Improving Length Generalization via Position Index Warping

Anonymous ACL submission

Abstract

Length generalization mitigates the impact of mismatched conditions in training and testing where models are trained on short sequences but evaluated on longer ones. Among many factors that may impact length generalization in Transformer-based models, positional encoding has been identified as a critical one, but in-depth analysis on its impact on the length generalization issue is still limited. In this work, we advance our understanding via analyzing positional biases introduced by different positional encoding approaches. Our analysis suggests a novel approach to improve length generalization. The method warps positional indices during training, which can be considered as a data augmentation technique. Empirical studies on various tasks (e.g., algorithmic reasoning tasks and language modeling) showcase the effectiveness of our proposed method.

1 Introduction

Transformer (Vaswani et al., 2017) based Language Models (LMs) often struggle to generalize to longer input sequences (Csordás et al., 2021; Delétang et al., 2023; Newman et al., 2020; Furrer et al., 2020). Several studies have identified Positional Encoding (PE) as a factor that impacts length generalization. For example, Press et al. (2022); Chen et al. (2023a); Sun et al. (2023); Ruoss et al. (2023); Tao et al. (2023); Chi et al. (2023) all manipulate PEs to improve length generalization.

Press et al. (2022) propose a novel PE called ALiBi where a positional bias is added to inner product between keys and queries. The bias term decays linearly as the positional difference (between keys and queries) increases. The authors empirically show that ALiBi generalizes better to longer inputs in language modeling tasks than a few other PE mechanisms. More recently, KERPLE (Chi et al., 2022) extends ALiBi by generalizing its linear positional bias to quadratic and logarithmic forms, achieving further improvement in terms of length generalization. Nevertheless, it is not entirely analyzed why ALiBi and KERPLE excel for length generalization. If it is merely due to the fact that they impose positional bias in a relative fashion, then why do they outperform another relative PE like Rotary Positional Encoding (RoPE) (Su et al., 2021)?

While ALiBi and KERPLE are successful for length generalization, nowadays, many published models (e.g., Falcon, LLaMA) are based on RoPE. Although it does not generalize to longer input sequences (Press et al., 2022; Chi et al., 2023; Kazemnejad et al., 2023; Chi et al., 2022), RoPE has its own merits (e.g., efficiency (Dao et al., 2022)) thus it wouldn't become a solution to the length generalization problem to simply replace the RoPE layers with ALiBi/KERPLE layers and retrain them. For this reason, it is more explored to develop orthogonal techniques as a countermeasure instead of innovating more new PEs. One prominent work is Position Interpolation (PI) (Chen et al., 2023a), where input positional indices at test time are linearly down-scaled to match the range of positional indices used in training. In contrast, randomized PE (Ruoss et al., 2023) works in an opposite direction, by up-scaling position indices at training time to match testing length. More recently, Peng et al. (2023) introduce a novel interpolation method, YaRN, which adjusts the base frequency of RoPE to preserve high-frequency information.

Our study analyzes the existing PEs and explores the factors that influence the varying capabilities of different PEs in terms of length generalization. We first investigate factors contributing to ALiBi's superior performance over RoPE in length generalization. We conduct a comparative analysis of positional biases in ALiBi and RoPE, particularly under the simplified conditions imposed by uniform query and key inputs. This comparison reveals that exposure to unseen positional biases at test time contributes to RoPE's limited length generalization. Furthermore, we demonstrate the techniques to mitigate the impact of positional biases, such as PI and Randomized PE, may still be ineffective in some real-world scenarios. We identify that the mismatch in position index density between training and testing, which both Randomized PE and PI confront, can impair the length generalization of models.

Through an understanding of how PEs impact length generalization, we establish desiderata for PEs to ensure effective handling of longer input sequences: *maintaining a large attention span*, *minimizing positional bias gap and reducing discrepancy in the position index density*. Building on these insights, we propose two approaches to warp position indices during training. One approach involves randomly down-scaling position indices and the other employs a non-linear skewing of these indices towards the tail. Our method is distinct from recent works (Chen et al., 2023a; Xiong et al., 2023; Roziere et al., 2023), which require fine-tuning on long sequences. Notably, our approach does not necessitate further training.

2 Backgrounds and Related Work

In this section, we review previous works manipulating PEs to improve length generalization. We focus on Relative Positional Encoding (RPE) methods, which has been shown to be more effective than absolute PE's (Shaw et al., 2018; Yang et al., 2019; Raffel et al., 2020; Dai et al., 2019). In the following sections, PE means RPE unless it is indicated otherwise. For analysis, we categorize PE methods into two types, based on how positional bias is injected to the self-attention modules of Transformer models.

Additive Positional Encoding. Additive PE adds positional bias to the inner product of keys and queries. The attention thus becomes:

$$\mathrm{softmax}_n[(\mathbf{W}_q\mathbf{q}_m)^\top(\mathbf{W}_k\mathbf{k}_n) + B(m-n)], \quad (1)$$

where m denotes the position of a query token and n indicates the position of a key token. The positional bias, $B(m-n)$, can be defined in different ways (Press et al., 2022; Raffel et al., 2020; Chi et al., 2023, 2022). Among them, a seminal one is ALiBi (Press et al., 2022), where $B(m-n) = -b \cdot (m-n)$ and b is a head-specific slope ($\frac{1}{2^i}(i = 1, \cdots, 8)$ when $n_{head} = 8$).

