PSC: Extending Context Window of Large Language Models via Phase Shift Calibration

Anonymous ACL submission

Abstract

Rotary Position Embedding (RoPE) is an efficient position encoding approach and is widely utilized in numerous large language models (LLMs). Recently, a lot of methods have been put forward to further expand the context window based on RoPE. The core concept of those methods is to predefine or search for a set of factors to rescale the base frequencies of RoPE. Nevertheless, it is quite a challenge for existing methods to predefine an optimal factor due to the exponential search space. In view of this, we introduce PSC (Phase Shift Calibration), a small module for calibrating the frequencies predefined by existing methods. With the employment of PSC, we demonstrate that many existing methods can be further enhanced, like PI, YaRN, and LongRoPE. We carry out extensive experiments on many models in various tasks and the results verify the effectiveness of our approach.

1 Introduction

Large-scale language models (LLMs) have shown impressive results across a variety of natural language processing (NLP) applications. For instance, OpenAI has shown that GPT-4 (OpenAI, 2023) can perform at a level comparable to humans in a range of professional tasks. Additionally, open-source models such as LLaMA2 (Touvron et al., 2023b) and Mistral (Jiang et al., 2023) have made significant contributions to the advancement and practical application of LLMs in both research and industry. However, one significant challenge that LLMs face is handling tasks that require processing long context, such as responding to questions based on multiple documents and summarizing lengthy texts such as books. In these scenarios, the perplexity of the responses can increase substantially, leading to a notable decrease in the performance of LLMs. Therefore, equipping LLMs with long-range ability has become a critical and pressing issue for both academic and commercial sectors.

An intuitive method is to fine-tune a pre-trained Transformer with a longer context length. Nevertheless, there are two limitations: first, models trained in this manner adapt to long context lengths very slowly (Chen et al., 2023b); second, fine-tuning updates all model parameters is memory-inefficient which prevents the model from adapting to a large context length (Chen et al., 2023c).

Optimizing position encodings is another major direction for extending the context window of LLMs. The original Transformer (Vaswani et al., 2017) that serves as the core component of LLMs uses sinusoidal functions of various frequencies to enhance the model's extrapolate capability. It could be regarded as an absolute position encoding mechanism. Since then, relative positional encoding techniques such as RoPE (Su et al., 2021) and ALiBi (Press et al., 2022) have further increased the length extrapolation of Transformers. Despite the effectiveness, many existing pre-trained LLMs that use these positional encoding methods exhibit weak extrapolation capabilities. For example, LLaMA (Touvron et al., 2023a) with 2048 predefined context size explodes perplexity metric when the input texts length is larger than 4096 (Chen et al., 2023b).

Recently, new positional encoding schemes have been proposed to overcome such limitations. (Chen et al., 2023b) and (kaiokendev, 2023) show that the effective context size could be extended by modifying RoPE via Position Interpolation, which has a much smaller upper bound than the extrapolated method and is more stable (Chen et al., 2023b). Neural Tangent Kernel (NTK) theory shows that it's difficult for multilayer perceptron (MLP) to learn high-frequency information in a low-dimensional domain. Therefore, NTK-based methods take the high-frequency information into account (block97, 2023b,a; emozilla, 2023). Furthermore, YaRN hypothesizes that previous methods lead to a closer embedding distribution and remedy the issue by using different interpolating

schemes at different frequencies (Peng et al., 2023). The shared characteristic of previous methods is that they utilize predefined frequency rescale factors. Some algorithms leverage optimal methods to estimate optimal frequencies directly, such as LongRoPE (Ding et al., 2024) and CLEX (Chen et al., 2023a). However, due to the exponential search space complexity, it is challenging for those methods to estimate an optimal frequency; they also need heavy searching cost, for instance, it costs LongRoPE nearly 3 days to search an optimal frequency for a 256k context window using an A100 GPU.

Existing techniques for encoding positional information are capable of addressing long-range dependencies. They rely on fixed patterns or require extensive searching within large parameter spaces. Consequently, adapter-based approaches like LoRA (Hu et al., 2022) have been adopted to enhance performance further. However, these methods still have limitations, stemming from the low rank of the adapter weights (Biderman et al., 2024) and the high-rank intrinsic of the long context tasks.

In this work, we introduce Phase Shift Calibration to assist position encoding methods to improve their long-range capabilities. The main idea is that we propose a module to calibrate the predefined frequency to approximate the optimal frequency. To this end, we first present that there is a rotary transformation between the actual frequencies and the optimal frequencies. The transformation can be represented as a block diagonal matrix. It is full-rank if the predefined frequencies are far from the optimal ones. Hence, it is challenging for low-rank adapter methods such as LoRA to learn the transformation. To remedy this issue, we introduce a calibration module into the base model, which approximates the rotary transformation matrix and helps calibrate the predefined frequencies to the ideal position. We conduct extensive experiments across different LLMs, position encoding schemes, and various long-context tasks. The results demonstrate the effectiveness of our methods.

2 Preliminaries and Related Work

Rotary Position Embedding (RoPE). Transformer models leverage positional information to exploit the order of tokens within texts. In our research, we concentrate on Rotary Position Embedding (RoPE) (Su et al., 2021) and its derivatives. RoPE acts as the positional encoding technique used across various Large Language Models (LLMs), such as the LLaMA (Touvron et al., 2023a) and the Mistral model (Jiang et al., 2023). Given a sequence of N word embeddings $\{\mathbf{x}_i\}_{i=1}^{\mathcal{N}}$, where $\mathbf{x}_i$ is a d-dimensional vector and d is the dimension of the embedding. RoPE applies a rotary transformation to each query/key embedding in a pairwise manner. Take $d = 2$ for example, RoPE converts each vector into the query vector and key vector via a transformation in a complex space:

$$\mathbf{q}_m = f_q(\mathbf{x}_m, m) = (\mathbf{W}_q\mathbf{x}_m)e^{im\theta} \quad (1)$$

$$\mathbf{k}_n = f_k(\mathbf{x}_n, n) = (\mathbf{W}_k\mathbf{x}_n)e^{in\theta} \quad (2)$$

where m, n are the position index, $i \doteq \sqrt{-1}$ is the imaginary unit. After rotary transformation, the attention scores are calculated as

$$softmax\left(\frac{\mathbf{q}_m^T\mathbf{k}_n}{\sqrt{d}}\right) \quad (3)$$

The rotary transformation introduces an $m-n$ term in the attention score:

