

Resonance RoPE: Improving Context Length Generalization of Large Language Models

Suyuchen Wang^{1,2}, Ivan Kobyzhev³, Peng Lu¹, Mehdi Rezagholizadeh³ and Bang Liu^{1,2†}

¹DIRO, Université de Montréal ²Mila - Quebec AI Institute ³Huawei Noah's Ark Lab

{suyuchen.wang, peng.lu, bang.liu}@umontreal.ca

{ivan.kobyzhev, mehdi.rezagholizadeh}@huawei.com

Abstract

This paper addresses the challenge of train-short-test-long (TSTL) scenarios in Large Language Models (LLMs) equipped with Rotary Position Embedding (RoPE), where models pre-trained on shorter sequences face difficulty with out-of-distribution (OOD) token positions in longer sequences. We introduce RESONANCE ROPE, a novel approach designed to narrow the generalization gap in TSTL scenarios by refining the interpolation of RoPE features for OOD positions, significantly improving the model performance without additional online computational costs. Furthermore, we present POSGEN, a new synthetic benchmark specifically designed for fine-grained behavior analysis in TSTL scenarios, aiming to isolate the constantly increasing difficulty of token generation on long contexts from the challenges of recognizing new token positions. Our experiments on synthetic tasks show that after applying RESONANCE ROPE, Transformers recognize OOD position better and more robustly. Our extensive LLM experiments also show superior performance after applying RESONANCE ROPE to the current state-of-the-art RoPE scaling method, YaRN, on both upstream language modeling tasks and a variety of downstream long-text applications.¹

1 Introduction

Recent advancements in Large Language Models (LLMs) have demonstrated their potential across a wide spectrum of natural language processing tasks, showcasing their ability to handle complex interactions, document analyses, professional writing, and advanced reasoning with a unified approach (OpenAI, 2023; Touvron et al., 2023a,b; Jiang et al., 2024). As these models are increasingly adapted for complex applications, challenges arise in scenarios requiring the comprehension or generation

of long texts. Specifically, the train-short-test-long (TSTL) scenario (Press et al., 2022) highlights a limitation where LLMs, pre-trained on shorter sequences, struggle with out-of-distribution (OOD) token positions in longer sequences, impacting their performance in real-world applications (Zhao et al., 2023).

Recent efforts to enhance TSTL performance have focused on LLMs equipped with Rotary Position Embedding (RoPE) (Su et al., 2024), such as LLaMA (Touvron et al., 2023a,b) and Mistral (Jiang et al., 2023), owing to their exceptional capabilities and widespread adoption. These initiatives aim to refine the test-time computation of RoPE position embedding by introducing a scaling factor to either the position index of each token (Chen et al., 2023) or RoPE's base value (Xiong et al., 2023; Liu et al., 2024; Peng et al., 2024). These methods ensure that the position embeddings for out-of-distribution (OOD) positions remain within the range experienced during pre-training. This minimizes the need for the model to adapt to new position embedding value ranges, a task that is inherently difficult.

In this paper, we introduce RESONANCE ROPE, a novel technique designed to further narrow the generalization gap on position embeddings in TSTL scenarios. Recognizing that RoPE's position embedding is governed by a complex, non-linear function, we posit that minimizing extrapolation on OOD positions, while crucial, is insufficient. We argue that *it is equally vital to address the interpolation of RoPE features at the OOD positions*. By implementing RESONANCE ROPE, we slightly scale each RoPE feature to correspond to an integer wavelength. This adjustment aligns each RoPE feature's wavelength with a specific token span length, enabling it to "resonate" with a particular local context length. This simple modification effectively reduces the generalization gap for over half of the position embedding features in LLaMA

¹https://github.com/sheryc/resonance_rope.

[†]Canada CIFAR AI Chair. Corresponding author.

and LLaMA2 under TSTL scenarios. Furthermore, our approach is compatible with RoPE and any RoPE-based scaling techniques, enhancing their performance in TSTL situations without the need for additional computational resources during training or inference.

Additionally, to facilitate further research on position embeddings, we present a new synthetic benchmark tailored for TSTL scenarios, named POSGEN. Improving position embeddings for TSTL requires a detailed analysis of the cause of failures in handling longer contexts. However, current benchmarks, such as those measuring perplexity in long context (Rae et al., 2020; Huang et al., 2021; Wu et al., 2022) and most synthetic TSTL tasks (Liu et al., 2023; Kazemnejad et al., 2023) face a common issue: the difficulty of generating the next token increases with context length. This makes it difficult to determine whether a model’s failure is due to its inability to generate more complex tokens or its failure to recognize out-of-distribution (OOD) positions. POSGEN addresses this limitation by standardizing the difficulty level of token generation across all positions. This ensures that any observed shortcomings are directly related to the model’s inability to identify and handle new token positions effectively.

Our contributions in this study are threefold:

1. We propose RESONANCE ROPE, an innovative modification to RoPE based on an in-depth analysis of the wavelengths of RoPE features, aiming to narrow the generalization gap in TSTL scenarios across RoPE and similar RoPE-based scaling techniques, without necessitating extra computational resources during runtime.
2. We present POSGEN, a newly developed synthetic benchmark tailored for TSTL scenarios. This benchmark is specifically designed to disentangle the complexities associated with generating tokens in longer contexts from the challenges posed by recognizing new positions or position embedding values.
3. Through rigorous testing of RESONANCE ROPE on both RoPE and YaRN within the POSGEN benchmark, we demonstrate its ability to enhance performance on out-of-distribution (OOD) positions, surpassing existing methods that do not include RESONANCE ROPE. Moreover, when applied to

YaRN, RESONANCE ROPE further improves LLM’s length extrapolation ability, as evidenced by lower perplexity in upstream TSTL language modeling and enhanced outcomes in downstream tasks involving lengthy contexts.

2 Related Work

2.1 Scaling of RoPE Position Encoding

Recent efforts in extending LLMs’ context window focus on manipulating position embedding (PE), particularly RoPE (Su et al., 2024), which is used in LLMs like LLaMA (Touvron et al., 2023a,b) and Mistral (Jiang et al., 2023). Main strategies include embedding scaling (Chen et al., 2023; Liu et al., 2024; Peng et al., 2024) and randomizing token positions (Ruoss et al., 2023; Zhu et al., 2024). Our emphasis is on the embedding scaling strategies.