Multiplicative Positional Encoding. Multiplicative PE multiplies positional bias to the query-key inner product. An important work in this category is RoPE (Su et al., 2021), and its attention is defined as:

$$\mathrm{softmax}_n[(\mathbf{W}_q\mathbf{q}_m)^\top \mathbf{R}_{\Theta,m-n}(\mathbf{W}_k\mathbf{k}_n)],$$
$$\mathbf{R}_{\Theta,m-n}$$
$$\stackrel{\mathrm{def}}{=} \mathrm{diag}\left\{\begin{bmatrix} \cos(m-n)\theta_s & \sin(m-n)\theta_s \\ -\sin(m-n)\theta_s & \cos(m-n)\theta_s \end{bmatrix}\right\} \quad (2)$$

where $\theta_s = 10000^{-\frac{2(s-1)}{d}}$, $s = 1, \ldots, d/2$, and d denotes the dimension of embedding for each head.

Improving PEs for Length Generalization. Some studies introduce augmentation techniques on top of existing PE methods (Ruoss et al., 2023; Tao et al., 2023; Li and McClelland, 2022; Kiyono et al., 2021). Randomized PE (Ruoss et al., 2023) assigns random (ordered) positional encodings in the full range of possible test positions to each training instance. Similarly, Tao et al. (2023) proposes to use randomly padded inputs. Moreover, Chen et al. (2023a) and Sun et al. (2023) demonstrate some pathological behaviors of RoPE, i.e.,exacerbated oscillations over long distances. To avoid the oscillation, PI (Chen et al., 2023a) linearly down-scales input position indices during inference to match the range of position indices used at training time. Sun et al. (2023) modifies the RoPE's formula to have less oscillation at long distances and uses sliding attention window during inference. Concurrently, Peng et al. (2023) introduce YaRN, which adjusts the base frequency of RoPE to preserve high-frequency information. And Roziere et al. (2023); Xiong et al. (2023); Bai et al. (2023) apply this interpolation technique in their approaches to fine-tune LMs for managing long input sequences.

We remark both PI and Randomized PE linearly scale position indices. As Randomized PE samples position indices from a uniform distribution, it has the effect of applying linear scaling in expectation.

3 Weakness of Existing Methods

In this section, we examine existing methods, identify their limitations and discuss the causes of their failures. In Section 3.1, we conduct an analysis of RoPE and ALiBi. The enhanced techniques applied to RoPE, such as PI and Randomized PE, are addressed in Section 3.2.

3.1 How RoPE and ALiBi Fail

We compare the positional biases introduced by RoPE and ALiBi. Specifically, we assume uniform and unit-length queries and keys throughout all positions, *i.e.*, $\mathbf{W}_q\mathbf{q}_m = \mathbf{W}_k\mathbf{k}_n$, and $\|\mathbf{W}_q\mathbf{q}_m\| = \|\mathbf{W}_k\mathbf{k}_n\| = 1$ for all m and n's. Simplifying this way allows us to focus on the only factor we care about, positional bias caused by PEs. We derive the raw attention scores (prior to calling softmax) for RoPE and ALiBi, respectively.

RoPE : As $\mathbf{R}_{\Theta,m-n}$ is block-diagonal with rotating each 2-element segment of $\mathbf{W}_q\mathbf{q}_m$ or $\mathbf{W}_k\mathbf{k}_n$, we can further simplify by assuming each of their 2-element segments has constant length, $\sqrt{2/d}$. With this idealization, RoPE's raw attention score is

$$\begin{aligned} & Attn_{raw}(m-n;\Theta) \\ & \stackrel{\text{def}}{=} (\mathbf{W}_q\mathbf{q}_m)^\top \mathbf{R}_{\Theta,m-n}(\mathbf{W}_k\mathbf{k}_n) \\ & = \frac{2}{d}\sum_{s=1}^{d/2} \cos(m-n)\theta_s \\ & \xrightarrow{d\to\infty} \mathbb{E}_{\theta\sim p(\theta)}[\cos(m-n)\theta], \end{aligned} \tag{3}$$

where $p(\theta)$ is logUniform$(10^{-4}, 1)$ if adopting the conventional $\theta_s = 10000^{\frac{-2(s-1)}{d}}$ where $s = 1, ..., d/2$. We remark that similar derivations are seen in (Su et al., 2021; Sun et al., 2023).

ALiBi : Following Eq. (1) and taking $B(m-n) = -\frac{1}{2^i}(m-n)$ for $i = 1, \ldots, n_{head}$, the raw attention score is simply

$$Attn_{raw}(m-n) = 1 - \frac{1}{2^i}(m-n). \tag{4}$$

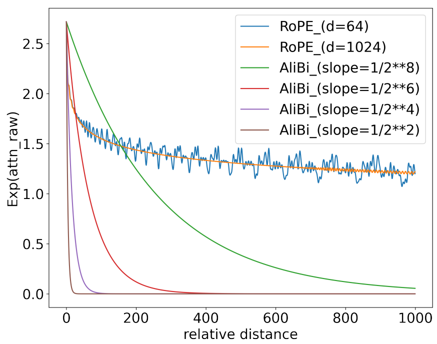

Figure 1: Exponentiated raw attention scores in RoPE (blue and orange lines) and ALiBi with various slopes (other remaining lines). For RoPE, we vary d to show its effect. We fix $m = 1000$ and vary n to track relative distance $m-n$ in the horizontal axis.

