RoFormer: Enhanced Transformer with Rotary Position Embedding

Jianlin Su
Zhuiyi Technology Co., Ltd.
Shenzhen
bojonesu@wezhuiyi.com

Yu Lu
Zhuiyi Technology Co., Ltd.
Shenzhen
julianlu@wezhuiyi.com

Shengfeng Pan
Zhuiyi Technology Co., Ltd.
Shenzhen
nickpan@wezhuiyi.com

Bo Wen
Zhuiyi Technology Co., Ltd.
Shenzhen
brucewen@wezhuiyi.com

Yunfeng Liu
Zhuiyi Technology Co., Ltd.
Shenzhen
glenliu@wezhuiyi.com

Abstract

Position encoding in transformer architecture provides supervision for dependency modeling between elements at different positions in the sequence. We investigate various methods to encode positional information in transformer-based language models and propose a novel implementation named Rotary Position Embedding(RoPE). The proposed RoPE encodes absolute position information with rotation matrix and naturally incorporates explicit relative position dependency in self-attention formulation. Notably, RoPE comes with valuable properties such as flexibility of being expand to any sequence lengths, decaying inter-token dependency with increasing relative distances, and capability of equipping the linear self-attention with relative position information. As a result, our experiment has shown that the enhanced transformer with rotary position embedding, or RoFormer, achieves comparable or superior performance on various language modeling tasks[1].

proposal

contribution

1 Introduction

The sequential order of words plays a vital role in natural language. Recurrent-based models (RNNs) encode tokens' order by recursively computing a hidden state along the time dimension. Convolution-based models (CNNs) [11] were typically considered position-agnostic, but recent work [16] has shown that the commonly used padding operation can implicitly learn position information. In recent years, the effectiveness of transformer-based models was shown on various natural language processing (NLP) tasks such as context representation learning [8], machine translation [37], and language modeling [28]. Unlike recurrent-based and convolution-based models, transformer-based models utilize the self-attention architecture to capture the dependency among tokens in the context, which provides better parallelization than RNNs and can model longer intra-token relations than CNNs.

Since transformer-based models contain no recurrence and no convolution, and the self-attention architecture is shown to be position-agnostic [44], different approaches have been proposed to inject position information into the model. One line of works focuses on absolute position encoding, where absolute position encoding which are trainable [11, 8, 19, 6, 28, 27] or generated

[1] Code is available at www.anonymous.com

Preprint. Under review.

by pre-defined function [37] were added to context representations. The other line of works [26, 32, 14, 7, 43, 29, 18, 12, 15] focuses on relative position encoding, which typically injects relative position information into the attention calculation. In addition to these approaches, [20] has proposed to model the dependency of position encoding from the perspective with Neural ODE [3], and [38] has proposed to model the position information in complex space. Recent works [21, 13] proposed similar approaches to ours. [21] proposed to view positional attention as covariance and draw samples from centered random processes and [13] proposed the relative position embeddings for kernelizable attention.

In this work, we first establish a formal description of the position encoding problem in self-attention architecture and revisit previous works in Section 2. We then propose the rotary position encoding (RoPE) and study its properties in Section 3. Finally, we report experiments in section 4. Our contributions are as follows:

Contributions

- We investigate previous works on relative position encoding and find most of them based on the decomposition of adding position encoding to the context representations. We propose to encode relative position by multiplying the context representations with a rotation matrix with a clear theoretical interpretation.
- We study the properties of RoPE and show that it decays with the relative distance increased, which is desired for natural language encoding. We argue that previous relative position encoding approaches are not compatible with linear self-attention and show that RoPE can be used in such mechanism.
- We demonstrate that RoFormer achieves comparable or superior performance than peer models across various tasks. Importantly, we showcase in both English and Chinese that our model is more efficient in language modeling pre-training in terms of convergence.

2 Background and related work

2.1 Preliminary

Let $\mathbb{S}_N = \{w_i\}_{i=1}^N$ be a sequence of N input tokens with w_i being the i^{th} element. The corresponding word embedding of $\mathbb{S}_N$ can be denoted as $\mathbb{E}_N = \{\boldsymbol{x}_i\}_{i=1}^N$, where $\boldsymbol{x}_i \in \mathbb{R}^d$ is the d-dimensional word embedding vector of token w_i without position information. The self-attention first incorporates position information to the word embeddings and transforms them into queries, keys, and value representations.

$$\begin{aligned} \boldsymbol{q}_m &= f_q(\boldsymbol{x}_m, m) \\ \boldsymbol{k}_n &= f_k(\boldsymbol{x}_n, n) \\ \boldsymbol{v}_n &= f_v(\boldsymbol{x}_n, n) \end{aligned} \tag{1}$$

Where $\boldsymbol{q}_m, \boldsymbol{k}_n, \boldsymbol{v}_n$ incorporates the m^{th} and n^{th} position by functions f_q, f_k and f_v respectively. The attention weights are then calculated using query and key representations, and the output $\mathbf{o_m}$ can be computed as the weighted sum of value representations.

$$\begin{aligned} \mathbf{o}_m &= \sum_{n=1}^{N} a_{m,n} \boldsymbol{v}_n \\ a_{m,n} &= \frac{\exp(\frac{\boldsymbol{q}_m^\intercal \boldsymbol{k}_n}{\sqrt{d}})}{\sum_{j=1}^{N} \exp(\frac{\boldsymbol{q}_m^\intercal \boldsymbol{k}_j}{\sqrt{d}})} \end{aligned} \tag{2}$$

The research on position encoding of transformer mainly focuses on choosing suitable function forms of eq. (1).