Existing embedding scaling strategies adjust position embedding for longer sequences to match the pre-training range, avoiding feature extrapolation. For instance, Chen et al. (2023) compresses position indices to fit the pre-training range, extending LLaMA’s (Touvron et al., 2023a) context to 16K with 1,000 steps of fine-tuning. Alternatively, Liu et al. (2024); Rozière et al. (2023); Xiong et al. (2023) modify RoPE’s rotary base and employ fine-tuning on extended sequences, termed Adjusted Base Frequency (ABF) or “NTK-aware” scaling. Code LLaMA (Rozière et al., 2023) achieved 16K context length with this method after 10,000 fine-tuning steps. YaRN (Peng et al., 2024) improved NTK-aware scaling by segmenting RoPE features and applying tailored extrapolation strategies, achieving 64K context length for LLaMA2 (Touvron et al., 2023b) with 400 fine-tuning steps. Distinguishingly, our RESONANCE ROPE focus on reducing feature interpolation on OOD positions, which we argue is another important factor in improving the length extrapolation capability of Transformer.

2.2 Long Context Evaluations

Evaluations of Transformer-based LLMs’ long-context capabilities are twofold: synthetic task assessments for length extrapolation strategies and real-world task evaluations at the LLM scale. Synthetic evaluations target simple tasks such as long sequence classification (Tay et al., 2021) and arithmetic language modeling (Liu et al., 2023; Kazemnejad et al., 2023). LLM scale evaluations measure metrics such as perplexity (PPL) in extensive

text corpora (e.g., PG19 (Rae et al., 2020), GovReport (Huang et al., 2021), GitHub (Wu et al., 2022) and complex tasks including summarization, question answering, and mathematical reasoning (An et al., 2023; Bai et al., 2023; Shaham et al., 2023).

3 Background

3.1 Rotary Position Embedding (RoPE)

In Transformers (Vaswani et al., 2017), the self-attention scores are softmax-normalized scaled attention logits $\mathbf{q}^\top \mathbf{k}$:

$$a_{m,n} = \text{Softmax} \left(\frac{\mathbf{q}_m^\top \mathbf{k}_n}{\sqrt{d}} \right)$$

Suppose the input to a single attention head is $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_l \in \mathbb{R}^d$, where l is the sequence length and d is the dimension of an attention head. RoPE injects the position information of each token into the \mathbf{q} and \mathbf{k} vectors by the following equations in the complex space:

$$\begin{aligned} \mathbf{q}_{m,[2j:2j+1]} &= \mathbf{W}_q \mathbf{x}_m e^{im\theta_j} \\ \mathbf{k}_{m,[2j:2j+1]} &= \mathbf{W}_k \mathbf{x}_m e^{im\theta_j} \\ \theta_j &= b^{\frac{-2j}{d}}, \end{aligned} \quad (1)$$

where $\mathbf{W}_q, \mathbf{W}_k$ are trainable parameters, and b is a constant called the rotary base, which is set to 10,000 (Su et al., 2024) or other integers or fractions (Xiong et al., 2023; Peng et al., 2024). This form makes the dot product between the m -th query \mathbf{q}_m and n -th key \mathbf{k}_n only depend on the input $\mathbf{x}_m, \mathbf{x}_n$ and their relative distance ($m - n$):

$$\begin{aligned} &\langle \mathbf{q}_{m,[2j:2j+1]}, \mathbf{k}_{n,[2j:2j+1]} \rangle \\ &= \Re \left[\mathbf{q}_{m,[2j:2j+1]}^* \mathbf{k}_{n,[2j:2j+1]} \right] \\ &= \Re \left[(\mathbf{W}_q \mathbf{x}_m)^* (\mathbf{W}_k \mathbf{x}_n) e^{i(m-n)\theta_j} \right] \\ &= g(\mathbf{x}_m, \mathbf{x}_n, m - n). \end{aligned}$$

RoPE’s real-number implementation divides the d -dimension space into multiple 2-dimensional subspaces and applies real rotation matrix to each of them. Formally, define a $d \times d$ block-diagonal matrix:

$$\mathbf{R}_{\Theta,m}^d = \begin{pmatrix} \mathbf{R}_{\theta_0,m} & \cdots & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{\theta_1,m} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{R}_{\theta_{\frac{d}{2}-1},m} \end{pmatrix}, \quad (2)$$

where $\Theta = \{\theta_0, \theta_1, \dots, \theta_{\frac{d}{2}-1}\}$, and each $\mathbf{R}_{\theta_j,m}$ is a 2×2 rotation matrix:

$$\mathbf{R}_{\theta_j,m} = \begin{pmatrix} \cos m\theta_j & -\sin m\theta_j \\ \sin m\theta_j & \cos m\theta_j \end{pmatrix}. \quad (3)$$

RoPE computes the attention logit $\mathbf{q}^\top \mathbf{k}$ as follows:

$$\mathbf{q}_m = \mathbf{R}_{\Theta,m}^d \mathbf{W}_q \mathbf{x}_m \quad (4)$$

$$\mathbf{k}_n = \mathbf{R}_{\Theta,n}^d \mathbf{W}_k \mathbf{x}_n \quad (5)$$

$$\mathbf{q}_m^\top \mathbf{k}_n = \mathbf{x}_m^\top \mathbf{W}_q \mathbf{R}_{\Theta,n-m}^d \mathbf{W}_k \mathbf{x}_n \quad (6)$$

For each two dimensions $[2j : 2j + 1]$ of \mathbf{q} and \mathbf{k} , its corresponding θ_j reflects a temporal wavelength λ_j . This wavelength describes the token length for the corresponding RoPE features to encounter approximately the same rotary angle $m\theta_j$ in Equation 3:

$$\lambda_j = \frac{2\pi}{\theta_j} = 2\pi b^{\frac{2j}{d}} \quad (7)$$

As an example, the wavelengths of LLaMA / LLaMA2’s RoPE features range from $2\pi \approx 6.28$ for θ_0 to $2 * 10000^{126/128}\pi \approx 54410.14$ for $\theta_{\frac{d}{2}-1}$.

3.2 Critical Dimensions of RoPE

In a TSTL scenario (Press et al., 2022), one takes a model trained on texts with lengths up to L , and tests it on a task with input lengths up to $L' = sL$, with the scaling factor $s > 1$. Recently, Liu et al. (2024) discovered that there may exist two “critical dimensions” in RoPE features, which correspond to the dimensions $[2c : 2c + 1]$ that satisfies $\lambda_c \geq L$ and $\lambda_{c-1} < L$. The dimensions of RoPE features above and below the critical dimension (which we denote as “post-critical dimensions” and “pre-critical dimensions”, respectively) have different behaviors in TSTL: for post-critical dimensions (i.e., $j > c$), since their wavelengths satisfy $\lambda_j > L$, the training corpus does not cover all possible rotary angles $m\theta_j$ on a unit circle. Thus, these dimensions will encounter OOD value range on longer sequences. This is not an issue for pre-critical dimensions due to their shorter temporal wavelengths.

The concept of RoPE’s critical dimensions implicitly guides the development of RoPE scaling methods. For example, previous RoPE scaling methods (Chen et al., 2023; Xiong et al., 2023; Peng et al., 2024) mainly focus on reducing or avoiding value extrapolation on post-critical dimensions, and minimize post-training modifications to the pre-critical dimensions.