$$\begin{aligned}\mathbf{q}_m^T\mathbf{k}_n &= Re\langle f_q(\mathbf{x}_m, m), f_k(\mathbf{x}_n, n)\rangle \\ &= (\mathbf{W}_q\mathbf{x}_m)^T \begin{pmatrix} c_1 & -c_2 \\ c_2 & c_1 \end{pmatrix} \mathbf{W}_k\mathbf{x}_n \\ &\doteq g(\mathbf{x}_m, \mathbf{x}_n, m-n), \end{aligned} \quad (4)$$

where

$$c_1 = cos(m-n)\theta, \quad c_2 = sin(m-n)\theta.$$

Hence, RoPE possesses the capability of encoding relative positional information via absolute positional encoding. For a general form where $d \geq 2$, RoPE divides the d-dimensional space into $d/2$ 2D complex sub-spaces:

$$\begin{aligned}&(\mathbf{x}_m)_1, (\mathbf{x}_m)_2, \cdots, (\mathbf{x}_m)_d \mapsto \\ &(\mathbf{x}_m)_1 + i(\mathbf{x}_m)_2, \cdots, (\mathbf{x}_m)_{d-1} + i(\mathbf{x}_m)_d \end{aligned} \quad (5)$$

In matrix form, the rotary-transformed query and key can be expressed as:

$$f_q = \mathbf{R}^d_{\Theta,m}\mathbf{W}_q\mathbf{x}_m \quad (6)$$

$$f_k = \mathbf{R}^d_{\Theta,n}\mathbf{W}_k\mathbf{x}_n \quad (7)$$

where

$$\mathbf{R}^d_{\Theta,m} = \begin{pmatrix} \mathbf{B}_{m,1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_{m,2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{m,d/2} \end{pmatrix},$$

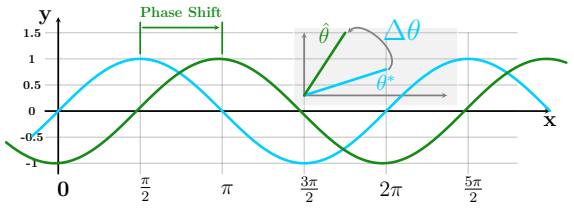

Figure 1: Phase shift leads to the $\sin$ / $\cos$ values deviating from their optimal positions. The θ^* is assumed to be an optimal frequency.

$$\mathbf{B}_{m,i} = \begin{pmatrix} \cos m\theta_i & -\sin m\theta_i \\ \cos m\theta_i & \sin m\theta_i \end{pmatrix}$$

and $\Theta = \{\theta_i = b^{-2(i-1)/d}, i \in [1, 2, \cdots, d/2]\}$ is the predefined frequencies. In many models, b is set to 10^4.

RoPE Extensions. Various RoPE-like positional encoding schemes have been proposed to enhance the capabilities of long-range dependencies. We can unify them into the following general form:

$$f_q := f_q(\mathbf{x}_m, m, h(\theta_i)) \tag{8}$$

Position Interpolation (Chen et al., 2023b) originally proposed to interpolate the position index m by modifing it into $\frac{L}{L'}m$, where L is the predefined context size and L' is the new context window beyond the pre-trained limit. Hence, $h^{PI}(\theta_i) = \frac{L}{L'}\theta_i$. The NTK-aware scheme modifies RoPE by taking into account the loss of high-frequency components through the utilization of the following formulation: $h^{NTK}(\theta_i) = \left(b \cdot s^{\frac{d}{d-2}}\right)^{-2i/d}$, where s is the scaling factor. YaRN (Peng et al., 2023) employs extrapolations in the high-frequency domain, interpolations in the low-frequency domain, and a blend of both in the intermediate frequencies. The frequency function $h^{YaRN}(\theta_i) = (1-\gamma)\frac{\theta_i}{s} + \gamma\theta_i$, where γ is the blend factor. LongRoPE (Ding et al., 2024) utilize evoluation-based search estimate optimal scale factors s_o, and the actual frequencies are scaled to $h^{LongRoPE}(\theta_i) = \frac{\theta_i}{s_o}$.

Low-rank Adaption. LoRA (Hu et al., 2022) posits that the weight adjustments in pre-trained models are characterized by a low intrinsic rank during adaptation. Given a pre-trained weight matrix $\mathbf{W} \in R^{d\times k}$, it is updated with a low-rank decomposition $\mathbf{W} + \Delta\mathbf{W} = \mathbf{W} + \mathbf{BA}$, where $\mathbf{B} \in R^{d\times r}$, $\mathbf{A} \in R^{r\times k}$, and $r \ll \min(d, k)$. During training, $\mathbf{W}$ remains fixed, while $\mathbf{A}$ and $\mathbf{B}$ are trainable.

3 Methodology

3.1 Phase Shift

Let θ^* denote the optimal frequency for long context extension of a large language model, $\hat{\theta}$ the frequency predefined or estimated by some algorithms, such as PI or LongRoPE. It is challenging to predefine a frequency $\hat{\theta}$ that is exactly equal to θ^* due to the exponential search space. The suboptimal frequencies cause the $\sin$ / $\cos$ values to move out of the ideal position, as shown in Figure 1. As a result, there exists a rotary transformation between the ideal position encoded embeddings and the actual embeddings:

$$\begin{aligned} f_q^*(\mathbf{x}_m, m) &= (\mathbf{W}_q\mathbf{x}_m)e^{im\theta^*} \\ &= (\mathbf{W}_q\mathbf{x}_m)e^{im\theta^* + im\hat{\theta} - im\hat{\theta}} \\ &= \hat{f}_q(\mathbf{x}_m, m)e^{im(\theta^* - \hat{\theta})}, \end{aligned} \tag{9}$$

$$f_k^*(\mathbf{x}_n, n) = \hat{f}_k(\mathbf{x}_n, n)e^{in(\theta^* - \hat{\theta})}, \tag{10}$$

where $f_q^*(\mathbf{x}_m, m)$ and $f_k^*(\mathbf{x}_n, n)$ are the ideal query and key with the optimal frequencies; $\hat{f}_q(\mathbf{x}_m, m)$ and $\hat{f}_k(\mathbf{x}_n, n)$ are the actual query and key with predefined frequenices.