2.2 Absolute position embedding

A typical choice of eq. (1) is

$$f_{t:t\in\{q,k,v\}}(\boldsymbol{x}_i, i) := \boldsymbol{W}_{t:t\in\{q,k,v\}}(\boldsymbol{x}_i + \boldsymbol{p}_i) \tag{3}$$

Where $\boldsymbol{p}_i \in \mathbb{R}^d$ is a d-dimensional vector depending of the position of token $\boldsymbol{x}_i$. [8, 19, 6, 28, 27] used a set of trainable vectors $\boldsymbol{p}_i \in \{\boldsymbol{p}_t\}_{t=1}^{L}$, where L is the maximum sequence length. On the other hand, [37] has proposed to generate $\boldsymbol{p}_i$ using the sinusoidal function.

$$\begin{cases} \boldsymbol{p}_{i,2t} & = \sin(k/10000^{2t/d}) \\ \boldsymbol{p}_{i,2t+1} & = \cos(k/10000^{2t/d}) \end{cases} \tag{4}$$

Where $\boldsymbol{p}_{i,2t}$ is the $2t^{th}$ element of the d-dimensional vector $\boldsymbol{p}_i$. In Section x, we will show that our proposed RoPE is related to this approach from the perspective of using the sinusoidal function, but ours incorporates relative position information by multiplying sinusoidal function to the context representation instead of adding.

proposal

2.3 Relative position embedding

[32] used a different setting of eq. (1) as following:

$$\begin{aligned} f_q(\boldsymbol{x}_m) &:= \boldsymbol{W}_q \boldsymbol{x}_m \\ f_k(\boldsymbol{x}_n, n) &:= \boldsymbol{W}_k(\boldsymbol{x}_n + \tilde{\boldsymbol{p}}_r^k) \\ f_v(\boldsymbol{x}_n, n) &:= \boldsymbol{W}_v(\boldsymbol{x}_n + \tilde{\boldsymbol{p}}_r^v) \end{aligned} \tag{5}$$

Where $\tilde{\boldsymbol{p}}_r^k, \tilde{\boldsymbol{p}}_r^v \in \mathbb{R}^d$ are trainable relative position embeddings. Note that $r = \text{clip}(m-n, r_{\text{min}}, r_{\text{max}})$ represents the relative distance between position m and n. They clipped the relative distance with the hypothesis that precise relative position information is not useful beyond a certain distance.

previous approach

Keeping the form of eq. (3), [7] has proposed to decompose the $\boldsymbol{q}_m^\intercal \boldsymbol{k}_n$ term in eq. (2) as

$$\boldsymbol{q}_m^\intercal \boldsymbol{k}_n = \boldsymbol{x}_m^\intercal \boldsymbol{W}_q^\intercal \boldsymbol{W}_k \boldsymbol{x}_n + \boldsymbol{x}_m^\intercal \boldsymbol{W}_q^\intercal \boldsymbol{W}_k \boldsymbol{p}_n + \boldsymbol{p}_m^\intercal \boldsymbol{W}_q^\intercal \boldsymbol{W}_k \boldsymbol{x}_n + \boldsymbol{p}_m^\intercal \boldsymbol{W}_q^\intercal \boldsymbol{W}_k \boldsymbol{p}_n \tag{6}$$

They replaced absolute position embedding $\boldsymbol{p}_n$ with its sinusoid-encoded relative counterpart $\tilde{\boldsymbol{p}}_{m-n}$ and replaced absolute position $\boldsymbol{p}_m$ in the third and fourth term with two trainable vectors $\mathbf{u}$, $\mathbf{v}$ independent of the query positions. Further, $\boldsymbol{W}_k$ is distinguished for the content-based and location-based key vectors $\boldsymbol{x}_n$ and $\boldsymbol{p}_n$, denoted as $\boldsymbol{W}_k$ and $\widetilde{\boldsymbol{W}}_k$, resulting in:

$$\boldsymbol{q}_m^\intercal \boldsymbol{k}_n = \boldsymbol{x}_m^\intercal \boldsymbol{W}_q^\intercal \boldsymbol{W}_k \boldsymbol{x}_n + \boldsymbol{x}_m^\intercal \boldsymbol{W}_q^\intercal \widetilde{\boldsymbol{W}}_k \tilde{\boldsymbol{p}}_{m-n} + \mathbf{u}^\intercal \boldsymbol{W}_q^\intercal \boldsymbol{W}_k \boldsymbol{x}_n + \mathbf{v}^\intercal \boldsymbol{W}_q^\intercal \widetilde{\boldsymbol{W}}_k \tilde{\boldsymbol{p}}_{m-n} \tag{7}$$