3.3 Yet another RoPE extensioN (YaRN)

YaRN (Peng et al., 2024) is the current state-of-the-art RoPE scaling method for TSTL. It introduces the “NTK-by-parts” scaling for RoPE, which applies different scaling strategies to each RoPE feature according to its temporal wavelength.

In a TSTL scenario with scaling factor s , YaRN scales the wavelength of the j -th RoPE feature λ_j to $\hat{\lambda}_j$ and further fine-tune the model:

$$\hat{\lambda}_j = (1 - \gamma_j)s\lambda_j + \gamma_j\lambda_j,$$

where γ_j is a piece-wise function depending on its corresponding wavelength λ_j , and two hyperparameters α and β :

$$\gamma_j = \begin{cases} 1, & \text{if } \lambda_j < L/\beta \\ 0, & \text{if } \lambda_j > L/\alpha \\ \frac{L/\lambda_j - \alpha}{\beta - \alpha}, & \text{otherwise} \end{cases}$$

Empirically, for the LLaMA family, Peng et al. (2024) suggests using $\alpha = 1$ and $\beta = 32$. This setting avoids value range extrapolation on post-critical dimensions, while reducing modifications to the original pre-critical dimensions.

In addition to the “NTK-by-parts” RoPE scaling strategy mentioned above, YaRN also comprises a scaling strategy on the attention scores, which reduces the change in the entropy of the attention score on longer sequences. We maintain the complete design of YaRN in our experiments, but our analysis will focus on its RoPE scaling strategy.

4 Proposed Method: RESONANCE ROPE

In this section, we introduce RESONANCE ROPE, a universal improvement for RoPE and RoPE-based scaling methods to (further) improve their length extrapolation performance.

Suppose we abstract RoPE’s Equation 4, 5: for any $\mathbf{x} \in \mathbb{R}^d$, we define $f(\mathbf{x}, m) = \mathbf{R}_{\Theta, m}^d \mathbf{W} \mathbf{x}$. In a TSTL scenario where we generalize an LLM from length L to length L' , let us denote a scaled RoPE function by \tilde{f} . To perform well on OOD positions it should reduce the *feature gap* $h(\tilde{f})$ between token features seen during training and token features after scaling that we can define for each i -th feature as:

$$h_i(\tilde{f}) = \max_{\mathbf{x} \in \mathbb{X}} \min_{\substack{m \in \{0, \dots, L-1\} \\ n \in \{L, \dots, L'-1\}}} |\tilde{f}(\mathbf{x}, m)_i - \tilde{f}(\mathbf{x}, n)_i|, \quad (8)$$

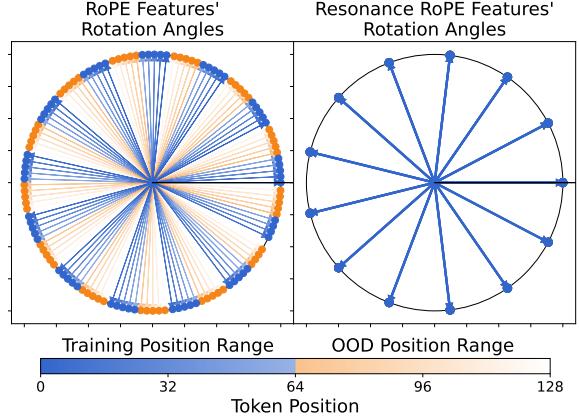


Figure 1: An illustration of RoPE’s rotation angles $m\theta_6$ and RESONANCE ROPE’s rotation angles $m\theta_6$ in Eqn. 3 in a TSTL scenario with training max length 64 and testing max length 128. RoPE’s non-integer feature wavelengths create a feature gap between the RoPE features of the training and OOD testing positions, while RESONANCE ROPE reduces this gap to 0.

where $i = 0, \dots, d-1$ and $\mathbb{X} \subset \mathbb{R}^d$ is the set of feature vectors to which we apply a position embedding. Note that the formulation of the feature gap is similar to the “embedded vector distance” metric proposed by Xiong et al. (2023). However, these two metrics target totally different aspects of RoPE scaling methods. A more detailed comparison can be found in Appendix B.

Existing RoPE scaling methods (Xiong et al., 2023; Peng et al., 2024) mainly focus on the post-critical dimensions of RoPE, since the rotary angle $m\theta_j$ on these dimensions extrapolates on OOD positions, hence creating a feature gap. In this section, we argue that reducing RoPE’s feature interpolation on the pre-critical dimensions is also beneficial for better length extrapolation.

Due to a non-linear relationship between RoPE feature \mathbf{R}_m^Θ and the token position m in Equation 3, the interpolation on RoPE features is potentially hard for the model to generalize to. We found that such potentially hard interpolation appears on the pre-critical dimensions $[0 : 2c-1]$, which have wavelengths λ_j shorter than the pre-trained sequence length L . By default, the rotary base b of RoPE features is an integer or a fraction, which makes their wavelength $\lambda_j = 2\pi b^{\frac{2j}{d}}$ not an integer. As the position index $m \in \mathbb{N}$ increases, a phase shift of $\Delta\phi$ occurs for the rotary angle $m\theta_j$ after each full rotation. This could potentially result in a large distribution gap between the RoPE features on positions seen during training and the OOD positions. This phenomenon is illustrated in Figure 1.

Algorithm 1 Pseudocode of RESONANCE ROPE.

Require: $\theta_0, \theta_1, \dots, \theta_{\frac{d}{2}-1} \in \Theta$

for $i \in \{0, 1, \dots, \frac{d}{2}-1\}$ **do**

$\lambda_i = 2\pi/\theta_i$

$\tilde{\lambda}_i = \text{round}(\lambda_i)$ \triangleright Round to integer wavelength

$\tilde{\theta}_i = 2\pi/\tilde{\lambda}_i$

end for

$\tilde{\Theta} = \{\tilde{\theta}_0, \tilde{\theta}_1, \dots, \tilde{\theta}_{\frac{d}{2}-1}\}$

Compute $\mathbf{R}_{\tilde{\Theta}}^d$ by Equation 2

Compute \mathbf{q}, \mathbf{k} by Equation 4, 5

We tackle this issue by developing a synergistic modification to the conventional RoPE embedding, referred to as RESONANCE ROPE. It aims to identify the optimal angular frequency that minimizes the interpolation gap, which ensures the corresponding wavelength closely matches the original one while imposing alignment of the wavelength to an integer. More specifically, for a given angular frequency set of RoPE $\Theta = \{\theta_1, \theta_2, \dots, \theta_{d/2}\}$, we round their wavelengths to their nearest integer to eliminate new rotary angles on each feature. We provide a pseudocode for RESONANCE ROPE in Algorithm 1.