We remark $Attn_{raw}(m-n)$ reflects how positional biases are injected, as content embeddings are uniform throughout all positions. To see the impact from these biases, we plot the numerator of softmax operations, $\exp(Attn_{raw}(m-n))$, for RoPE and ALiBi in Figure 1. Clearly, $\exp(Attn_{raw}(m-n))$ of ALiBi converges to zero, but RoPE converges to strictly positive values. In fact, regardless of $p(\theta)$, $\exp(\mathbb{E}_{\theta\sim p(\theta)}[\cos(m-n)\theta]) \geq e^{-1}$. This indicates ALiBi's attention span is narrower, echoing the smaller empirical receptive field claimed by Chi et al. (2023). In contrast, RoPE's attention span is wider, abling to attend to tokens far apart.

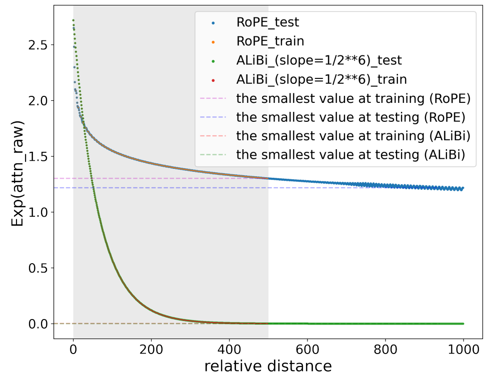

Figure 2: Exponentiated raw attention scores of RoPE (blue and orange lines) and ALiBi (green and red lines) during training ($l = 500$) and testing ($l = 1000$). In the case of RoPE, when the test input length is larger than the training length, it confronts the positional biases unseen during training. In contrast, ALiBi avoids the issue due to its narrow attention span. The gray zone represents the lengths seen at training time.

While the ability to attend to distant tokens may be beneficial for length generalization, multiple studies (Press et al., 2022; Chi et al., 2023; Kazemnejad et al., 2023) show that RoPE's performance significantly drops for longer input sequences. This can be explained by examining how positional biases vary for different sequence lengths, shown in Figure 2. When doing inference on longer sequences, RoPE will be exposed to positional biases it did not see during training. In contrast, ALiBi does not have this issue due to its narrower attention span. The gap between the two dotted lines (the pink line and the purple line in Figure 2) clearly illustrates the difference between positional biases at training and testing time in RoPE. We conjecture this gap is the culprit of RoPE's failure at length generalization.

However, ALiBi sacrifices its attention span. This limitation becomes evident in tasks requiring Transformer models to refer to distant tokens, e.g., copying ancient input tokens to outputs (Ruoss et al., 2023; Kazemnejad et al., 2023; Li and McClelland, 2022; Ontañón et al., 2022). To illustrate this, we set up a small experiment. We train a 2-

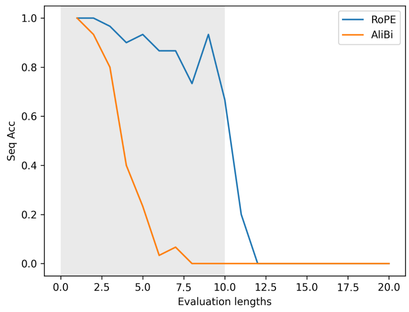

Figure 3: In *Copy* task, ALiBi (orange line) underperforms RoPE (blue line) due to its narrow attention span. The gray zone represents the lengths seen during training.

layer Transformer, with input sequences of varying lengths, ranging from 1 to 10 tokens. The model is then evaluated on test inputs whose lengths range from 1 to 20 tokens. As shown in Figure 3, ALiBi starts to fail beyond a certain point of evaluation lengths, which is earlier than RoPE. This observation suggests ALiBi's narrow attention span could be a disadvantage for length generalization.

Therefore, we deduce that an ideal solution for achieving length generalization is to *minimize positional bias gap* while *maintaining a large attention span*.

3.2 Limitation of Linearly Scaled Position Indices

Applying linear scaling to the position index in PI (Chen et al., 2023a) and Randomized PE (Ruoss et al., 2023) on RoPE prevents positional bias gaps by ensuring a match between the position index ranges used in training and testing phases. Moreover, as both techniques are based on RoPE, they inherently sustain a large attention span. However, they still suffer from a performance drop at long test sequences, as shown in Table 1. Why does this performance drop happen?

Taking PI[1] as an example, the integer position index, i, is multiplied by the ratio between lengths of training and test data,

$$pos(i) = i \times \frac{\text{training input length}}{\text{test input length}}, \quad (5)$$

where $i = 0, 1, \ldots, L_{test} - 1$ and L_{test} denotes test input length. Down-scaling position indices during testing makes *intervals of (relative) position indices at testing time denser than during training*. The discrepancy in the density causes the following two issues:

[1]Note that we adopt a no fine-tuning setup in our approach to PI, assuming real-world scenarios where additional fine-tuning data with longer sequences may be hard to get. This setup is also adopted in Chen et al. (2023b); Li et al. (2023).

(i) **Fractional indices.** At testing time, PI introduces many fractional indices (Eq. 5), which are not seen during training.

(ii) **Attention sensitivity is too low to identify positions precisely.** We define *attention sensitivity* as:

$$Attn\_sen(i) = Attn_{raw}(pos(i-1)) - Attn_{raw}(pos(i)), \quad (6)$$

where $i = 0, 1, \ldots, l-1$ and l indicates the sequence length. With this definition, towards the end of $Attn_{raw}$, attention sensitivity tend to diminish significantly, a phenomenon that becomes more pronounced in longer sequences because of the long-tail shape of the function (see Figure A in Appendix A). During training, the model learns to distinguish positions with attention sensitivity greater than $Attn\_sen(L_{train})$, where L_{train} denotes training input length. However, when dealing with longer test sequences, we have to face lower attention sensitivity at the tail of the raw attention score function, which is not experienced during training. It leads to a performance degradation. Concurrently, Peng et al. (2023) also points out the challenge in accurately detecting positions when the relative distance is extremely close, especially when relying on the use of PI.

