongRoPE		S^{3}PE	
	8×	[50×]	[8×]	50×	[8×]	50×	[8×]	50×	[8×]	50×	8×	50×
s2	2.62	3.19	2.12	1.06	2.38	2.94	2.56	**8.81**	3.50	5.31	4.56	6.00
s3	3.06	24.44	7.50	9.31	2.94	26.56	7.31	20.69	9.69	29.56	11.94	**31.88**
s4	11.31	32.62	15.12	31.44	10.25	30.62	24.19	32.12	20.31	38.38	21.00	**39.88**
s5	14.75	34.69	31.31	35.81	14.31	34.44	28.75	39.62	30.19	42.19	31.31	**42.94**
s6	18.12	42.44	34.69	44.38	19.94	43.56	37.75	38.25	36.06	44.50	35.38	**48.00**

as summarization and inference, we employ RULER's NIAH (Needle in a Haystack) task as an evaluation metric to balance performance and model discriminability. The NIAH task in RULER comprises subtasks such as single-needle and multi-needle retrieval. For each sequence length interval, we calculate the average score across all subtasks.

Results. The table uses "[]" to indicate existing positional encoding schemes, with another set used as a control. After conducting experiments, we obtained the average NIAH scores from RULER across all positional encodings for models of various sizes. The comparative results are shown in Table 2.

From Table 2, it is evident that S^{3}PE$_{50\times}$ consistently achieves optimal performance across all model sizes. Examining the results for PI reveals that when model parameters are smaller, PI$_{50\times}$ performs worse after fine-tuning compared to PI$_{8\times}$; however, as model size increases, this trend improves. This observation supports Finding 2 in Section 3.2. The performance drop for PI$_{50\times}$ is likely due to excessive interpolation disrupting the model's ability to recognize minimal token distances. Yet, as model parameters increase, the model's capacity to capture positional information from finer granularity strengthens through fine-tuning, gradually restoring PI$_{50\times}$'s effectiveness.

Apart from PI, the performance of each positional encoding with $50\times$ interpolation consistently surpasses its $8\times$ counterpart, further corroborating Finding 3 in Section 3.2.

Focusing on model performance in the longest input sequence interval post-extrapolation, a comparison of the evaluation results is presented in Table 3.

The results in Table 3 supplement those in Table 2, supporting the conclusions drawn from Table 2. From Table 3, we observe that, except for PI in the smaller model (s2), all positional encodings with a scaling factor of 50x achieve better scores in the 16k-32k interval compared to their 8x counterparts. This indicates that within a certain range, increasing the interpolation factor enhances model performance in larger context windows after training.

By combining insights from Tables 2 and 3, we can draw the following conclusions:

- S^{3}PE$_{50\times}$ consistently outperforms ABF$_{50\times}$, and, apart from PI, the performance of each positional encoding with a 50x scaling factor consistently surpasses its 8x counterpart, further validating Finding 3 in Section 3.2.

- S^{3}PE$_{50\times}$ maintains an optimal position with a consistent advantage of approximately 2-3 points in overall average scores, establishing it as the most effective interpolation scheme among those evaluated.

Further details and results from our experiments will be released in future updates.

5 Conclusion

In this work, we compared several mainstream positional encoding interpolation schemes, including PI, NTK-aware, ABF, YaRN, and LongRoPE, and demonstrated that these encodings can all be expressed in a unified form. Based on this framework, we proposed guiding principles for optimal interpolation, leading to the discovery of a new interpolation scheme, S^3PE, which outperforms all the above-mentioned positional encodings. We conducted an in-depth analysis of the shape and maximum value of the interpolation factor α_j and carried out comprehensive and detailed experiments. These experiments fill a gap in the literature by providing a fine-tuning-based, thorough comparison of existing positional encodings. The results further validate that S^3PE consistently achieves the best performance among all positional encodings to date and generalizes well across different model sizes.

References

[1] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[2] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[3] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[4] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[5] Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. Extending context window of large language models via positional interpolation. *arXiv preprint arXiv:2306.15595*, 2023.

[6] Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, et al. Effective long-context scaling of foundation models. *arXiv preprint arXiv:2309.16039*, 2023.

[7] emozilla. Dynamically Scaled RoPE further increases performance of long context LLaMA with zero fine-tuning, 2023.

[8] Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. Yarn: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071*, 2023.

[9] Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024.

[10] bloc97. NTK-Aware Scaled RoPE allows LLaMA models to have extended (8k+) context size without any fine-tuning and minimal perplexity degradation, 2023.

[11] Yao Fu, Rameswar Panda, Xinyao Niu, Xiang Yue, Hannaneh Hajishirzi, Yoon Kim, and Hao Peng. Data engineering for scaling language models to 128k context. *arXiv preprint arXiv:2402.10171*, 2024.

[12] Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, et al. Minicpm: Unveiling the potential of small language models with scalable training strategies. *arXiv preprint arXiv:2404.06395*, 2024.

[13] Shengding Hu, Xin Liu, Xu Han, Xinrong Zhang, Chaoqun He, Weilin Zhao, Yankai Lin, Ning Ding, Zebin Ou, Guoyang Zeng, et al. Predicting emergent abilities with infinite resolution evaluation. In *The Twelfth International Conference on Learning Representations*, 2023.