After applying this technique, each RoPE feature repeats after $\tilde{\lambda}_i$ tokens, and therefore “resonates” with a specific span length and eliminates the interpolation gap between pre-trained and OOD positions on pre-critical dimensions. We illustrate the effect of RESONANCE ROPE on RoPE’s feature gap on one of the pre-critical dimensions in Figure 1. Moreover, we can prove the feature gap reducing ability of our method. As for above, we formalize RESONANCE ROPE’s computation rule as $\tilde{f}(\mathbf{x}, m) = \mathbf{R}_{\Theta, m}^d \mathbf{Wx}$.

Theorem 1. *For a RoPE-equipped model with context window L , RESONANCE ROPE \tilde{f} reduces the feature gap on pre-critical dimensions to 0. Specifically, $\forall \mathbf{x} \in \mathbb{X}, \forall n \in \mathbb{N} \setminus \{0, \dots, L-1\}$, we have:*

$$\min_{m \in \{0, \dots, L-1\}} |\tilde{f}(\mathbf{x}, m)_i - \tilde{f}(\mathbf{x}, n)_i| = 0$$

for all $i = 0, \dots, 2c-1$.

See the proof in Appendix A. Note that although each pre-critical RoPE feature $\mathbf{R}_{\tilde{\theta}_j, m}$ repeats, the combination of all $\{\mathbf{R}_{\tilde{\theta}_j, m}\}_{j < c}$ only repeats after the least common multiple (LCM) of all pre-critical dimensions’ wavelengths. For LLaMA2, this LCM value is greater than 7×10^{51} .

Because of its simplicity, RESONANCE ROPE can be applied on top of RoPE and all RoPE-based scaling methods to reduce their feature gap in TSTL and further improve their performance. Meanwhile, this method only involves an offline computation of the scaled θ , thus introducing no online computation overhead.

5 Evaluating Position Embeddings with POSGEN

In this section, we propose our new position embedding evaluation suite: POSGEN, based on an analysis of common failure patterns on existing position embedding evaluation methods.

We consider a next token prediction task, where we expect the model to generate the token x_l given the input sequence $\{x_0, \dots, x_{l-1}\}$. In TSTL scenarios, when a model succeeds in correctly generating a token up to position L but fails systematically afterwards, we observe two failure patterns:

- **Failure due to harder algorithmic difficulty on generating later tokens.** The rule of generating a new token x_l may vary with the sequence length l . Generally, tokens placed later in the sequence depend on more context tokens, which incurs a more complex dependency pattern. During training on shorter sequences, the model only learns the token dependency rules involving up to L tokens, and might fail on longer sequences because it has never been exposed to the more complex dependency rules.
- **Failure due to unrecognized new token positions.** The difference between training and testing lengths in the TSTL setting creates a feature gap between the position indices or position embeddings in training and inference. This feature gap makes it difficult for the model to generalize to new positions due to unrecognized features. RoPE scaling methods mainly focus on reducing this type of length extrapolation failure.

Currently, neither perplexity-based evaluations (Rae et al., 2020; Huang et al., 2021; Wu et al., 2022) nor synthetic TSTL evaluations (Kazemnejad et al., 2023; Liu et al., 2023) can effectively distinguish these two failure patterns, since the token generation difficulty tends to increase with respect to the sequence length in these tasks. To facilitate research on better position representations,

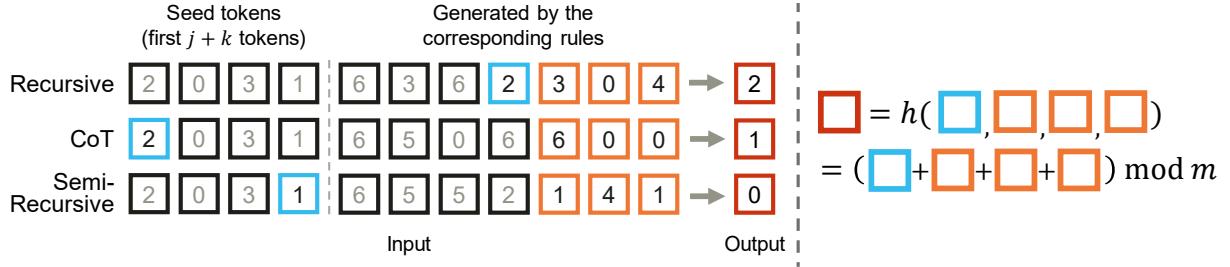


Figure 2: An example of the three subtasks of POSGEN. This figure shows the process of generating the 12th token shown in the red boxes for each subtask. In this example, h is a modular addition task with the modulus $m = 7$ and the difficulty-controlling parameters $j = 1, k = 3$. The output token depends on: (1) only the local $j + k$ tokens in the recursive task; (2) k local tokens and the beginning j tokens in the CoT task; and (3) k local tokens and j tokens with a varied dependency distance in the semi-recursive task.

we design POSGEN, which controls the difficulty in generating tokens throughout the sequence to be identical, which effectively distinguishes the two types of TSTL failures. Failures in this benchmark are only due to the inability to recognize new token positions in TSTL scenarios.

Our POSGEN framework comprises three sub-tasks, with each extracting the general token dependency pattern of a different type of reasoning task. Suppose that we define a fixed function $h : \mathbb{V}^{j+k} \rightarrow \mathbb{V}$, where \mathbb{V} is the model’s vocabulary and j, k are predefined constants controlling the task’s difficulty. The three subtasks of POS-GEN are as follows:

- Recursive.** This task simulates the token dependency pattern of generating a Fibonacci-style sequence, where new tokens depend on $j + k$ neighboring tokens only: $x_l = h(x_{l-(j+k)}, \dots, x_{l-1})$ when $l \geq j + k$.
 - Chain-of-Thought (CoT).** This task simulates the token dependency pattern of CoT reasoning (Wei et al., 2022), where new tokens depend on k neighboring tokens (simulating the previous reasoning step) and j tokens in the front (simulating the original question): $x_l = h(x_0, \dots, x_{j-1}, x_{l-k}, \dots, x_{l-1})$ when $l \geq j + k$.
 - Semi-recursive.** This task simulates the token dependency pattern of the last-letter concatenation task (Zhou et al., 2023), where new tokens depend on both k neighboring tokens (simulating the current progress) and j tokens with varied distances according to a specific rule (simulating the word sequence): $x_l = h(x_{\lfloor l-(j+k)/2 \rfloor - j}, \dots, x_{\lfloor l-(j+k)/2 \rfloor - 1}, x_{l-k}, \dots, x_{l-1})$ when $l \geq j + k$.