In general form, the position-encoded query and key can be expressed as:

$$\begin{aligned} f_q^*(\mathbf{x}_m, m) &= \widetilde{\mathbf{R}}^d_{\Theta^* - \hat{\Theta}, m}\mathbf{R}^d_{\hat{\Theta}, m}\mathbf{W}_q\mathbf{x}_m \\ &= \widetilde{\mathbf{R}}^d_{\Theta^* - \hat{\Theta}, m}\hat{f}_q(\mathbf{x}_m, m), \end{aligned} \tag{11}$$

$$f_k^*(\mathbf{x}_n, n) = \widetilde{\mathbf{R}}^d_{\Theta^* - \hat{\Theta}, n}\hat{f}_q(\mathbf{x}_n, n), \tag{12}$$

where $\widetilde{\mathbf{R}}^d_{\Theta^* - \hat{\Theta}, n}$ is a block diagonal matrix with each block as

$$\begin{bmatrix} \cos n(\theta_i^* - \hat{\theta}_i) & -\sin n(\theta_i^* - \hat{\theta}_i) \\ \sin n(\theta_i^* - \hat{\theta}_i) & \cos n(\theta_i^* - \hat{\theta}_i) \end{bmatrix},$$

Θ^* denotes the optimal frequency set $\{\theta_i^*\}$, $\hat{\Theta}$ denotes actual frequency set $\{\hat{\theta}_i\}$, and $i \in [0, d/2]$. Let $\widetilde{\mathbf{W}} = \mathbf{R}^d_{\hat{\Theta}, m}\mathbf{W}_q$ and $\mathbf{I}$ denote the identity matrix, then $f_q^* = \widetilde{\mathbf{R}}\widetilde{\mathbf{W}} = \widetilde{\mathbf{W}} + (\widetilde{\mathbf{R}} - \mathbf{I})\widetilde{\mathbf{W}}$. When low-rank adapter methods such as LoRA are employed to finetune the model, we need to utilize two low-rank matrices $\mathbf{A}$ and $\mathbf{B}$ to approximate the additional matrix. Specifically, $\mathbf{BA}_{\text{LoRA}} \rightarrow (\widetilde{\mathbf{R}} - \mathbf{I})\widetilde{\mathbf{W}}$.

Approximating the matrix becomes difficult if the pre-established frequencies are not ideal. For instance, if none of the pre-established frequencies are optimal, then $\widetilde{\mathbf{R}} - \mathbf{I}$ becomes a matrix of

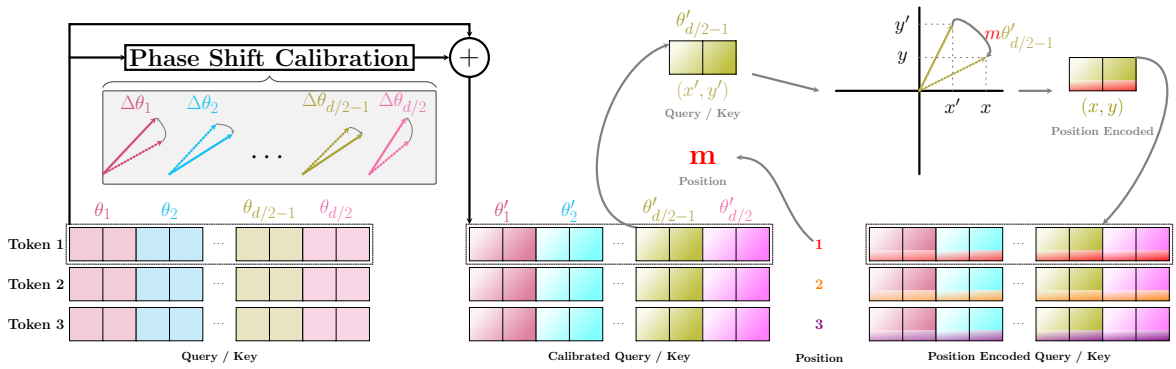

Figure 2: The embeddings are calibrated to an ideal position, and then existing position encode methods are adopted.

full rank since it is a block diagonal matrix with all non-zero elements, while $\mathbf{BA}$ remains a low-rank matrix. The accuracy of the LoRA weight approximation may be compromised due to this discrepancy in rank. Moreover, even if only a single frequency is suboptimal, the rank of $\widetilde{\mathbf{R}} - \mathbf{I}$ does not become a small number. Taking LLaMA-2 as an example, each layer of LLaMA-2 contains 32 attention heads. If there is only one suboptimal frequency, the rank of $\widetilde{\mathbf{R}} - \mathbf{I}$ could reach 32. In contrast, the LoRA method typically utilizes a low-rank matrix with a rank that does not exceed 16 in practical applications.

Beyond the matter of rank inconsistency, the diversity in the distribution of frequencies, initial phases, and the norms of the embeddings leads to a sophisticated mapping among attention layers, thereby increasing the complexity of the fine-tuning procedure.

3.2 Phase Shift Calibration (PSC)

Drawing inspiration from the ResNet (He et al., 2016) in the field of computer vision, we propose a phase shift calibration module to tackle this issue. Figure 2 demonstrates the key components of our approach. We posit that the embedding can be divided into two components: one is the base embedding, which LoRA can effectively learn; and the other is shift embedding, which should be acquired separately. This shift embedding arises from the phase shift discussed in the preceding section. To be specific, $f_q^*(\mathbf{x}_m, m) \simeq \hat{f}_q\left(\mathbf{P}(\mathbf{x}_m) \odot \mathbf{x}_m + \mathbf{x}_m, m\right)$, where $\mathbf{P}$ presents a two-layer Multilayer Perceptron (MLP) composed of a learnable block diagonal matrix and $\odot$ is the element-wise production.

In practice, since the frequencies of RoPE are organized block-wise instead of pair-wise (Wolf et al., 2020), we hence design a head-wise block diagonal matrix. More specifically:

Algorithm 1 Pseudocode of phase shift calibration in Pytorch-like style.

```
# q, k, v: queries, keys, and values;
# W_1^q, W_2^q, W_1^k, W_2^k: block diagonal matrices reshaped into shape (number heads/number key value heads, head dim, head dim);
q_t = silu(einsum('bnsd,ndr->bnsr', q, W_1^q))
p_q = 1/2 tanh(einsum('bnsr,nrd->bnsd', q_t, W_2^q))
k_t = silu(einsum('bnsd,ndr->bnsr', k, W_1^k))
p_k = 1/2 tanh(einsum('bnsr,nrd->bnsd', k_t, W_2^k))
q, k = apply_rotary_pos_emb(q+p_q*q, k+p_k*k)
out = self_attn(q, k, v)
```

$$\mathbf{P}(\mathbf{x}) = \sigma_2\left(\mathbf{W}_2\left(\sigma_1\left(\mathbf{W}_1\mathbf{x}\right)\right)\right), \qquad (13)$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are block diagonal matrices with each block size $R^{d_h \times d_h}$, and d_h is the size of single head dimension. For LLaMA and Mistral model, $d_h = 128$, our approach incorporates only a small set of parameters ($< 1\%$), therefore it is parameter efficient. σ_1 and σ_2 are activation functions, we set σ_1 = SiLU and $\sigma_2 = \frac{1}{2}$Tanh.