It is worth mentioning that they remove the position information in the value term by setting $f_v(\boldsymbol{x}_j) := \boldsymbol{W}_v \boldsymbol{x}_j$. Later works [29, 12, 18, 15] followed this step by only considering inject relative position information into the attention weights. [29] revised eq. (6) as

$$\boldsymbol{q}_m^\intercal \boldsymbol{k}_n = \boldsymbol{x}_m^\intercal \boldsymbol{W}_q^\intercal \boldsymbol{W}_k \boldsymbol{x}_n + b_{i,j} \tag{8}$$

Where $b_{i,j}$ is a trainable bias. [18] investigated the middle two terms of **??** and found little correlations between absolute positions and words. Follow [29], they have proposed to model a pair of words or positions by using different projection matrices.

$$\boldsymbol{q}_m^\intercal \boldsymbol{k}_n = \boldsymbol{x}_m^\intercal \boldsymbol{W}_q^\intercal \boldsymbol{W}_k \boldsymbol{x}_n + \boldsymbol{p}_m^\intercal \mathbf{U}_q^\intercal \mathbf{U}_k \boldsymbol{p}_n + b_{i,j} \tag{9}$$

[12] argued that a relative positions of word pair can only be fully modeled by using both the middle two terms of **??**, so they have proposed to replace the absolute position embeddings $\boldsymbol{p}_m$ and $\boldsymbol{p}_n$ in these two terms with relative position embeddings $\tilde{\boldsymbol{p}}_{m-n}$.

$$\boldsymbol{q}_m^\intercal \boldsymbol{k}_n = \boldsymbol{x}_m^\intercal \boldsymbol{W}_q^\intercal \boldsymbol{W}_k \boldsymbol{x}_n + \boldsymbol{x}_m^\intercal \boldsymbol{W}_q^\intercal \boldsymbol{W}_k \tilde{\boldsymbol{p}}_{m-n} + \tilde{\boldsymbol{p}}_{m-n}^\intercal \boldsymbol{W}_q^\intercal \boldsymbol{W}_k \boldsymbol{x}_n \tag{10}$$

[27] has compared four variants of relative position embeddings and shown that the variant similar to eq. (10) is the most efficient among the other three.

All these works modified eq. (6) based on the decomposition of eq. (3) under the self-attention setting in eq. (2), which is originally from [37]. They share the same nature that the position information is injected by deliberately adding to the context representations. Different from these work, our approach aims to derive the relative position encoding directly from eq. (1) under some constraints. In section 3.2.2, we show that the derived approach is more interpretable by incorporating relative position information with the rotation of context representations.

approach

3 Proposed approach

In this section, we discuss the proposed rotary position embedding (RoPE). We first formulate the relative position encoding problem in section 3.1, we then derive the RoPE in section 3.2 and investigate its properties in section 3.3.

3.1 Formulation

Language modeling in Transformer integrates position information of individual tokens through self-attention. We start from eq. (1) and notice that the $\boldsymbol{q}_m^\intercal \boldsymbol{k}_n$ term in eq. (2) actually facilitates information exchange between tokens at different positions. In order to incorporate relative position information, we require the inner product of query $\boldsymbol{q}_m$ and key $\boldsymbol{k}_n$ be formulated by a function g, which takes only the word embeddings $\boldsymbol{x}_m$, $\boldsymbol{x}_n$, and their relative position $m - n$ as input variables. In other words, we hope the inner product encodes position information only in the relative form:

$$\langle f_q(\boldsymbol{x}_m, m), f_k(\boldsymbol{x}_n, n) \rangle = g(\boldsymbol{x}_m, \boldsymbol{x}_n, m - n) \tag{11}$$

Next, finding such a encoding mechanism is equivalent to solve the function $f_q(\boldsymbol{x}_m, m)$ and $f_k(\boldsymbol{x}_n, n)$ that conforms above relation.

3.2 Rotary position embedding

3.2.1 A 2D case

We start from simple case with dimension $d = 2$. Under this setting, we make use of the geometric property of vectors on 2D plane and its complex form to prove (refer to supplementary materials for more details) that a solution to our formulation eq. (11) is:

$$\begin{aligned} f_q(\boldsymbol{x}_m, m) &= (\boldsymbol{W}_q \boldsymbol{x}_m) e^{im\theta} \\ f_k(\boldsymbol{x}_n, n) &= (\boldsymbol{W}_k \boldsymbol{x}_n) e^{in\theta} \\ g(\boldsymbol{x}_m, \boldsymbol{x}_n, m - n) &= \mathrm{Re}[(\boldsymbol{W}_q \boldsymbol{x}_m)(\boldsymbol{W}_k \boldsymbol{x}_n)^* e^{i(m-n)\theta}] \end{aligned} \tag{12}$$