Based on the equation for each subtask, when given the first $j + k$ tokens, one can generate a sequence with unlimited length as the ground truth sequence. We show an example of POSGEN in Figure 2. As a TSTL benchmark, we train a model on a subtask with sequence length up to L , and evaluate the model’s accuracy on a longer sequence with length $L' > L$ generated by the same rule on the unseen positions $L < m \leq L'$, which we refer to as the “OOD Accuracy” (OOD Acc). This metric measures how well a model can recognize the OOD positions and continue following the generation rule learned during training. As a benchmark for position embeddings, a standard usage of this benchmark is to train a small Transformer (e.g., a 2-layer Transformer as used in our experiments) with different position embeddings on its training set with only short sequences, and test its OOD Accuracy on the test set with longer sequences. We provide our experiment setting for POSGEN in more details in Section 6.1.1 and Appendix C.1.

6 Experiments

We evaluate RESONANCE ROPE on three different TSTL tasks: a small-scale evaluation on our proposed POSGEN task, and LLM-scale evaluations with LLaMA2-Chat (Touvron et al., 2023b) on both language modeling perplexity and real-world long context applications.

6.1 Synthetic Task Evaluation

6.1.1 Experiment Setup

We first apply RESONANCE ROPE on RoPE and YaRN, assessing the model’s performance on POS-GEN for unseen position recognition. We test on a modular addition task, which was proved to be learnable by a one-layer Transformer (Nanda et al., 2023). We configured $j = 1$, $k = 3$, and defined

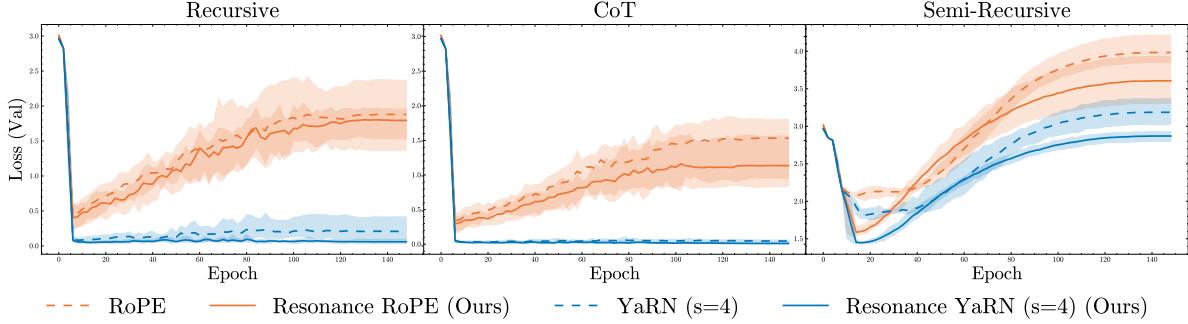


Figure 3: The validation loss curves of Transformers using RoPE and YaRN PEs with and without our RESONANCE scaling on the three subtasks of POSGEN.

| Setting | Recursive | CoT | Semi-Rec. |
|------------------|-------------------|-------------------|-------------------|
| RoPE | 65.29±0.43 | 69.56±0.33 | 17.96±0.03 |
| Res. RoPE (Ours) | 62.64±0.15 | 75.25±0.10 | 29.78±0.07 |
| YaRN | 95.93±0.04 | 98.71±0.00 | 33.70±0.04 |
| Res. YaRN (Ours) | 98.30±0.00 | 99.58±0.00 | 48.46±0.03 |

Table 1: The accuracy on OOD Positions (OOD Acc.) on POSGEN’s test set. All results are in percentage (%). We report both the mean and variance across five runs with different random seeds. We compare the same RoPE-based PE with or without our RESONANCE scaling. The best performance for each pair of settings on each subtask is marked in **Bold**.

$$h(x_0, x_1, x_2, x_3) = \sum_{i=0}^3 x_i \bmod 17 \text{ with vocabulary } \mathbb{V} = \{0, \dots, 16\}.$$

Our experiments involved training a two-layer Transformer. Each layer follows T5-Small’s configurations (Raffel et al., 2020) except for the position embeddings. In this model, each attention head has 64 dimensions. We apply different RoPE-based embeddings with the rotary base equal to 10,000. The models are trained on sequences of length $L = 64$, and evaluating on lengths of $L' = 256$ for OOD Accuracy. In this experiment setting, each head has 32 RoPE features, out of which the first 17 are pre-critical dimensions with a wavelength less than the maximum training length. We generated 10,000 training sequences, and 1,000 each for validation and testing, and ensured that the first $j + k = 4$ tokens in each sequence do not overlap to testify whether the model learns the correct generation mechanism. We averaged results over 5 seeds. A more detailed setting is provided in Appendix C.1.

6.1.2 Results and Analysis

Table 1 displays the comparison of the OOD accuracy. In most cases, RESONANCE ROPE and RESONANCE YARN outperform their counterparts lacking the Resonance technique, showcasing significantly better performance and reduced variance

in OOD scenarios. This improvement indicates a superior adaptation to OOD position embeddings through minimized Positional Encoding (PE) interpolation. An exception is observed when applying RESONANCE ROPE to the Recursive subtask, likely due to the dominance of extrapolated post-critical dimensions in OOD positions. This issue can be mitigated by employing a RoPE scaling technique such as YaRN, which effectively counters the extrapolation of post-critical dimensions. Among all configurations, RESONANCE YARN exhibits the highest OOD performance, demonstrating the synergy between RoPE scaling methods and the Resonance technique.

Figure 3 plots validation losses against training epochs for different PEs, illustrating the training dynamics. The introduction of the Resonance technique leads to a reduction in the lowest validation loss for both RoPE and YaRN, with RESONANCE ROPE achieving even lower validation losses than YaRN in the Semi-Recursive subtask. Furthermore, the validation loss trajectories for RESONANCE ROPE and RESONANCE YARN remain lower than those of their counterparts in all subtasks, further demonstrating the enhanced OOD generalization capability of our approach.

6.2 LLM Fine-tuning Evaluation

6.2.1 Experiment Setup

In this section, we apply our proposed RESONANCE ROPE to the current state-of-the-art RoPE scaling method, YaRN (Peng et al., 2024). More specifically, we replace the original position embeddings of LLaMA2 7B and 13B (Touvron et al., 2023b) with a series of scaled position embeddings, including the NTK-Aware scaling (bloc97, 2023; Xiong et al., 2023; Liu et al., 2024), Dynamic NTK-Aware Scaling (Peng et al., 2024; Rozière et al., 2023), and YaRN (Peng et al., 2024).