There could be two forms of phase shift calibration according to its position: (1) pre-calibration with the form $\hat{f}_q(\mathbf{P}(\mathbf{x}_m) \odot \mathbf{x}_m + \mathbf{x}_m, m)$ applies phase shift calibration before the position encoding module; (2) post-calibration which form is $\left(\mathbf{P}\left(\hat{f}_q(\mathbf{x}_m, m)\right) + 1\right) \odot \hat{f}_q(\mathbf{x}_m, m)$ applies phase shift calibration after the position encoding module. We will compare the two forms in the experimental section. Additionally, our approach is remarkably straightforward to implement. Algorithm 1 shows the Pytorch-like style of our method.

Extention Method	Context Window	Evaluation Context Length							
		2048	4096	6144	8192	10240	12288	14336	16384
-	4k	8.08	7.71	39.21	$> 10^2$	$> 10^2$	$> 10^2$	$> 10^2$	$> 10^3$
PI	16k	16.74	15.55	15.04	14.76	14.60	14.53	14.51	14.59
YaRN	16k	8.45	8.09	7.97	7.92	7.90	7.91	7.93	9.44
PI_{FT}	16k	8.20	7.79	7.61	7.51	7.44	7.39	7.35	7.32
$\text{PI}_{\text{FT}}^{\text{PSC}}$	16k	**8.16**	**7.76**	**7.58**	**7.48**	**7.41**	**7.36**	**7.32**	**7.28**
$\text{LongRoPE}_{\text{FT}}$	16k	8.04	7.68	7.52	7.42	7.36	7.31	7.28	7.26
$\text{LongRoPE}_{\text{FT}}^{\text{PSC}}$	16k	**8.03**	**7.67**	**7.51**	**7.41**	**7.35**	**7.30**	**7.26**	**7.24**
YaRN_{FT}	16k	8.07	7.70	7.53	7.44	7.38	7.33	7.29	7.27
$\text{YaRN}_{\text{FT}}^{\text{PSC}}$	16k	**8.05**	**7.67**	**7.51**	**7.41**	**7.35**	**7.30**	**7.26**	**7.24**

Table 1: Sliding window perplexity (S=256) of ten 96k PG19 documents over LLaMA-2 7B. The "-" means the base LLaMA2 model. $\diamondsuit_{FT}$ means the extended model is fine-tuned with LoRA (r=8). $\diamondsuit_{FT}^{PSC}$ means the extended model is fine-tuned with LoRA (r=8) and injected with the PSC module.

Extention Method	Context Window	Evaluation Context Length							
		4096	8192	12288	16384	20480	24576	28672	32768
PI_{FT}	32k	7.95	7.65	7.53	7.44	7.39	7.36	7.34	7.34
$\text{PI}_{\text{FT}}^{\text{PSC}}$	32k	**7.88**	**7.60**	**7.47**	**7.38**	**7.33**	**7.30**	**7.28**	**7.27**
YaRN_{FT}	32k	7.76	7.49	7.38	7.31	7.26	7.23	7.22	7.23
$\text{YaRN}_{\text{FT}}^{\text{PSC}}$	32k	**7.70**	**7.44**	**7.33**	**7.26**	**7.21**	**7.19**	**7.17**	**7.17**

Table 2: Sliding window perplexity (S=256) of ten 96k PG19 documents over LLaMA-2 7B (32k).

4 Experiments

We demonstrate that phase shift calibration successfully realizes the context window extension of large language models by using RoPE extensions as its position encoding schemes. Furthermore, our approach is compatible with a broad range of position encoding techniques, including search-based methods (LongRoPE), position interpolation (PI), and the combination of interpolation and extrapolation techniques (YaRN).

4.1 Experimental Settings

Model. We conduct experiments on LLaMA-2 and Mistral with various position encode approaches. In addition, we assess our approach by utilizing several well-known publicly available models, including Together.ai (Together.ai, 2023), CodeLlama (Rozière et al., 2023), and LongLoRA (Chen et al., 2023c).

Datasets. In order to comprehensively and meticulously analyze our technique, we employ several datasets to train and assess our context-extended model. We initially carry out experiments by utilizing a small dataset sampled from the RedPajama (Computer, 2023) dataset, and the length of each text in the sampled dataset is greater than 4K. We also utilize the PG19 (Rae et al., 2020) train split dataset chunked into 64k segments for training. While conducting the evaluation, we use the PG19 validation split and the Proof-pile (Azerbayev et al., 2022) test split. Details are shown in the appendix.

4.2 Evaluation

Long-sequence Language Modeling. We make a comparison of the long sequence language modeling performance using the perplexity metric. The sliding window method from (Press et al., 2022) with S=256 is adopted.

We initially present the evaluation results on the LLaMA-2 model and its context window extensions using various approaches in Table 1, Table 2, and Table 3. We extend LLaMA-2 with diverse position encoding schemes such as PI, YaRN, and LongRoPE. When fine-tuning, we employ LoRA with a rank of 8. We can notice that the fine-tuned models show lower perplexity than the non-fine-tuned ones. Phase shift calibration can enhance all the base position encoding schemes. It even boosts the performance of the optimal-based method LongRoPE. The possible reason might be that the scale factor search space is exponential, which makes it

Extention Method	Context Window	Evaluation Context Length								
		4096	8192	16384	24576	32768	40960	49152	57344	65536
PI_{FT}	64k	8.18	7.87	7.65	7.57	7.53	7.52	7.51	7.49	7.48
PI_{FT}^{PSC}	64k	**8.09**	**7.79**	**7.57**	**7.49**	**7.46**	**7.44**	**7.43**	**7.41**	**7.39**

Table 3: Sliding window perplexity (S=256) of ten 96k PG19 documents over LLaMA-2 7B (64k).

difficult to search for an ideal frequency, and the objective signal may be too sparse as only 5 PG19 texts are used to guide the search. More significantly, by comparing Table 1, Table 2, and Table 3, we can notice that the advantage of applying phase shift calibration becomes greater as the extended context window changes from 16k to 64k. The reason perhaps is that as the context window increases, the largest possible rescale factor also increases. In other words, the frequency solution spaces are enlarged, which makes it even more difficult to predefine an ideal frequency. With phase shift calibration, the frequencies are pre-calibrated to an ideal position.