where $\mathrm{Re}[\cdot]$ is the real part of a complex number and $(\boldsymbol{W}_k \boldsymbol{x}_n)^*$ represents the conjugate complex number of $(\boldsymbol{W}_k \boldsymbol{x}_n)$. $\theta \in \mathbb{R}$ is a preset non-zero constant. We can further write $f_{\{q,k\}}$ in matrix multiplication:

$$f_{\{q,k\}}(\boldsymbol{x}_m, m) = \begin{pmatrix} \cos m\theta & -\sin m\theta \\ \sin m\theta & \cos m\theta \end{pmatrix} \begin{pmatrix} W_{\{q,k\}}^{(11)} & W_{\{q,k\}}^{(12)} \\ W_{\{q,k\}}^{(21)} & W_{\{q,k\}}^{(22)} \end{pmatrix} \begin{pmatrix} x_m^{(1)} \\ x_m^{(2)} \end{pmatrix} \tag{13}$$

where $(x_m^{(1)}, x_m^{(2)})$ is $\boldsymbol{x}_m$ expressed in the 2D coordinates. Similarly, function g can be turned into matrix form. Thus, we come up with a solution to formulation in section 3.1 under the 2D case. Specifically, incorporating relative position embedding is straightforward: simply rotate the affine-transformed word embedding vector by amount of angle in multiples of its position index. Due to this characteristic, we name it *Rotary Position Embedding*.

3.2.2 General form

In order to generalize our result in 2D to any $\boldsymbol{x}_i \in \mathbb{R}^d$ where d is even, we divide the d-dimension space in to $d/2$ sub-spaces and combine them in merit of the linearity of inner product, turning $f_{\{q,k\}}$ into:

$$f_{\{q,k\}}(\boldsymbol{x}_m, m) = \boldsymbol{R}_{\Theta,m}^d \boldsymbol{W}_{\{q,k\}} \boldsymbol{x}_m \tag{14}$$

where

$$\boldsymbol{R}^d_{\Theta,m} = \begin{pmatrix} \cos m\theta_1 & -\sin m\theta_1 & 0 & 0 & \cdots & 0 & 0 \\ \sin m\theta_1 & \cos m\theta_1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cos m\theta_2 & -\sin m\theta_2 & \cdots & 0 & 0 \\ 0 & 0 & \sin m\theta_2 & \cos m\theta_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cos m\theta_{d/2} & -\sin m\theta_{d/2} \\ 0 & 0 & 0 & 0 & \cdots & \sin m\theta_{d/2} & \cos m\theta_{d/2} \end{pmatrix} \tag{15}$$

is the rotary matrix with pre-defined parameters $\Theta = \{\theta_i = 10000^{-2(i-1)/d}, i \in [1, 2, ..., d/2]\}$. An graphic illustration of RoPE is shown in fig. 1.

Applying our RoPE to self-attention in eq. (2), we have:

$$\boldsymbol{q}_m^\intercal \boldsymbol{k}_n = (\boldsymbol{R}^d_{\Theta,m}\boldsymbol{W}_q\boldsymbol{x}_m)^\intercal(\boldsymbol{R}^d_{\Theta,n}\boldsymbol{W}_k\boldsymbol{x}_n) = \boldsymbol{x}_m^\intercal\boldsymbol{W}_q R^d_{\Theta,n-m}\boldsymbol{W}_k\boldsymbol{x}_n \tag{16}$$

where $\boldsymbol{R}^d_{\Theta,n-m} = (\boldsymbol{R}^d_{\Theta,m})^\intercal \boldsymbol{R}^d_{\Theta,n}$. Notice that $\boldsymbol{R}^d_\Theta$ is an orthogonal matrix, which ensures the stability during the process of encoding position information. In addition, due to the sparsity of R^d_Θ, applying matrix multiplication directly as in eq. (16) is not computational efficient, we provide another realization in the supplementary material.

In contrast to the additive nature of position embedding method used by other works, i.e. eqs. (3) to (10), our approach is multiplicative. Moreover, RoPE naturally incorporates relative position information through rotation matrix product instead of altering terms in the expanded formulation of additive position encoding when applied with self-attention.

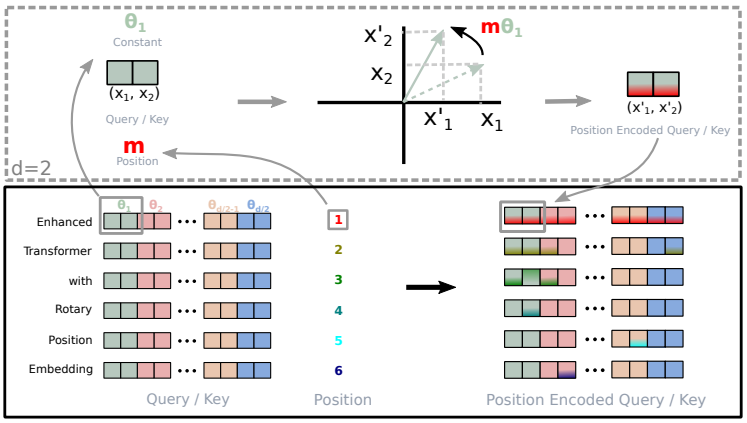

Figure 1: Implementation of Rotary Position Embedding(RoPE).