| Setting | Ctx Len. | Coursera | GSM | QuALITY | TOEFL | CodeU | SFiction | Avg. |
|---|----------|--------------|--------------|--------------|--------------|-------------|--------------|--------------|
| LLaMA2-Chat 7B | | | | | | | | |
| Dynamic NTK-Aware (no FT) | 32K | 31.98 | 32.00 | 34.65 | 59.11 | 1.11 | 36.72 | 32.59 |
| NTK-Aware ($s = 8$, no FT) | 32K | 36.77 | 3.00 | 26.73 | 34.2 | 1.11 | 50.78 | 25.43 |
| YaRN ($s = 8$, FT@32K, 50 epochs.) | 32K | 36.05 | 19.00 | 33.17 | 50.56 | 4.44 | 56.25 | 33.24 |
| Resonance YaRN ($s = 8$, FT@32K, 50 epochs.) | 32K | <u>36.48</u> | 22.00 | 34.16 | <u>55.76</u> | 0.00 | <u>57.03</u> | 34.24 |
| YaRN ($s = 8$, FT@4K, 400 epochs.) | 32K | 35.03 | 24.00 | <u>37.62</u> | <u>57.62</u> | 4.44 | 60.94 | 36.61 |
| Resonance YaRN ($s = 8$, FT@4K, 400 epochs.) | 32K | 36.34 | <u>27.00</u> | 40.59 | 56.51 | <u>3.33</u> | 61.72 | 37.58 |
| LLaMA2-Chat 13B | | | | | | | | |
| Dynamic NTK-Aware (no FT) | 16K | 29.22 | 39.00 | 40.59 | 63.94 | 1.11 | 39.84 | 35.62 |
| NTK-Aware ($s = 4$, no FT) | 16K | 40.26 | 21.00 | 38.12 | 65.43 | 1.11 | 46.88 | 35.47 |
| YaRN ($s = 4$, FT@16K, 100 epochs.) | 16K | 38.08 | 39.00 | <u>43.07</u> | 65.43 | 0.00 | 63.28 | 41.48 |
| Resonance YaRN ($s = 4$, FT@16K, 100 epochs.) | 16K | 38.66 | 39.00 | 43.56 | 65.06 | 1.11 | <u>62.50</u> | 41.65 |
| YaRN ($s = 4$, FT@4K, 400 epochs.) | 16K | <u>41.72</u> | 34.00 | 41.09 | 66.91 | <u>2.22</u> | 48.44 | 39.06 |
| Resonance YaRN ($s = 4$, FT@4K, 400 epochs.) | 16K | 41.86 | <u>35.00</u> | 42.57 | <u>65.80</u> | 5.56 | 48.44 | 39.87 |

Table 2: Long text evaluations on some closed-ended tasks in L-Eval. “Ctx Len” means the target context length of the model after scaling its PE. “FT@32K, 50 epochs” means the model is fine-tuned on 32K sequence length for 50 epochs. The settings with “no FT” are not fine-tuned after modifying its position embedding. We highlight the best and second-best performance for each base model in **Bold** and Underline, respectively.

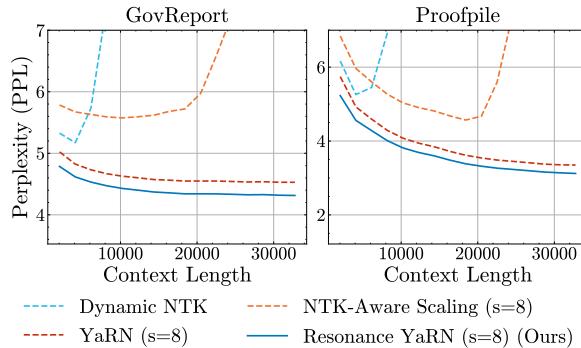


Figure 4: The perplexity of LLaMA-Chat 7B with different position embeddings on GovReport and Proofpile.

For YaRN and RESONANCE YARN, We use a scaling factor of 8 and 4 for LLaMA2 7B and 13B to extend their context window from 4K to 32K and 16K, respectively. For the configurations that require fine-tuning, we fine-tune the LLM with the scaled position embedding on the training set of PG19 (Rae et al., 2020) with the fine-tuning setting and hyperparameters adopted directly from YaRN (Peng et al., 2024), with the only difference being that we control the total training token count to be approximately 100M. A more detailed fine-tuning setting can be found in Appendix C.2. We test the model’s performance on two TSTL scenarios: language modeling evaluation on long-text sequences and long-text downstream application performance.

6.2.2 Perplexity on Long Sequence

We evaluate the model’s language modeling performance on GovReport (Huang et al., 2021) and Proofpile (Azerbaiyev, 2022). We randomly select 50 samples from each dataset and report the final perplexity in text fragments of gradually increased

length. We report the results in Figure 4. Of the tested methods, RESONANCE YARN achieves the lowest perplexity across all context lengths. Especially, RESONANCE YARN achieves a lower perplexity compared to YaRN with the same set of hyperparameters optimized for YaRN, demonstrating the benefit of applying the Resonance technique to existing RoPE scaling methods.

6.2.3 Real-world Task Evaluation

Lastly, we test the real-world task performance of LLaMA2-Chat 7B and 13B’s performance with different RoPE scaling strategies on L-Eval (An et al., 2023)’s close ended task suite, a long-text LLM benchmark covering a wide range of domains such as school lectures, long conversations and novels. We fine-tune the model with different RoPE scaling strategies using two different strategies: training on shorter sequences (4K length) for more epochs, and training on longer sequences (32K or 16K length) for less epochs. All settings requiring fine-tuning keep the training token count to be approximately 100M. The results are listed in Table 2.

Although no single setting in the experiment achieves the best result on all subtasks, we observe that applying RESONANCE YARN achieves better average performance in different training settings and model sizes compared to its counterpart YaRN setting. This further proves the compatibility of the Resonance technique and RoPE scaling methods, and the better length extrapolation performance brought by our proposed method.

7 Conclusion

We introduce RESONANCE ROPE, a novel enhancement of RoPE that focuses on minimizing the interpolation of RoPE features for OOD positions, thereby reducing the generalization gap and improving LLM’s performance on train-short-test-long (TSTL) scenarios. Additionally, we present a novel synthetic benchmark, POSGEN, which provides a fine-grained analysis of the model’s TSTL performance regarding various token dependency patterns. Extensive experiments on our proposed POSGEN and two LLM-based evaluations demonstrate RESONANCE ROPE’s efficacy in identifying OOD positions and its compatibility with current RoPE scaling strategies. Future work includes exploring RESONANCE ROPE’s performance on other foundational models, and the identification of more optimal wavelength combinations for RoPE features.

Limitations

Our proposed RESONANCE ROPE focus on reducing the interpolation of only RoPE’s pre-critical dimensions on OOD positions. However, this method does not solve the extrapolation issue on RoPE’s post-critical dimensions, which has been shown to be also detrimental to LLM’s length extrapolation performance. Thus, the technique of RESONANCE ROPE needs to be combined with another RoPE scaling method that can reduce extrapolation on RoPE’s post-critical dimensions, e.g., YaRN, to achieve the full potential of LLM in TSTL scenarios. Such combination has been our focus in Section 6.2.