4 Method: Warping Position Indices

Based on the analyses in the previous sections, we establish three desiderata when improving PE in terms of length generalization: (i) not sacrificing attention span, (ii) minimizing positional bias gap, and (iii) reducing discrepancy in the position index density between training and testing phases. To meet these criteria, we build on PI, which already meets the first two, and propose a novel method to satisfy the third criterion by tackling the inconsistency issue in the position index density between training and testing phases.

Section 3.2 outlines the challenges anticipated from using a denser density of positional indices at testing time. To address these issues, we devise strategies to expose the model to denser positional indices during its training phase. Given that training sequences are shorter than testing sequences, it is impractical to expose a model to denser positional indices over the entire range. Thus, we introduce two distinct approaches for *warping training position indices*, each targeting different segments of the curve (orange line) shown in Figure. 1. The first approach, *creating fractional position indices*,

Table 1: The perplexity of RoPE at different test lengths. The model is trained and evaluated on WikiText-103. Even though Randomized PE (Chen et al., 2023a) and PI without fine-tuning (Ruoss et al., 2023) succeed in improving RoPE, they still suffer from performance degradation in longer sequences. These performance drops can be caused by a discrepancy in the density of position indices between training and testing phases.

Methods/Test length	**512**	**1024**	**2048**	**4096**
RoPE	19.22	37.08	133.50	307.38
Position Interpolation	19.22	24.77	48.39	99.56
Randomized PE ($max\_position = 8072$)	63.78	65.72	67.89	69.62

addresses unseen fractional position indices on the upper left of the curve. The second approach, *skewing positions non-linearly towards the tail*, specifically addresses attention sensitivity discrepancies as well as introduces fractional position indices in the bottom right of the curve.

To avoid compromising model performance across seen sequences, we alter the position indices for only a portion of the training instances, chosen at random, while retaining the original position indices for the remaining instances. By adopting these augmented position indices during training, the model can be robust against various position indices at inference. Our strategy can be considered as PE augmentation, like those suggested in several studies (Ruoss et al., 2023; Tao et al., 2023; Li and McClelland, 2022; Kiyono et al., 2021).

Creating fractional position indices (*head_warping*): We linearly down-scale position indices to match fractional positions we will see at testing time:

$$pos_{warp}(j) = \alpha \cdot j, \tag{7}$$

where $0 < \alpha < 1$ and $j = 0, 1, \cdots, L_{train} - 1$.

The ways to set α varies, depending on whether input sequence lengths are fixed. In the case of PI, α is set to $\frac{1}{r}$ where $r = \frac{\text{test input length}}{\text{training input length}}$ and the position indices r times denser than training position indices are used at inference assuming training input lengths and test input lengths are fixed (e.g., in language modeling). On the other hand, when there are variations in sequence lengths during both training and testing, the value of α is sampled randomly within the range of $0 < \alpha < 1$. Sampling α exposes the model to a range of fractional position indices. Further details on determining α for each experimental setup can be found in Section 5.

Skewing positions non-linearly towards the tail (*tail_warping*): With knowing the decreased attention sensitivity (defined in Eq. 6) in the tail of the raw attention function at testing time, we introduce a non-linear skewing function that redistributes position indices towards the end of indices. Warping position indices towards the tail leads to intervals of position indices at the tail denser, allowing us to train a model with reduced attention sensitivity. The skew function is applied to position indices as:

$$pos_{warp,f_{skew}}(j) = l \cdot f_{skew}(\frac{j}{l}), \tag{8}$$

where $j = 0, 1, \cdots, L_{train} - 1$. Let $f_{skew} : [0, 1] \to [0, 1]$ be a function satisfying the boundary conditions $f_{skew}(0) = 0$ and $f_{skew}(1) = 1$. f_{skew} is a concave function in $[\epsilon, 1]$, where $\epsilon \approx 0$ and $\epsilon > 0$. Figure 4 illustrates the cumulative distribution function (CDF) of $Beta(2, 5)$, which is an example of f_{skew}.

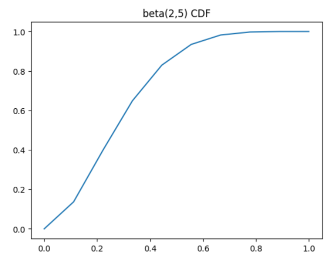

Figure 4: The CDF of Beta(2,5)

f_{skew} can be determined by evaluating whether using $pos_{warp,f_{skew}}(\cdot)$ could lead to low attention sensitivity experienced at inference. As a criterion to determine f_{skew}, we define the *total difference* in attention sensitivity between training and test phases as:

$$d_{i,f} = |Attn\_sen_f^{tr}(L_{train} - i) \\ - Attn\_sen^{te}(L_{test} - i)|,$$

$$\text{total\_diff} = \sum_{i=0}^{c} d_{i,f}, \tag{9}$$

where $Attn\_sen_f^{tr}$ represents attention sensitivity at training time where the warped position indices ($pos_{warp,f_{skew}}$) are used and $Attn\_sen^{te}$ indicates attention sensitivity at inference with the use of PI.

Table 2: The *total difference* at varying training/test input lengths. The test input lengths vary from 2 to 4 times the training input length. When the training input length is 20, we set c as 10. When the training input length is 500, we set c as 200. *sqrt* represents a square root function and *beta* denotes the CDF of $Beta(2, 5)$. Applying the skewing functions to position indices results in reducing the *total difference*. The best results are **bold** and the second-best results are <u>underlined</u>.