3.3 Properties of RoPE

Long-term decay: Following [37], we choose $\theta_i = 10000^{-2i/d}$. One can prove that this setting provides a long-term decay property (refer to supplementary materials for more details), which means the inner-product will decay when the relative position increase. This property coincides with the intuition that a pair of tokens with long relative distance should have less connection.

RoPE with linear attention: The self-attention can be rewritten in a more general form.

general form

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V})_m = \frac{\sum_{n=1}^N \text{sim}(\boldsymbol{q}_m, \boldsymbol{k}_n)\boldsymbol{v}_n}{\sum_{n=1}^N \text{sim}(\boldsymbol{q}_m, \boldsymbol{k}_n)} \tag{17}$$

The original self-attention chooses $\mathrm{sim}(\boldsymbol{q}_m, \boldsymbol{k}_n) = \exp(\boldsymbol{q}_m^\intercal \boldsymbol{k}_n / \sqrt{d})$. Notice that the original self-attention need to compute the inner product of query and key for every pair of tokens, which has quadratic complexity $\mathbb{O}(N^2)$. Follow [17] , linear attentions reformulate equation 17 as

$$\mathrm{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V})_m = \frac{\sum_{n=1}^{N} \phi(\boldsymbol{q}_m)^\intercal \varphi(\boldsymbol{k}_n) \boldsymbol{v}_n}{\sum_{n=1}^{N} \phi(\boldsymbol{q}_m)^\intercal \varphi(\boldsymbol{k}_n)} \tag{18}$$

where $\phi(\cdot), \varphi(\cdot)$ are usually non-negative functions. [17] has proposed $\phi(x) = \varphi(x) = \mathrm{elu}(x) + 1$ and first computed the multiplication between keys and values using the associative property of matrix multiplication. [33] has proposed to use softmax function to normalize queries and keys separately before the inner product, which is equivalent to $\phi(\boldsymbol{q}_i) = \mathrm{softmax}(\boldsymbol{q}_i)$ and $\phi(\boldsymbol{k}_j) = \exp(\boldsymbol{k}_j)$. For more details about linear attentions, we encourage readers to refer to original papers. In this section, we focus on discussing incorporating RoPE with equation 18. Since RoPE injects position information by rotation, which keeps the norm of hidden representations unchanged, we can combine RoPE with linear attentions by multiplying the rotation matrix with the outputs of the non-negative functions.

$$\mathrm{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V})_m = \frac{\sum_{n=1}^{N} \left(\boldsymbol{R}_{\Theta,m}^{d} \phi(\boldsymbol{q}_m)\right)^\intercal \left(\boldsymbol{R}_{\Theta,n}^{d} \varphi(\boldsymbol{k}_n)\right) \boldsymbol{v}_n}{\sum_{n=1}^{N} \phi(\boldsymbol{q}_m)^\intercal \varphi(\boldsymbol{k}_n)} \tag{19}$$

It is worth mentioning that we keep the denominator unchanged to avoid the risk of dividing zero, and the summation in the numerator could contain negative terms. Although the weights for each value $\boldsymbol{v}_i$ in equation 19 are not strictly probabilistic normalized, we argue that such computation can still model the importance of values.

4 Experiment

We evaluate the proposed RoFormer on various NLP tasks. First, the performance of our model on machine translation is investigated and reported in section 4.1. Then, we compare our RoPE implementation with BERT[8] during the pre-training stage in section 4.2. Based on the pre-trained model, in section 4.3, we further carry out evaluations across different downstream tasks from GLUE benchmarks[34]. In Addition, we experiment the proposed RoPE with linear attention in PerFormer[5] in section 4.4. By the end, additional tests on Chinese data are included in section 4.5. All the experiments are done on two cloud severs with 4 x V100 GPUs.

4.1 Machine Translation

We first demonstrate the performance of RoFormer on sequence-to-sequence language translation task. Details of dataset, baseline model, implementation and experiment results are discussed.

Dataset We choose the standard WMT 2014 English-German dataset[2], which consists of approximately 4.5 million sentence pairs.

Baseline The baseline is the Transformer-base model proposed by [37].

Implementation details Build on top of the baseline model, we modify its self-attention block by incorporating RoPE, turning it into the RoFormer model. We replicate the setup of [37] for English-to-German translation with vocabulary of 37k based on a joint source and target byte pair encoding(BPE)[31]. During the evaluation, a single model is obtained by averaging the last 5 checkpoints. The result uses beam search with a beam size of 4 and length penalty 0.6. We implement the experiment in PyTorch in the fairseq toolkit (MIT License)[24]. Our model is optimized with the Adam optimizer using $\beta_1 = 0.9$, $\beta_2 = 0.98$, learning rate is increased linearly from $1e-7$ to $5e-4$ and then decayed proportionally to the inverse square root of the step number. Label smoothing with 0.1 is also adopted. We report the BLEU[25] score on test set as the final metric.