We also incorporate the phase shift calibration module into several well-known publicly available models, like Together.ai, CodeLlama, and LongLoRA. We fine-tune the enhanced model using the PG19 dataset and assess it on the Proof-pile dataset. Table 4 presents the outcomes, and we can note that phase shift calibration enhances LongLora and YaRN more prominently than it does for Together and CodeLlama. This may be due to that the Together and CodeLlama are pre-trained and fine-tuned with full parameter updates, while the remaining ones utilize the LoRA-like method.

Passkey Retrieval. The passkey retrieval task proposed by (Mohtashami and Jaggi, 2023) gauges a model's effective context window size. This task aims to require a model to fetch a simple passkey from a large set of useless tokens. In our assessment, we conduct 10 iterations of the passkey retrieval task with the context window sizes ranging from 2k to 36k. The random passkey is positioned at a random location that is uniformly distributed among the collection of the tokens. The prompt template is presented in the appendix.

The comparison of retrieval accuracy with various approaches is presented in Figure 3. We can notice that the accuracy of the LLaMA-2 base model drops instantly to 0 when the sequence length goes beyond its pre-trained context window length. Although extending the context window using YaRN without fine-tuning can raise the accuracy beyond the 4k pre-trained context size, the accuracy is lower and the performance is less stable compared to the fine-tuning-based models. With fine-tuning, position encoding methods such as PI and YaRN can significantly enhance the retrieval accuracy. However, the accuracy becomes unstable as the evaluated context length gets closer to the context window size. For example, at 32k, the accuracy of LLaMA-2 7B (YaRN FT) drops to 90%, while at 33k, the accuracy of LLaMA-2 7B (PI FT) drops to 90%. Both LLaMA-2 7B (PI PSC) and LLaMA-2 7B (YaRN PSC) show a 100% retrieval accuracy up to a 34k context length when the phase shift calibration module is enabled.

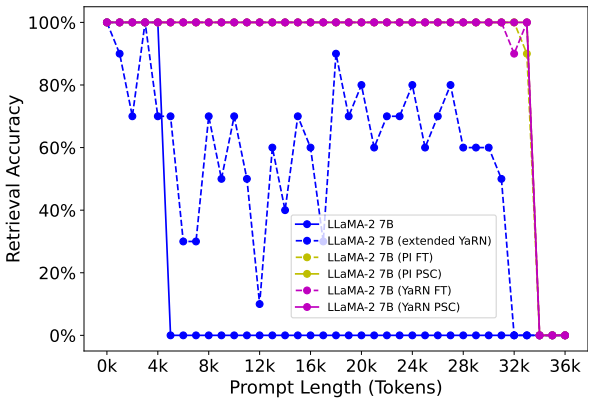

Figure 3: A comparison of passkey retrieval accuracy for context-augmented Large Language Models (LLMs). "Extend YaRN" indicates that the model incorporates YaRN without undergoing fine-tuning. "FT" denotes that the models have been fine-tuned using LoRA (r=8), while "PSC" signifies that the models have been fine-tuned with the phase shift calibration module activated. (The graphs for LLaMA-2 7B (PI PSC) and LLaMA-2 (YaRN PSC) coincide as they exhibit the same results: with 100% accuracy up to 34k.)

Standard Benchmarks We assess different methods in comparison with the original LLaMA-2 model by using the Hugging Face Open LLM Leaderboard (Face, 2023). Specifically, the Language Model Evaluation Harness library (Gao et al., 2023) is utilized to carry out the evaluation. We em-

Model Size	Model Name	Extention Method	Context Window	Evaluation Context Length				
				4096	8192	16384	32768	65536
7B	Together	PI	32k	2.47	2.31	2.19	2.11	$> 10^2$
7B	$\text{Together}_{\text{PSC}}$	PI	32k	**2.46**	**2.30**	**2.18**	**2.10**	$> 10^2$
7B	CodeLlama	NTK	100k	2.57	2.38	2.25	2.16	2.15
7B	$\text{CodeLlama}_{\text{PSC}}$	NTK	100k	2.57	2.38	**2.24**	**2.15**	**2.12**
7B	LongLoRA	PI	32k	2.57	2.38	2.25	2.16	$> 10^2$
7B	$\text{LongLoRA}_{\text{PSC}}$	PI	32k	**2.50**	**2.32**	**2.20**	**2.12**	$> 10^2$
7B	YaRN	YaRN	64k	2.50	2.34	2.23	2.14	2.08
7B	YaRN_{PSC}	YaRN	64k	**2.46**	**2.30**	**2.19**	**2.11**	**2.05**

Table 4: Sliding window perplexity (S=256) of ten 128k Proof-pile documents over various models.

Model Size	Model Name	Extention Method	Context Window	ARC-c	HellaSwag	MMLU	TruthfulQA
7B	Llama2	-	4k	52.47	**78.97**	**46.24**	38.96
7B	Together	PI	32k	47.27	77.41	45.33	38.4
7B	$\text{Together}_{\text{PSC}}$	PI	32k	47.35	77.39	45.57	37.66
7B	CodeLlama	NTK	100k	43.69	65.03	39.56	37.2
7B	$\text{CodeLlama}_{\text{PSC}}$	NTK	100k	42.75	64.81	39.77	36.31
7B	LongLora	PI	32k	50.51	76.32	37.81	37.92
7B	$\text{LongLora}_{\text{PSC}}$	PI	32k	50.60	76.82	39.39	38.71
7B	YaRN	YaRN	64k	**52.99**	78.25	42.46	38.32
7B	YaRN_{PSC}	YaRN	64k	52.30	78.11	42.12	**39.81**

Table 5: Performance of context-extended methods on the Hugging Face Open LLM benchmark suite.

ploy 25-shot ARC-Challenge (Clark et al., 2018), 10-shot HellaSwag (Zellers et al., 2019), 5-shot MMLU (Hendrycks et al., 2021), and 0-shot TruthfulQA (Lin et al., 2021).

The experiments aim to assess the degradation of model performance along with the context-extended window. We compare different models equipped with the phase shift calibration module with the relevant baselines and the original LLaMA-2 model. The results are summarized in Table 5. We can notice that models armed with the phase shift calibration show comparable performance to the related baselines. PSC can even outperform the related baselines. For instance, $\text{LongLoRA}_{\text{PSC}}$ outperforms LongLoRA on all datasets, $\text{Together}_{\text{PSC}}$ attains the second-best performance on the MMLU dataset. Even more notable, YaRN_{PSC} even achieves the best performance on the TruthfulQA dataset, with the accuracy performance increased by 0.85%.