Training Input Length	f_{skew}	Test Input Length	
		2 times	**4 times**
20	w\o	0.099	0.069
	w/ *sqrt*	**0.016**	<u>0.063</u>
	w/ *beta*	<u>0.054</u>	**0.012**
500	w\o	0.061	0.066
	w/ *sqrt*	**0.028**	<u>0.031</u>
	w/ *beta*	<u>0.031</u>	**0.013**

To focus on the tail of the distributions, we sum over the last c position indices.

The optimal skewing function, f^*_{skew}, should minimize the total difference (i.e., reduce the train/test attention sensitivity mismatch). However, conducting an exhaustive search for f^*_{skew} is infeasible. Instead, we conduct an experiment with the idealized $Attn_{raw}$ (Eq. 3) using different candidates for f_{skew} including the CDF of $Beta(2, 5)$ and a square root function (See Figure B in Appendix A for illustration). We identify which function minimizes the total difference with $Attn_{raw}$. In Table 2, we demonstrate the *total difference* with and without applying f_{skew} using the idealized $Attn_{raw}$. We remark warping position indices using either of the two functions leads to reducing the total difference. Notably, the empirical study suggests the choice of f_{skew} depends on the ratio of test input length to training input length. Indeed, in the case of larger ratios, the CDF of $Beta(2, 5)$ exhibits greater effectiveness, likely due to its stronger skewing towards larger position indices. This skewness aligns better with longer test input lengths, making it a more suitable choice compared to the square root function, which is less skewed. We provide more details on how to determine f_{skew} in Section 5.

5 Experiments

In this section we showcase our method enhances the ability to generalize on longer sequences. We evaluate our method on a variety of tasks including algorithmic reasoning tasks and language modeling. With the primary goal of enhancing RoPE, we mainly compare against the original **RoPE** (Su et al., 2021), and its enhancements through two techniques: **PI** (Chen et al., 2023a) and **Randomized PE** (Ruoss et al., 2023). Note that unlike Chen et al. (2023a), we focus on zero-shot scenarioes where no additional fine-tuning is allowed. Furthermore, we compare our method against **ALiBi** (Press et al., 2022), which is categorized as additive PE.

5.1 Algorithmic Reasoning Tasks

We adopt the experimental setup from Kazemnejad et al. (2023), using algorithmic reasoning tasks. The model is trained on sequences up to a specific length and tested on both seen and longer lengths within each task. Notably, having a large attention span beyond a training length is a critical factor to excel at these tasks, as we observed in Section 3.1.

Dataset & Model Architecture. We showcase the efficacy of our method on *copy, reverse, sort* and *summation* tasks. Training instances adhere to a length distribution of Uniform$(2, M)$, while testing sequences follow Uniform$(2, 5M)$. And M is 20 for *copy* and *reverse* and 8 for *sort* and *summation*. Further details are provided in Appendix B.1. We employ a decoder-only Transformer architecture based on T5 (Raffel et al., 2020).

Training & Inference Procedure. We apply *created fractional position indices* (Eq. 7) to 15% of training instances and *skewing position indices towards the tail* (Eq. 8) for 15% of the instances. For *creating fractional position indices*, we randomly sample α (in Eq. 7) from $\{0.4, 0.5, 0.6, 0.7, 0.8\}$ with equal probabilities. For f_{skew}, we use the square root function. The remaining training instances undergo no warping. We also provide the detailed training procedure and additional experimental results in Appendix B.1. During inference, our method uses PI. As PI is initially proposed under fixed training and test input lengths, we adapt the interpolation ratio to $\frac{M}{L_{te}}$ for each test input length L_{te} ($> M$) where M denotes the maximum training input length. Similarly, Peng et al. (2023) also suggest a *dynamic scaling method* that adjusts position indices by considering varying lengths during inference. For Randomized PE, we limit the maximum length to $10M$, which is smaller than 2048 used as the maximum length in the original paper (Ruoss et al., 2023).

Results. The proposed PE exhibits superior performance compared to other baselines across a majority of tasks, as illustrated in Fig. 5. The various

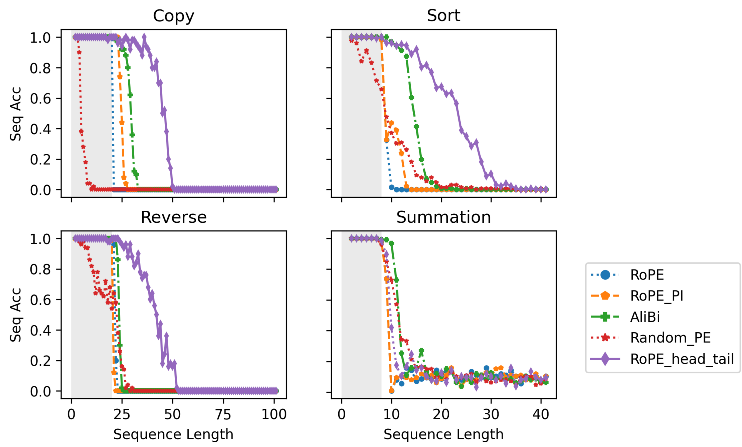

Figure 5: The sequence accuracy w.r.t. varying evaluation lengths on different algorithmic reasoning tasks. Ours (RoPE_head_tail) is capable to extrapolate on long inputs without performance degradation in seen lengths, which surpasses the baselines in most cases. The shaded areas denote the lengths seen during training.