Secondly, applying LLMs to long text sequences requires considerations of both performance and efficiency due to the super-linear complexity of Transformers w.r.t input length. As an improvement of the position embeddings, we focus only on improving Transformers’ performance in TSTL scenarios. An interesting future direction would be to apply RESONANCE ROPE to efficient Transformers for both performance and efficiency enhancements.

Lastly, benchmarking LLMs is still an open question, as there is currently no benchmark to thoroughly test the performance of LLMs, especially on long-sequence tasks. We expect that a more comprehensive long-text benchmark would further improve the validity of the experiment results.

References

- Chenxin An, Shansan Gong, Ming Zhong, Mukai Li, Jun Zhang, Lingpeng Kong, and Xipeng Qiu. 2023. [L-eval: Instituting standardized evaluation for long context language models](#). *CoRR*, abs/2307.11088.
- Zhangir Azerbayev. 2022. [zhangir-azerbayev/proof-pile](#).
- 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. 2023. [Longbench: A bilingual, multitask benchmark for long context understanding](#). *CoRR*, abs/2308.14508.
- bloc97. 2023. NTK-Aware Scaled RoPE allows LLaMA models to have extended (8k+) context size without any fine-tuning and minimal perplexity degradation.
- Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023. [Extending context window of large language models via positional interpolation](#). *CoRR*, abs/2306.15595.
- Tri Dao. 2023. [Flashattention-2: Faster attention with better parallelism and work partitioning](#). *CoRR*, abs/2307.08691.
- Luyang Huang, Shuyang Cao, Nikolaus Nova Parulian, Heng Ji, and Lu Wang. 2021. [Efficient attentions for long document summarization](#). In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1419–1436. Association for Computational Linguistics.
- 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.
- Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. [Mistral of experts](#). *CoRR*, abs/2401.04088.
- Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. 2023. [The impact of positional encoding on length generalization in transformers](#). In *Advances in Neural Information Processing Systems*, volume 36, pages 24892–24928.

- Bingbin Liu, Jordan T. Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. 2023. [Transformers learn shortcuts to automata](#). In *The Eleventh International Conference on Learning Representations*.
- Xiaoran Liu, Hang Yan, Chenxin An, Xipeng Qiu, and Dahua Lin. 2024. [Scaling laws of roPE-based extrapolation](#). In *The Twelfth International Conference on Learning Representations*.
- Ilya Loshchilov and Frank Hutter. 2019. [Decoupled weight decay regularization](#). In *7th International Conference on Learning Representations*.
- Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. 2023. [Progress measures for grokking via mechanistic interpretability](#). In *The Eleventh International Conference on Learning Representations*.
- OpenAI. 2023. [GPT-4 technical report](#). *CoRR*, abs/2303.08774.
- 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*.
- 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*.
- Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. 2020. [Compressive transformers for long-range sequence modelling](#). In *8th International Conference on Learning Representations*.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. [Exploring the limits of transfer learning with a unified text-to-text transformer](#). *J. Mach. Learn. Res.*, 21:140:1–140:67.
- Jie Ren, Samyam Rajbhandari, Reza Yazdani Am-inabadi, Olatunji Ruwase, Shuangyan Yang, Min-jia Zhang, Dong Li, and Yuxiong He. 2021. [Zero-offload: Democratizing billion-scale model training](#). In *2021 USENIX Annual Technical Conference*, pages 551–564. USENIX Association.
- Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémie 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.
- Anian Ruoss, Grégoire Delétang, Tim Genewein, Jordi Grau-Moya, Róbert Csordás, Mehdi Bennani, Shane Legg, and Joel Veness. 2023. [Randomized positional encodings boost length generalization of transformers](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1889–1903, Toronto, Canada. Association for Computational Linguistics.
- Uri Shaham, Maor Ivgi, Avia Efrat, Jonathan Berant, and Omer Levy. 2023. [ZeroSCROLLS: A zero-shot benchmark for long text understanding](#). In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 7977–7989, Singapore. Association for Computational Linguistics.
- 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.
- Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. 2021. [Long range arena : A benchmark for efficient transformers](#). In *9th International Conference on Learning Representations*.
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. [Llama: Open and efficient foundation language models](#). *CoRR*, abs/2302.13971.
- Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenjin 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 Miaylov, 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, Bin 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. 2023b. [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, Łukasz 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, pages 5998–6008.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. [Chain-of-thought prompting elicits reasoning in large language models](#). In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022*.

Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. 2022. [Memorizing transformers](#). In *The Tenth International Conference on Learning Representations*.

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

Liang Zhao, Xiaocheng Feng, Xiachong Feng, Bing Qin, and Ting Liu. 2023. [Length extrapolation of transformers: A survey from the perspective of position encoding](#). *CoRR*, abs/2312.17044.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V. Le, and Ed H. Chi. 2023. [Least-to-most prompting enables complex reasoning in large language models](#). In *The Eleventh International Conference on Learning Representations*.

Dawei Zhu, Nan Yang, Liang Wang, Yifan Song, Wen-hao Wu, Furu Wei, and Sujian Li. 2024. [PoSE: Efficient context window extension of LLMs via positional skip-wise training](#). In *The Twelfth International Conference on Learning Representations*.

A Proof of Theorem 1

Proof. All we need is to prove that for each $\mathbf{x} \in \mathbb{R}^d$, each $n \in \mathbb{N} \setminus \{0, \dots, L-1\}$ and each $i = 0, \dots, 2c-1$ we can find $m \in \{0, \dots, L-1\}$, such that $\tilde{f}(\mathbf{x}, m)_i = \tilde{f}(\mathbf{x}, n)_i$. By definition, it is equivalent to solving the equations:

$$(\mathbf{R}_{\tilde{\Theta}, m}^d \mathbf{W} \mathbf{x})_i = (\mathbf{R}_{\tilde{\Theta}, n}^d \mathbf{W} \mathbf{x})_i$$

for m , given i , n , and x .