4.3 Ablation Study

In this section, we present ablation studies on the phase shift calibration modules. We aim to address the following questions: (1) Since the phase shift calibration module introduces a few additional parameters, can a LoRA with a large rank outperform the PSC module? (2) What is the effectiveness of the phase shift calibration module at different positions of the base model? (3) What is the performance of the phase shift calibration with respect to the number of fine-tuning steps?

More Parameters. We fine-tune the base model with different ranks and position encoding methods and assess the performances. The results are presented in Table 6. Several discoveries are apparent. First, with phase shift calibration, we can obtain stable improvements across various token lengths at different ranks. Second, increasing the rank size of LoRA leads to almost no performance gain. Additionally, Table 2 shows the results of model fine-tuning with LoRA rank 8. By comparing it with Table 6, we can observe that even if the LoRA rank is doubled, the performance gains are negligible. Hence, the performance of phase shift calibration does not stem from more parameters but from calibrating the frequencies to the optimal

Rank	Extention Method	Context Window	Evaluation Context Length							
			4096	8192	12288	16384	20480	24576	28672	32768
12	YaRN_{FT}	32k	7.76	7.49	7.38	7.30	7.26	7.23	7.22	7.22
12	$\text{YaRN}_{\text{FT}}^{\text{PSC}}$	32k	**7.70**	**7.43**	**7.33**	**7.26**	**7.21**	**7.18**	**7.17**	**7.17**
16	YaRN_{FT}	32k	7.75	7.49	7.37	7.30	7.26	7.23	7.22	7.23
16	$\text{YaRN}_{\text{FT}}^{\text{PSC}}$	32k	**7.70**	**7.44**	**7.33**	**7.26**	**7.21**	**7.18**	**7.18**	**7.17**

Table 6: Sliding window perplexity (S=256) of ten 96k PG19 documents over LLaMA-2 7B.

Extention Method	Context Window	Evaluation Context Length							
		4096	8192	12288	16384	20480	24576	28672	32768
$\text{PI}_{\text{before}}^{\text{PSC}}$	32k	**8.24**	**7.93**	**7.80**	**7.72**	**7.66**	**7.63**	**7.62**	**7.62**
$\text{PI}_{\text{after}}^{\text{PSC}}$	32k	8.50	8.21	8.10	8.05	8.02	8.03	8.07	8.16

Table 7: Sliding window perplexity (S=256) of ten 96k PG19 documents over LLaMA-2 7B.

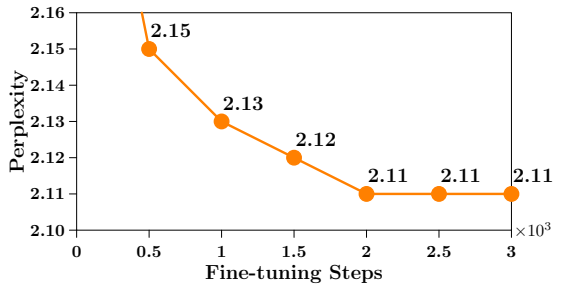

Figure 4: An ablation study on the fine-tuning process utilizing phase shift calibration. The perplexity is assessed with a context length of 32k.

states.

Pre-calibration vs Post-calibration. We evaluate the effectiveness of the phase shift calibration module at different positions. In this experiment, we only update the parameters of the PSC while keeping the other parameters frozen. The results are summarized in Table 7. We have several key findings with these results. First, by comparing it with Table 2, we can observe that the phase shift calibration itself can improve the perplexity of the models. When combined with LoRA, it can further enhance the performance. Second, applying the phase shift calibration before the position encoding method is better than applying it after the position encoding method. The possible reason is that the position encoding method introduces complex non-linear distortion to the query/key embeddings.

Ablation on Fine-tuning Steps. We present the relationship between perplexity and fine-tuning steps for the Mistral-7B model extended to a 32K context window on the Proof-pile test set. As Figure 4 indicates, the perplexity drops rapidly to 2.15 at step 500, and then gradually converges to 2.11 at step 2000. Further fine-tuning the model from step 2000 does not lead to any further improvement.

5 Conclusion

In this work, we present PSC: Phase Shift Calibration, an approach for calibrating the existing extended position encoding methods. We first present that there is a rank inconsistency issue when the predefined frequencies are not optimal. A phase shift calibration module is designed to remedy this issue. We conduct extensive experiments on various tasks, and the results show that PSC is compatible with various context extension methods, including interpolation, mixing of interpolation/extrapolation, and search-based techniques. With PSC, the long-range abilities of LLMs can be further enhanced. Moreover, our method only introduces a few more parameters ($< 1\%$), which is parameter-efficient. This work thus supports many natural language processing tasks that require long-range capabilities. We discuss several promising future works in the appendix.

6 Limitations

This paper introduces a phase shift calibration module to the base model to further enhance the performance of existing position encoding methods. Since the introduced phase shift calibration module contains a small set of trainable parameters, our method requires fine-tuning of the enhanced models and needs a bit more GPU memory than simply fine-tuning with LoRA.

References

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebr'on, and Sumit K. Sanghai. 2023. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. *ArXiv*, abs/2305.13245.

Z. Azerbayev, E. Ayers, and B. Piotrowski. 2022. Proof-pile.

Dan Biderman, Jose Gonzalez Ortiz, Jacob Portes, Mansheej Paul, Philip Greengard, Connor Jennings, Daniel King, Sam Havens, Vitaliy Chiley, Jonathan Frankle, Cody Blakeney, and John P. Cunningham. 2024. Lora learns less and forgets less. *ArXiv*, abs/2405.09673.

block97. 2023a. Add ntk-aware interpolation "by parts" correction.

block97. 2023b. Ntk-aware scaled rope allows llama models to have extended (8k+) context size without any fine-tuning and minimal perplexity degradation.

Guanzheng Chen, Xin Li, Zaiqiao Meng, Shangsong Liang, and Lidong Bing. 2023a. Clex: Continuous length extrapolation for large language models. *arXiv preprint arXiv:2310.16450*.

Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023b. Extending context window of large language models via positional interpolation. *ArXiv*, abs/2306.15595.

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2023c. Longlora: Efficient fine-tuning of long-context large language models. *arXiv:2309.12307*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *ArXiv*, abs/1803.05457.

Together Computer. 2023. Redpajama: An open source recipe to reproduce llama training dataset.