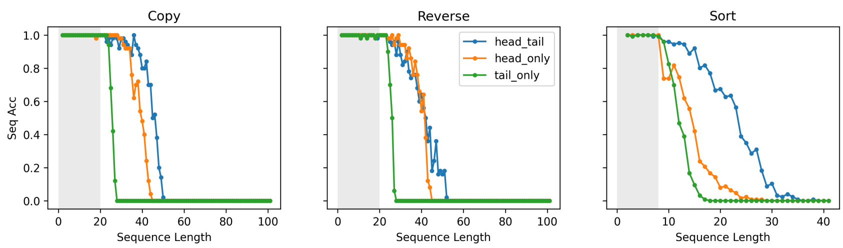

Figure 6: Warping position indices using the two proposed approaches leads to performance improvement. The *head_tail* indicates where we apply *creating fractional position indices* to 15% of training instances and *skewing position indices towards the tail* for 15% of the instances. The *head_only (or tail_only)* represents when we apply *creating fractional position index (skewing position indices towards the tail)* to 30% of training instances.

PEs show similar performance trends across different test input lengths in *summation*, which is also shown in Kazemnejad et al. (2023). This similarity could arise since the task relies less on position information compared to other tasks. The average number of sequence accuracies across all test input lengths are provided in Table. A in Appendix B.1.

We further highlight the effectiveness of each warping approach as depicted in Figure 6. Applying both of the proposed approaches (*head_tail*) yields the most significant improvement. This synergy arises from the different focus of each warping approach on separate parts, i.e., the beginning and tail of position indices. When using either approach alone, warping indices in the beginning (*head_only*) shows more effective than warping towards the tail (*tail_only*). However, warping towards the tail helps to succeed in generalization on longer sequences when used in conjunction with warping indices in the beginning.

Interestingly, the inferior performance of ALiBi on these tasks echos the observation by Kazemnejad et al. (2023); Ruoss et al. (2023), but contradicts Press et al. (2022)'s finding (superiority in language modeling). We attribute this to ALiBi's narrow attention span (elaborated in Section 3.1), which can be fatal in algorithmic tasks, that requires to refer to distant tokens.

The empirical results of Randomized PE in our setup differ from Ruoss et al. (2023)'s observations in two aspects. First, Randomized PE's gains in long sequences come at the cost of reduced effectiveness with short ones. This trade-off is apparent for tasks like *reverse* and *sort* (Figure 5), while other PEs achieve nearly perfect accuracy on seen lengths. Using random position information, even in an ordered manner, can attribute to the decline. Second, the performance of Randomized PE highly depends on the maximum position (L) as illustrated in Figure C in Appendix B.1. The smaller the maximum position, the better the performance. Note that even with the smaller $L(= 3M)$, Randomized PE is still inferior to our method (Table A in Appendix B.1). There are several differences between our setup and theirs (Ruoss et al., 2023), including token counts, training samples, steps, and evaluation metrics, which may cause inconsistent outcomes. The inconsistent empirical results indicate

Table 3: Perplexity evaluation on WikiText-103 (Merity et al., 2017). The training context size is 512. The best results are bold and the second best results are underlined.

PE category	Model	Test Length 512	1024	2048	3072	4096
Multiplicative	RoPE	**19.22**	37.08	133.50	223.02	307.38
	Position Interpolation	**19.22**	24.77	48.39	76.14	99.56
	Randomized PE ($L = 8072$)	63.78	65.72	67.89	69.23	69.62
	Randomized PE ($L = 4096$)	39.35	39.10	39.21	39.47	39.28
	Randomized PE ($L = 1024$)	19.83	<u>18.80</u>	·	·	·
Additive	ALiBi	<u>19.31</u>	**18.38**	**17.88**	**17.81**	**17.67**
Multiplicative	Ours	19.80	18.87	<u>18.26</u>	<u>18.45</u>	<u>19.00</u>

Randomized PE's effectiveness might be confined to specific experimental setups.

5.2 Language modeling

We train a language model with shorter context window size and evaluate on longer input sequences. We use WikiText-103 for training and testing.

Model Architecture & Training Procedure. We adopt the GPT-2 architecture (Radford et al., 2019), a causal Transformer, with *base* configurations. We train with a 512 context window size, and evaluate on up to 8 times longer inputs. We apply both of the proposed approaches randomly to 15% of training instances, as in the algorithmic tasks. The α is fixed to $\frac{1}{6}$. For f_{skew}, we use the CDF of $Beta(2, 5)$. The detailed hyperparameters and the additional experimental results are elaborated in Appendix B.2.

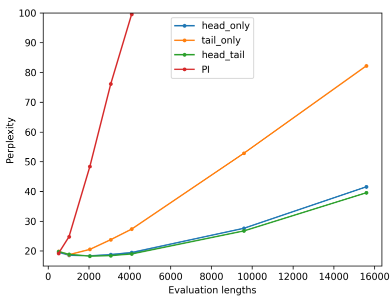

Figure 7: Perplexity in language modeling. Warping position indices decreases perplexity compared to the baseline (red line). Even if there is a marginal distinction between *head_only* and *head_tail* in evaluation lengths up to 4000, as the test input length increases, the effect of warping towards the tail becomes more noticeable. The legends of the graph are the same as Figure 6.

Results. Table 3 reports perplexity of each method. The proposed method outperform other multiplicative PEs by large margins, especially when the test length increases. In addition, our method enhances RoPE's capacity to generalize on long inputs significantly with a trivial performance drop on seen lengths.

Figure 7 illustrates the effect of each warping approach, reflecting our observations in Section 5.1. That is, although *head_only* is more effective when only one method is applied, it is the most effective when both methods are applied together (*head_tail*). In addition, warping towards the tail helps to achieve better performance on longer sequences. This suggests that for enhanced language modeling performance, prioritizing the alignment of position indices in the beginning is more crucial than the alignment in the tail.