The RoPE feature matrix $\mathbf{R}_{\tilde{\Theta}, m}^d$ is defined as block-diagonal with 2×2 blocks given by Equation 3. Hence, given i , x and n , the equation reduces to equality of a linear combination of trigonometric functions:

$$a \cos m\tilde{\theta}_i + b \sin m\tilde{\theta}_i = a \cos n\tilde{\theta}_i + b \sin n\tilde{\theta}_i$$

for $a, b \in \mathbb{R}$, depending on x and i . This equality clearly holds if $m\tilde{\theta}_i - n\tilde{\theta}_i$ is a multiple of 2π :

$$(m - n)\tilde{\theta}_i = 2\pi k,$$

for some $k \in \mathbb{Z}$. By our construction, $\frac{2\pi}{\tilde{\theta}_i}$ is a natural number. Hence, to finish the proof that we can solve our initial equation for m , we need to show that we can find integer k to satisfy:

$$\left(n - \frac{2\pi}{\tilde{\theta}_i} k \right) \in \{0, \dots, L-1\}$$

for $n \in \mathbb{N} \setminus \{0, \dots, L-1\}$. This is where we use the pre-critical dimension condition: for $i = 0, \dots, 2c-1$, by definition of c , we have the inequality $0 \leq \frac{2\pi}{\tilde{\theta}_i} < L$. Taking $k = \lfloor \frac{n\tilde{\theta}_i}{2\pi} \rfloor$ will give us the required range for m and hence finish the proof. \square

B Comparison Between Feature Gap and Embedded Vector Distance

Our proposed feature gap metric, as defined in Equation 8, shares similarities with the “embedded vector distance” metric introduced by Xiong et al. (2023):

$$d(f, \hat{f}) = \max_{x \in \mathcal{X}} \min_{\substack{k \in \{0, \dots, N-1\} \\ j \in \{0, \dots, \hat{N}-1\}}} \text{dist}[f(x, k), \hat{f}(x, j)] \quad (9)$$

where $\mathcal{X} \subset \mathbb{R}^d$ represents the set of vectors requiring positional embedding. This equation assesses the discrepancy in Rotary Position Embedding (RoPE) before and after a scaling operation.

The distance calculation specifically compares the original RoPE, $f(\cdot, \cdot)$, to the scaled RoPE, $\hat{f}(\cdot, \cdot)$, with token positions beginning at zero. It aims to quantify the alterations in position embedding due to the scaling process.

In contrast, our feature gap metric is tailored for a more practical and common scenario, where models are trained or fine-tuned on short sequences using the already scaled RoPE embeddings. This setting emphasizes the generalization gap of the RoPE features between training and testing position ranges. The core hypothesis is that a smaller discrepancy in the RoPE features of new token positions relative to those encountered during training correlates with enhanced model generalization to novel token positions. Our metric diverges from the “embedded vector distance” in two significant aspects to better align with our use-case:

- The distance computation shifts to compare scaled RoPE across different token positions, reflecting the operational context where training involves short sequences (train-short) and testing involves longer sequences (test-long).
- We modify the token position ranges, k and j , to represent token positions observed during training (in-distribution) and testing (out-of-distribution), respectively, to directly measure the generalization gap.

This adaptation of the metric allows for a more targeted evaluation of the model’s ability to generalize across different token positional distributions, which is critical in scenarios where sequence length varies significantly between training and deployment.

C Detailed Experiment Settings

In this section, we provide the detailed experiment settings for both our synthetic task evaluation on POSGEN and LLM-based evaluations on both upstream language modeling evaluation and downstream real-world application evaluations.

C.1 Synthetic Task Evaluation on POSGEN

For the synthetic task experiments in Section 6.1.1, we train a two-layer Transformer on each of the subtasks, with each layer following the configuration of a T5-Small model (Raffel et al., 2020). For each subtask, we train the model with different position embeddings on a training set with 10,000 sequence samples of length 64. The validation and

test sets each contains 1,000 sequence samples with length 256. The sequences in the training, validation and test sets do not overlap in the first $j + k$ tokens. For all YaRN and RESONANCE YARN settings, we train the model with YaRN and RESONANCE YARN applied to the model with a scaling factor $s = 4$, which corresponds to the TSTL setting of our evaluation. Each model is trained on each subtask for 150 epochs with a language modeling-style cross-entropy loss. Training was done with AdamW optimizer (Loshchilov and Hutter, 2019), using learning rate 2×10^{-4} and weight decay 1×10^{-2} . We use a batch size of 128 for all experiments. All hyperparameters were tuned to maximize YaRN’s validation set performance on the Semi-Recurrent subtask. All synthetic task evaluations were performed on a single NVIDIA V100 32G GPU.

C.2 LLM Evaluations

For the LLM-based evaluations in Section 6.2, we fine-tune LLaMA2-Chat 7B or LLaMA2-Chat 13B (Touvron et al., 2023b) after replacing its original RoPE position embedding with RoPE scaled with different strategies:

- **NTK-Aware Scaling** (bloc97, 2023; Xiong et al., 2023; Liu et al., 2024), which scales the base b in Equation 1 to $s \cdot b$, where s is the scaling factor. We evaluate the performance without fine-tuning as used in bloc97 (2023).
- **Dynamic NTK-Aware Scaling** (Peng et al., 2024; Rozière et al., 2023). This method dynamically computes the scaling factor considering the current sequence length L_c and the original context window length L : $s = \frac{L_c}{L}$. Due to the high cost of frequently recomputing RoPE features, we evaluated its performance without fine-tuning.
- **YaRN** (Peng et al., 2024). We evaluate its performance after fine-tuning.

For NTK-Aware scaling and Dynamic NTK-Aware scaling settings, we replace the original RoPE position embeddings in the model with the scaled ones and test their performance without fine-tuning following (bloc97, 2023; Peng et al., 2024). For YaRN and RESONANCE YARN settings, we fine-tune the model for approximately 100M tokens on PG19’s training set (Rae et al., 2020). Our target scaled length for the 7B and 13B models is

32K and 16K, respectively, which corresponds to a scaling factor $s = 8$ and $s = 4$ for the position embeddings of the two models.

For both the long-sequence perplexity evaluation in Section 6.2.2 and real-world task evaluations in Section 6.2.3, the hyperparameters for the LLM experiments follow the configurations provided in Peng et al. (2024)², with the only modification that we fine-tune the model on approximately 100M tokens. More specifically, we use $\alpha = 1$ and $\beta = 32$ for YaRN and RESONANCE YARN as suggested by Peng et al. (2024). The model was trained with a language modeling-style cross entropy loss. Training was done with the AdamW optimizer (Loshchilov and Hutter, 2019) using learning rate 2×10^{-5} and weight decay 1×10^{-2} . We use a batch size of 1 on each of the GPUs. The learning rate warm-up is applied to the first 5% of the total training steps. Models were fine-tuned with BF16 precision, FlashAttention 2 (Dao, 2023) and DeepSpeed ZeRO-3 Offload (Ren et al., 2021) on four NVIDIA A100 40G GPUs.

For the real-world task evaluations in Section 6.2.3, we further compare two different fine-tuning strategies:

1. **Fine-tuning on long sequences for less epochs.** We directly fine-tune the model on the target sequence lengths after applying the scaled position embeddings. For LLaMA2-Chat 7B and 13B, we fine-tune the model on sequences with length 32,768 for 50 steps and sequences with length 16,384 for 100 steps, respectively.
2. **Finetuning on short sequences for more epochs.** We fine-tune the model on the original pre-training sequence length after applying the scaled position embeddings. For both LLaMA2-Chat 7B and 13B, we fine-tune the model on sequences with length 4,096 for 400 steps.

²<https://github.com/jquesnelle/yarn>.