Tri Dao. 2023. FlashAttention-2: Faster attention with better parallelism and work partitioning.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*.

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. *Preprint*, arXiv:2402.13753.

emozilla. 2023. Dynamically scaled rope further increases performance of long context llama with zero fine-tuning.

Hugging Face. 2023. Open llm leaderboard.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023. A framework for few-shot language model evaluation.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '16, pages 770–778.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Xiaodong Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *ICLR*, abs/2009.03300.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

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

kaiokendev. 2023. Things i'm learning while training superhot.

Stephanie C. Lin, Jacob Hilton, and Owain Evans. 2021. Truthfulqa: Measuring how models mimic human falsehoods. In *Annual Meeting of the Association for Computational Linguistics*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

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

OpenAI. 2023. Gpt-4 technical report.

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

Ofir Press, Noah A. Smith, and Mike Lewis. 2022. Train short, test long: Attention with linear biases enables input length extrapolation.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, and Timothy P. Lillicrap. 2020. Compressive transformers for long-range sequence modelling. *ICLR*.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, I. Evtimov, Joanna Bitton, Manish P Bhatt, Cristian Cantón Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre D'efossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2023. Code llama: Open foundation models for code. *ArXiv*, abs/2308.12950.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. 2021. Roformer: Enhanced transformer with rotary position embedding.

Together.ai. 2023. Llama-2-7b-32k.

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. 2023a. Llama: Open and efficient foundation language models. *ArXiv*.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Cantón Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony S. Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel M. Kloumann, A. V. 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, R. Subramanian, Xia Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zhengxu Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288.

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

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Annual Meeting of the Association for Computational Linguistics*.

A Appendix

A.1 Settings

Training. For training, we employ the AdamW optimizer (Loshchilov and Hutter, 2019) with $\beta_1 = 0.9$ and $\beta_2 = 0.95$. We utilize a learning rate of 2×10^{-4} when training on the sampled RedPajama dataset, and 2×10^{-5} otherwise. The weight decay is set to zero, and a linear warmup of 20 steps is applied. All experiments are conducted using the Transformers (Wolf et al., 2020) framework, and Flash Attention 2 (Dao et al., 2022; Dao, 2023) is utilized to optimize memory usage. For a fair comparison, all models are trained for 3000 steps on 4 A800 GPUs. We set the batch size to the value that maximizes GPU memory utilization and adopt a gradient accumulation step size of 4. When training LongRoPE, we add three additional rescale factors corresponding to PI, NTK, and YaRN to the initial population.

Evaluation. When training our model with the RedPajama dataset, we evaluate our method by using the PG19 validation split. We pick 10 random samples from the PG19 validation split with at least 96k tokens. When we train our model on the PG19 train split dataset chunked into 64k segments, we evaluate the model using the Proof-pile (Azerbayev et al., 2022) test split. Likewise, we select 10 random samples from Proof-pile with at least 128k tokens.

Passkey prompt. To measure the effective context window size, we utilize the prompt employed by existing literature (Mohtashami and Jaggi, 2023; Chen et al., 2023c; Ding et al., 2024). The prompt is shown as follows:

Passkey prompt

There is an important info hidden inside a lot of irrelevant text. Find it and memorize them.I will quiz you about the important information there.
The grass is green. The sky is blue. The sun is yellow. Here we go. There and back again. (repeat M times)
The pass key is <PASS KEY>. Remember it. <PASS KEY> is the pass key.
The grass is green. The sky is blue. The sun is yellow. Here we go. There and back again. (repeat N times)
What is the pass key? The pass key is

The <PASS KEY> is the number to retrieve, we randomly generate a passkey in the range $[1, 50000]$ during each testing time. The text length varies with the values of M and N.

A.2 More Experiments

LLaMA-2 7B. We present the experimental results for the YaRN method in Table 8 due to space constraints. The settings are identical to those used in the experiment shown in Table 3.

Mistral 7B. We also extend the Mistral 7B v0.1 model (Jiang et al., 2023), which is another famous open-source model. We extend Mistral with YaRN (Peng et al., 2023) to 32k and perform an ablation study on the phase shift calibration module. For training, we use a small dataset sampled from the RedPajama (Computer, 2023) dataset with token length $\geq$ 4k. we utilize a constant learning rate 2×10^{-4} with a linear warmup of 20 steps. We fine-tune the models for 3000 steps. We evaluate the models using Proof-pile (Azerbayev et al., 2022) test split and 10 documents with token length $\geq$ 128k are sampled. The results are described in Table 9. We can observe that with the phase shift calibration module enabled, the performance of long-range abilities gets further improved upon YaRN.

LLaMA-2 13B. In addition, we assess our approach on the LLaMA-2 13B model (Touvron et al., 2023b). The models are fine-tuned with sampled documents from RedPajama (Computer, 2023) dataset. Each document has token length $\geq$ 4k. We set the learning rate as 2×10^{-4} and use a linear warmup of 20 steps. Both PI (Chen et al., 2023b) and YaRN (Peng et al., 2023) are employed in our evaluation. Table 10 shows the results. The results exhibit similar performance improvement as the evaluations on the LLaMA-2 7B model. It demonstrates our method is compatible with various LLMs and position encoding approaches.

A.3 Complexity

Phase shift calibration defines block diagonal matrices for query/key embeddings. Each block is a $d_h \times d_h$ matrix, where d_h is the dimension of a single head. As a result, it introduces additional 64M parameters for LLaMA-2 7B, accounting for 0.95% ($< 1\%$) of the total parameters. Since Mistral adopts GQA (Ainslie et al., 2023), the PSC introduces 40M parameters for Mistral 7B, account

Extention Method	Context Window	Evaluation Context Length								
		4096	8192	16384	24576	32768	40960	49152	57344	65536
YaRN_{FT}	64k	7.85	7.59	7.41	7.34	7.32	7.32	7.30	7.29	7.32
$\text{YaRN}_{\text{FT}}^{\text{PSC}}$	64k	**7.75**	**7.49**	**7.31**	**7.25**	**7.23**	**7.22**	**7.21**	**7.19**	**7.19**

Table 8: Sliding window perplexity (S=256) of ten 96k PG19 documents over LLaMA-2 7B (64k). $\diamondsuit_{FT}$ means the extended model is fine-tuned with LoRA (r=8). $\diamondsuit_{FT}^{PSC}$ means the extended model is fine-tuned with LoRA (r=8) and injected with the PSC module.