Notably, Randomized PE shows the similar trend to the algorithmic tasks in Section 5.1. Randomized PE causes not only increases in perplexity for longer inputs but also substantial performance degradation in seen lengths. The decline in seen lengths can be attributed to the practice of injecting random noise, which can be particularly ineffective when dealing with fixed training and test input lengths. We also notice a strong correlation between the performance of Randomized PE and L (the maximum position) as well.

ALiBi exhibits superior performance compared to multiplicative PEs in language modeling, which is inconsistent with the results seen in the algorithmic tasks. This inconsistency could stem from the diminished importance of using distant tokens in language modeling, whereas algorithmic tasks rely on their usage.

6 Conclusion

By investigating the impact of various positional encodings on length generalization, we explored their positional biases. Our novel method warps position indices during training as a form of data augmentation. We validated the effectiveness of our approach through empirical studies on various tasks.

7 Limitations

In this work, we only focus on a decoder-only Transformer. We leave applying our approach to other architectures for future work. We have not exhaustively explored all potential hyperparameters, leaving room for future exploration.

Acknowledgements

References

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

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

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

Ta-Chung Chi, Ting-Han Fan, Peter J Ramadge, and Alexander Rudnicky. 2022. Kerple: Kernelized relative positional embedding for length extrapolation. *Advances in Neural Information Processing Systems*.

Ta-Chung Chi, Ting-Han Fan, Alexander Rudnicky, and Peter Ramadge. 2023. Dissecting transformer length extrapolation via the lens of receptive field analysis. In *Association for Computational Linguistics*.

Róbert Csordás, Kazuki Irie, and Jürgen Schmidhuber. 2021. The devil is in the detail: Simple tricks improve systematic generalization of transformers. *Empirical Methods in Natural Language Processing*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *Association for Computational Linguistics*.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*.

Grégoire Delétang, Anian Ruoss, Jordi Grau-Moya, Tim Genewein, Li Kevin Wenliang, Elliot Catt, Chris Cundy, Marcus Hutter, Shane Legg, Joel Veness, et al. 2023. Neural networks and the chomsky hierarchy. *International Conference on Learning Representations*.

Daniel Furrer, Marc van Zee, Nathan Scales, and Nathanael Schärli. 2020. Compositional generalization in semantic parsing: Pre-training vs. specialized architectures. *arXiv preprint arXiv:2007.08970*.

Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. 2023. The impact of positional encoding on length generalization in transformers. *Advances in Neural Information Processing Systems*.

Shun Kiyono, Sosuke Kobayashi, Jun Suzuki, and Kentaro Inui. 2021. Shape: Shifted absolute position embedding for transformers. *Empirical Methods in Natural Language Processing*.

Shanda Li, Chong You, Guru Guruganesh, Joshua Ainslie, Santiago Ontanon, Manzil Zaheer, Sumit Sanghai, Yiming Yang, Sanjiv Kumar, and Srinadh Bhojanapalli. 2023. Functional interpolation for relative positions improves long context transformers.

Yuxuan Li and James L McClelland. 2022. Systematic generalization and emergent structures in transformers trained on structured tasks. *arXiv preprint arXiv:2210.00400*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. *International Conference on Learning Representations*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. *International Conference on Learning Representations*.

Benjamin Newman, John Hewitt, Percy Liang, and Christopher D Manning. 2020. The eos decision and length extrapolation. *In Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*.

Santiago Ontanón, Joshua Ainslie, Vaclav Cvicek, and Zachary Fisher. 2022. Making transformers solve compositional tasks. *Association for Computational Linguistics*.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023. Yarn: Efficient context window extension of large language models. *arXiv 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. *International Conference on Learning Representations*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*.

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. *The Journal of Machine Learning Research*.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. 2023.

Code llama: Open foundation models for code. *arXiv preprint arXiv: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. *Association for Computational Linguistics*.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. *North American Chapter of the Association for Computational Linguistics*.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2021. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*.

Yutao Sun, Li Dong, Barun Patra, Shuming Ma, Shaohan Huang, Alon Benhaim, Vishrav Chaudhary, Xia Song, and Furu Wei. 2023. A length-extrapolatable transformer. *Association for Computational Linguistics*.

Mingxu Tao, Yansong Feng, and Dongyan Zhao. 2023. A frustratingly easy improvement for position embeddings via random padding. *arXiv preprint arXiv:2305.04859*.

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

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

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in Neural Information Processing Systems*.

Improving Length Generalization via Position Index Warping

Supplementary Material

A Visualization

A.1 Long-tail Shapes in Raw Attention Score

In Figure A, attention sensitivity in the tail of raw attention scores decreases as the input sequence length increases.

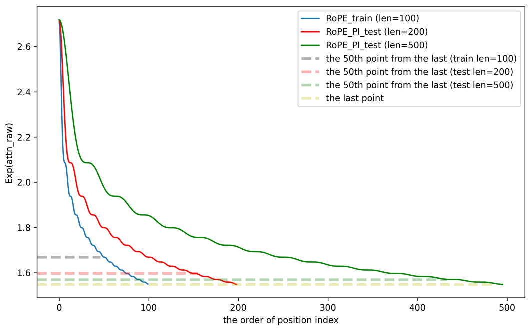

Figure A: The blue line indicates exponentiated attention scores during training ($l = 100$). The red ($l = 200$) and green ($l = 500$) lines represent where position interpolation (PI) is applied at inference. The dotted lines (gray, pink and green) represent the 50th point from the last for each training and testing case. The yellow dotted line represents the last point. The gaps between the yellow dotted line and the remaining lines decreases as the length increases. This shows that the attention sensitivity in the tail is lower for longer sequences.