Results We train the baseline model and our RoFormer under the same settings and report the result in table 1. Our model achieves higher BLEU score comparing to the baseline Transformer.

Table 1: The proposed RoFormer achieves better BLEU score than Transformer[37] on the WMT 2014 English-to-German translation task[2].

Model	BLEU
Transformer-base[37]	27.3
RoFormer	**27.5**

4.2 Pre-training of language modeling

Our next experiment compares RoPE with the original sinusoidal position encoding implementation in BERT during pre-training.

Dataset and metrics We use the BookCorpus[45] and the Wikipedia Corpus[10] from Huggingface Datasets library (Apache License 2.0) for pre-training. The corpus is further split into train and validation sets at 8:2 ratio. We record the masked language-modeling(MLM) loss throughout training and report it as the comparing metric.

Baseline The well-known BERT[8] is adopted as our baseline model, specifically, we choose the BERT-base-uncased realization for comparison.

Implementation details For RoFormer, we replace the sinusoidal position encoding in the self-attention block of the baseline model with our proposed RoPE and realizes self-attention according to eq. (16). We train both BERT and RoFormer with batch size 64 and maximum sequence length of 512 for 100k steps. AdamW[22] is used as the optimizer with learning rate 1e-5.

Results The MLM loss during pre-training is shown on the left plot of fig. 2. Compare to the vanilla BERT, RoFormer experiences faster convergence.

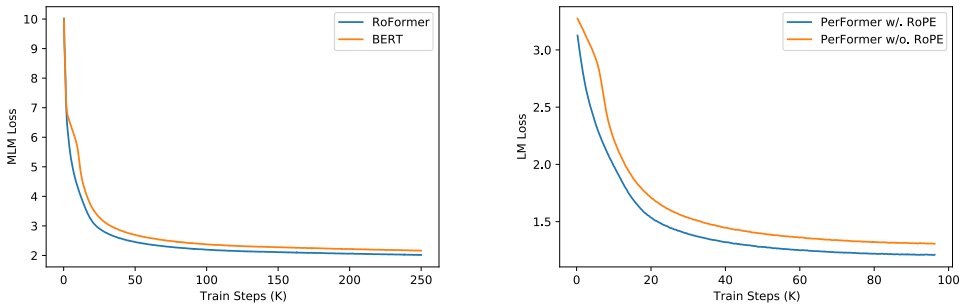

Figure 2: Evaluation of RoPE in language modeling pre-training. **Left**: training loss for BERT and RoFormer. **Right**: training loss for PerFormer with and without RoPE.

4.3 Fine-tuning on GLUE tasks

Following previous experiment, we fine tune the pre-trained RoFormer model across various GLUE tasks in order to evaluate its generalization ability towards downstream NLP tasks.

Dataset and metrics We look at several datasets from GLUE, i.e. MRPC[9], SST-2[35], QNLI[30], STS-B[1], QQP[4], MNLI[40]. We use F1-score for MRPC and QQP dataset, spearman correlation for STS-B, and accuracy for the remaining as the evaluation metrics.

Implementation details We use Huggingface Transformers library (Apache License 2.0)[41] to fine tune each of the aforementioned downstream tasks for 3 epochs, with a maximum sequence length of 512, batch size of 32 and learning rates 2,3,4,5e-5. As has been done in the baseline model work[8], we report the best averaged results on the evalulation set.

Results We table our results in table 2. The metrics show that RoFormer achieves comparable overall performance to BERT on various downstream tasks when fine tuned.

Table 2: Comparing RoFormer and BERT by fine tuning on downstream GLEU tasks.

Model	MRPC	SST-2	QNLI	STS-B	QQP	MNLI(m/mm)
BERT[8]	88.9	93.5	90.5	85.8	71.2	84.6/83.4
RoFormer	**89.5**	90.7	88.0	**87.0**	**86.4**	80.2/79.8

4.4 Performer with RoPE

Performer[5] provide an alternative attention mechanism, the linear attention, which is designed to avoid quadratic computation cost that scales with input sequence length. As discussed in section 3.3, the proposed RoPE can be easily implemented in the PerFormer model to realize relative position encoding while keeping its linearly scaled complexity in self-attention. We demonstrate its performance with pre-training task of language modeling.

Dataset We carry out tests on the Enwik8 dataset[23], which is from English Wikipedia that includes markup, special characters and text in other languages in addition to English text.

Implementation details Similar to experiments in previous sections, we incorporate RoPE into the 12 layer char-based PerFormer with 768 dimensions and 12 heads[2]. The loss curves are reported from the pre-training for models with and without RoPE under the same settings, i.e. learning rate 1e-4, batch size 128 and a fixed maximum sequence length of 1024, etc.