Extention Method	Context Window	Evaluation Context Length							
		4096	8192	12288	16384	20480	24576	28672	32768
-	8k	2.23	2.09	4.60	26.25	79.97	$> 10^2$	$> 10^2$	$> 10^2$
YaRN_{FT}	32k	2.42	2.25	2.20	2.17	2.15	2.14	2.13	2.13
$\text{YaRN}_{\text{FT}}^{\text{PSC}}$	32k	**2.40**	**2.23**	**2.17**	**2.15**	**2.14**	**2.13**	**2.12**	**2.11**

Table 9: Sliding window perplexity (S=256) of ten 128k Proof-pile documents over Mistral 7B. The "-" means the base Mistral 7B v0.1 model. $\diamondsuit_{FT}$ means the extended model is fine-tuned with LoRA (r=8). $\diamondsuit_{FT}^{PSC}$ means the extended model is fine-tuned with LoRA (r=8) and injected with the PSC module.

Extention Method	Context Window	Evaluation Context Length							
		2048	4096	6144	8192	10240	12288	14336	16384
-	4k	7.25	6.91	48.98	$> 10^2$	$> 10^3$	$> 10^3$	$> 10^3$	$> 10^3$
PI	16k	12.42	11.65	11.30	11.11	10.96	10.87	10.81	10.80
YaRN	16k	7.54	7.22	7.10	7.06	7.03	7.03	7.04	8.14
PI_{FT}	16k	7.35	6.99	6.84	6.75	6.68	6.64	6.60	6.57
$\text{PI}_{\text{FT}}^{\text{PSC}}$	16k	**7.32**	**6.97**	**6.82**	**6.73**	**6.67**	**6.62**	**6.58**	**6.55**
YaRN_{FT}	16k	7.26	6.92	6.76	6.68	6.62	6.57	6.54	6.52
$\text{YaRN}_{\text{FT}}^{\text{PSC}}$	16k	**7.23**	**6.89**	**6.74**	**6.65**	**6.60**	**6.55**	**6.52**	**6.49**

Table 10: Sliding window perplexity (S=256) of ten 96k PG19 documents over LLaMA-2 13B. The "-" means the base LLaMA2 model. $\diamondsuit_{FT}$ means the extended model is fine-tuned with LoRA (r=8). $\diamondsuit_{FT}^{PSC}$ means the extended model is fine-tuned with LoRA (r=8) and injected with the PSC module.

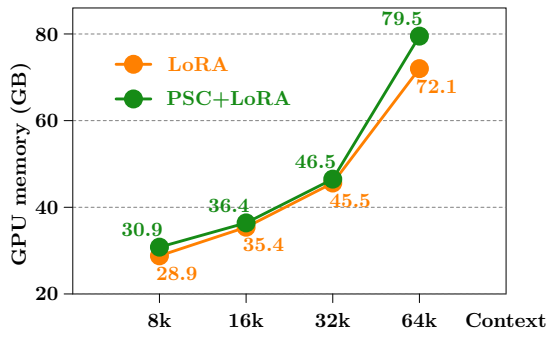

Figure 5: GPU memory consumption by LoRA and PSC.

for 0.6% ($< 1\%$) of the total parameters. Figure 5 shows the GPU memory used by LoRA and PSC.

A.4 Initial Phase and Norm distribution

The RoPE and its extensions consider each pair (x, y) in the embeddings as a complex number. And perform a rotary transformation on each pair. Due the complicated distribution of (x, y), it is challenging to predefine a set of frequencies to conduct the rotary transforms. We show the initial phase and norm distributions of some sampled (x, y) pairs from different layers and heads in Figure 6 , Figure 7, Figure 8, Figure 9, Figure 10 and Figure 11, which have complicated distributions of phase and norm.

A.5 Future Work

Our method shows consistent improvements upon various position encoding methods. For future work, we would investigate PSC applications where long-range capabilities are needed, such as long-cycle conversations and LLM-based long-

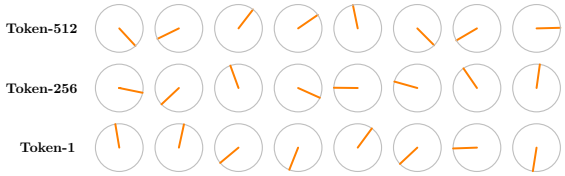

Figure 6: The phase of the first eight (x, y) pairs from 3 sampled tokens in layer 6 and head 2 of the LLaMA-2 7B.

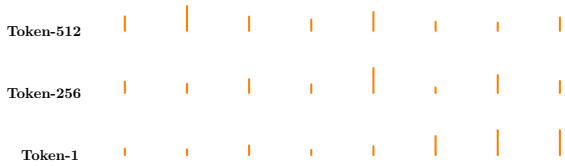

Figure 7: The norm of the first eight (x, y) pairs from 3 sampled tokens in layer 6 and head 2 of the LLaMA-2 7B.

term user historical behavior understanding. We would also try to seek phase shift calibration methods that without the need for fine-tuning.

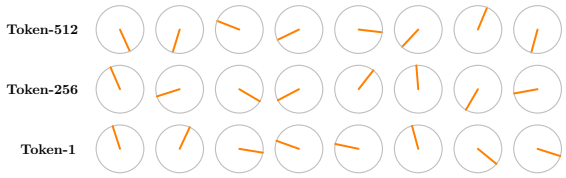

Figure 8: The phase of the first eight (x, y) pairs from 3 sampled tokens in layer 6 and head 28 of the LLaMA-2 7B.

Figure 9: The norm of the first eight (x, y) pairs from 3 sampled tokens in layer 6 and head 28 of the LLaMA-2 7B.

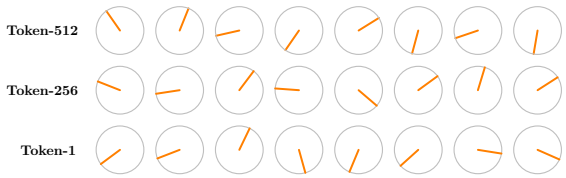

Figure 10: The phase of the first eight (x, y) pairs from 3 sampled tokens in layer 16 and head 28 of the LLaMA-2 7B.

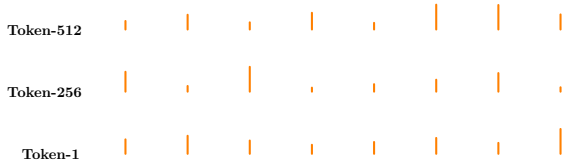

Figure 11: The norm of the first eight (x, y) pairs from 3 sampled tokens in layer 16 and head 28 of the LLaMA-2 7B.