A.2 Different Skewness Levels of f_{skew}

Figure B illustrates the square root function, which is an example of f_{skew}.

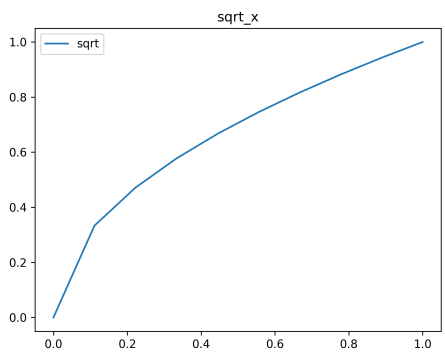

Figure B: The square root function.

B Experimental details

B.1 Algorithmic Reasoning Tasks

Datasets. For detailed explanation on each of *copy, reverse, sort* and *summation* tasks, refer to Kazemnejad et al. (2023). For each task, we use 100K samples for training and 10K samples for testing. The performance evaluation is based on sequence accuracy, the exact-match accuracy of answers compared to the ground truth.

Hyperparameters. We use the hyperparameters suggested in Kazemnejad et al. (2023). We employ a decoder-only Transformer architecture based on T5 (Raffel et al., 2020), with the *base* configuration: $n_{layer} = 12$, $n_{head} = 12$, and $d_{model} = 768$. The total number of trainable parameters is 107M. We use the AdamW optimizer (Loshchilov and Hutter, 2019) with learning rate of $3e$-5 and weight decay of 0.05. We set a polynomial a learning rate scheduler and a warm-up for 6% of training steps. We use a batch size of 64 and train models for 40,000 steps.

Main Results Supplement. The average of sequence accuracy across all testing lengths is shown in Table A. Our method shows its superiority over other baselines.

The Effect of L in Randomized PE. Figure C shows the performance of Randomized PE varies depending on the maximum position length. As L increases, the performance degrades.

The Effect of f_{skew}. We change f_{skew} from the square root function to the CDF of $Beta(2, 5)$ and evaluate how f_{skew} impacts on its performance, as demonstrated in Table A. Using the square root function shows more effective than the CDF of $Beta(2, 5)$ in algorithmic reasoning tasks.

B.2 Language Modeling

Hyperparameters. We use the hyperparameters suggested in *HuggingFace*[2] for causal language modeling with the *base* configuration. The total number of trainable parameters is 117M. We use the AdamW optimizer with a learning rate of $5e$-5 without a weight decay. We set β_1 as 0.9 and β_2 as 0.999. We use a 64 batch size.

[2] https://huggingface.co

Table A: The average of sequence accuracy (%) on all testing lengths from 2 to 5M. R.P. denotes Randomized PE. $10M, 5M$ and $3M$ denotes the maximum position length (L)

Task	**RoPE**	**RoPE+PI**	**ALiBi**	**R.P.** ($10M$)	**R.P.** ($5M$)	**R.P.** ($3M$)	**Ours (sqrt)**	**Ours (beta)**
Copy	19.2	23.5	27.9	3.9	8.8	26.8	**43.2**	**43.2**
Reverse	20.4	19.3	22.4	17.5	19.3	39.3	**39.8**	39.4
Sort	18.8	21.5	33.6	21.4	29.8	43.0	**53.7**	48.6
Sum	27.9	27.5	**33.0**	32.2	·	·	29.5	·

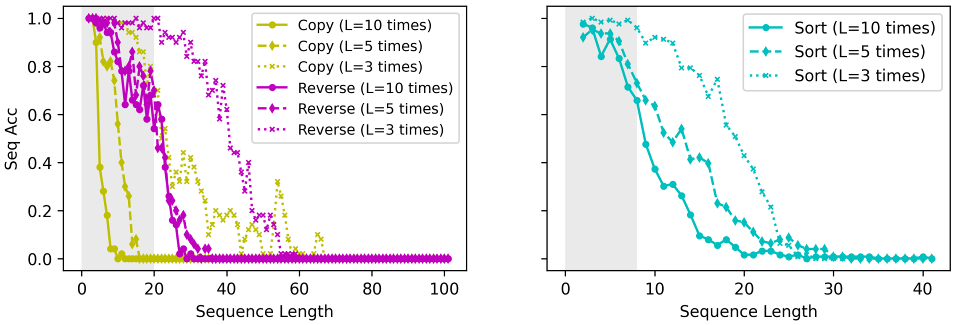

Figure C: The sequence accuracy w.r.t. varying evaluation lengths of Randomized PE with different L. For each task, L is set to 3x, 5x and 10x the maximum training length. The sequence accuracy appears inversely proportional to the maximum position (L). The shaded areas indicate seen lengths.

Table B: Perplexity evaluation on WikiText-103. The training context size is 512.

α	**512**	**1024**	**2048**	**3072**	**4096**
1/2	19.06	21.65	26.18	26.22	32.26
1/4	19.34	18.36	18.19	19.34	20.88
1/6	19.57	18.59	18.36	18.79	19.48
1/8	19.87	18.93	18.62	18.83	19.31

The Effect of α. We evaluated how α impacts on performance in language modeling as shown in Table B. The larger α, the larger the perplexity in longer test sequences. Conversely, the smaller α, the less effective it was for relatively short sequences. Thus, we decided to use α of $\frac{1}{6}$ to balance the performance in short and long sequences.