Results As shown on the right plot of fig. 2, substituting RoPE into Performer leads to to rapid convergence and lower loss under the same amount of the training steps. This improvement as well as the linear complexity makes Performer more attractive.

4.5 Evaluation on Chinese Data

In addition to experiments on English data, we show additional results on Chinese data. Besides experiment with pre-training, to illustrate the performance of RoFormer on long texts, we choose a task with most documents exceeding 512 characters for downstream evaluation and report the results.

Implementation In this experiment, we base our RoFormer model on WoBERT[36] and replace its absolute position embedding with our proposed RoPE. As a cross-comparison with other pre-trained Transformer-based models in Chinese, i.e. BERT[8], WoBERT[36], and NEZHA[39], we tabulate their tokenization level and position embedding information in table 3.

Table 3: Cross-comparison between our RoFormer and other pre-trained models on Chinese data. 'abs' and 'rel' annotates absolute position embedding and relative position embedding, respectively.

Model	BERT[8]	WoBERT[36]	NEZHA[39]	RoFormer
tokenization level	char	word	char	word
position embedding	abs.	abs.	rel.	RoPE

Pre-training We pre-train RoFormer on approximately 34GB data which consists of contents from Chinese Wikipedia, news, forums, etc. The pre-training is carried out in multiple stages with changing batch size and maximum input sequence length in order to adapt the model with various scenarios. As shown in table 4, the accuracy of RoFormer elevates with increasing upper bound of sequence length, which demonstrates the ability of RoFormer in dealing with long texts. We claim that this is attribute to the excellent generalizability of the proposed RoPE.

[2]For this experiment, we adopt code (MIT License) from `https://github.com/lucidrains/performer-pytorch`

Table 4: Pre-training strategy of RoFormer on Chinese dataset. The training procedure is divided into various consecutive stages. In each stage, we train the model with a specific combination of maximum sequence length and batch size.

stage	max seq. length	batch size	training steps	loss	accuracy
1	512	256	200k	1.73	65.0%
2	1536	256	12.5k	1.61	66.8%
3	256	256	120k	1.75	64.6%
4	128	512	80k	1.83	63.4%
5	1536	256	10k	1.58	67.4%
6	512	512	30k	1.66	66.2%

Downstream Tasks & Dataset We choose Chinese AI and Law 2019 Similar Case Matching (CAIL2019-SCM)[42] dataset to illustrate the ability of RoFormer in dealing with long texts, i.e. semantic text matching. CAIL2019-SCM contains 8964 triplets of cases published by the Supreme People's Court of China. The input triplet, denoted as (A, B, C), are fact descriptions of three cases. The task is to predict whether the pair (A, B) is closer than (A, C) under a predefined similarity measure. Due to the background of CAIL2019-SCM dataset, most of its documents contain more than 512 characters, which is challenging for existing methods to capture document level information. The raw dataset is split into train, validation and test set at a ratio of 6:2:2.

Results We apply the pre-trained RoFormer model to CAIL2019-SCM with different input lengths. The model is compared with the pre-trained BERT and WoBERT model on the same pre-training data, as shown in table 5. With short text cut-offs, i.e. 512, the result from RoFormer is comparable to WoBERT and is slightly better than the BERT implementation. However, when increase the maximum input text length to 1024, RoFormer outperforms WoBERT by an absolute improvement of 1.5%.

Table 5: Experiment results on CAIL2019-SCM task. Numbers in the first column denote the maximum cut-off sequence length. The results are presented in terms of percent accuracy.

Model	validation	test
BERT-512	64.13%	67.77%
WoBERT-512	64.07%	68.10%
RoFormer-512	64.13%	68.29%
RoFormer-1024	**66.07%**	**69.79%**

5 Conclusions

Limitations of the work Although we provide theoretical groundings as well as promising experimental justifications, our method is limited by following facts: 1) Despite the fact that we mathematically format the relative position relations as rotations under 2D sub-spaces, there lacks of thorough explanations on why it converges faster than baseline models that incorporates other position encoding strategies. 2) Although we have proved that our model has favourable property of long-term decay for intern-token products(in section 3.3), which is similar to existing position encoding mechanisms, our model shows superior performance on long texts than peer models, we have not come up with a faithful explanation.

Potential negative social impacts Like many other Transformer-based models, pre-training our model requires certain amount of hardware power consumption that might indirectly cause extra carbon emissions.

Conclusion In this work, we proposed a new position embedding method that incorporates explicit relative position dependency in self-attention to enhance the performance of transformer architectures. Our theoretical analysis indicates that relative position can be naturally formulated using vector production in self-attention, with absolution position information being encoded through rotation

matrix. In addition, we mathematically illustrated the advantageous properties of the proposed method when applied in transformer. Finally, experiment on both English and Chinese data demonstrate that our method encourages faster convergence in pre-training. Additional tests also indicate the superior performance of our model when applied to tasks with long texts.

References

[1] Hussein Al-Natsheh. Udl at semeval-2017 task 1: Semantic textual similarity estimation of english sentence pairs using regression model over pairwise features. 08 2017.

[2] Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Alevs Tamchyna. Findings of the 2014 workshop on statistical machine translation. pages 12–58, 06 2014.

[3] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6572–6583, 2018.

[4] Z. Chen, H. Zhang, and L. Zhang, X.and Zhao. Quora question pairs., 2018.

[5] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, A. Gane, Tamás Sarlós, Peter Hawkins, J. Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy J. Colwell, and Adrian Weller. Rethinking attention with performers. *ArXiv*, abs/2009.14794, 2020.

[6] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*, 2020.

[7] Zihang Dai, Z. Yang, Yiming Yang, J. Carbonell, Quoc V. Le, and R. Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. In *ACL*, 2019.

[8] J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

[9] William B. Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005.

[10] Wikimedia Foundation. Wikimedia downloads, `https://dumps.wikimedia.org`, 2021.

[11] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *International Conference on Machine Learning*, pages 1243–1252. PMLR, 2017.

[12] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. *ArXiv*, abs/2006.03654, 2020.

[13] Max Horn, Kumar Shridhar, Elrich Groenewald, and Philipp FM Baumann. Translational equivariance in kernelizable attention. *arXiv preprint arXiv:2102.07680*, 2021.

[14] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, I. Simon, C. Hawthorne, Andrew M. Dai, M. Hoffman, M. Dinculescu, and D. Eck. Music transformer. *arXiv: Learning*, 2018.

[15] Zhiheng Huang, Davis Liang, Peng Xu, and Bing Xiang. Improve transformer models with better relative position embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3327–3335, Online, November 2020. Association for Computational Linguistics.

[16] Md. Amirul Islam, Sen Jia, and Neil D. B. Bruce. How much position information do convolutional neural networks encode? *ArXiv*, abs/2001.08248, 2020.

[17] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, pages 5156–5165. PMLR, 2020.

[18] Guolin Ke, Di He, and T. Liu. Rethinking positional encoding in language pre-training. *ArXiv*, abs/2006.15595, 2020.

[19] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2020.

[20] Xuanqing Liu, Hsiang-Fu Yu, Inderjit S. Dhillon, and Cho-Jui Hsieh. Learning to encode position for transformer with continuous dynamical model. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 6327–6335. PMLR, 2020.

[21] Antoine Liutkus, Ondrej Cifka, Shih-Lun Wu, Umut Simsekli, and Yang. Relative positional encoding for transformers with linear complexity. *arXiv preprint arXiv:2105.08399*, 2021.

[22] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. *arXiv e-prints*, page arXiv:1711.05101, November 2017.

[23] Matt Mahoney. Large text compression benchmark, `http://www.mattmahoney.net/dc/text.html`, 2006.

[24] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. pages 48–53, 01 2019.

[25] Kishore Papineni, Salim Roukos, Todd Ward, and Wei Jing Zhu. Bleu: a method for automatic evaluation of machine translation. 10 2002.

[26] Ankur P. Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. In *EMNLP*, 2016.

[27] A. Radford and Karthik Narasimhan. Improving language understanding by generative pre-training. 2018.

[28] A. Radford, Jeffrey Wu, R. Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[29] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, W. Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020.

[30] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. pages 2383–2392, 01 2016.

[31] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. 08 2015.

[32] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. In *NAACL-HLT*, 2018.

[33] Zhuoran Shen, Mingyuan Zhang, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Efficient attention: Attention with linear complexities. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3531–3539, 2021.

[34] Amapreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. 04 2018.

[35] Richard Socher, A. Perelygin, J.Y. Wu, J. Chuang, C.D. Manning, A.Y. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. *EMNLP*, 1631:1631–1642, 01 2013.

[36] Jianlin Su. Wobert: Word-based chinese bert model - zhuiyiai. Technical report, 2020.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, L ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[38] Benyou Wang, Donghao Zhao, Christina Lioma, Qiuchi Li, Peng Zhang, and Jakob Grue Simonsen. Encoding word order in complex embeddings. In *International Conference on Learning Representations*, 2020.

[39] Victor Junqiu Wei, Xiaozhe Ren, Xiaoguang Li, Wenyong Huang, Yi Liao, Yasheng Wang, Jiashu Lin, Xin Jiang, Xiao Chen, and Qun Liu. Nezha: Neural contextualized representation for chinese language understanding. 08 2019.

[40] Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. pages 1112–1122, 01 2018.

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

[42] Chaojun Xiao, Haoxi Zhong, Zhipeng Guo, Cunchao Tu, Zhiyuan Liu, Maosong Sun, Tianyang Zhang, Xianpei Han, Zhen hu, Heng Wang, and Jianfeng Xu. Cail2019-scm: A dataset of similar case matching in legal domain. 11 2019.

[43] Z. Yang, Zihang Dai, Yiming Yang, J. Carbonell, R. Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*, 2019.

[44] Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? In *International Conference on Learning Representations*, 2020.

[45] Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *arXiv preprint arXiv:1506.06724*, 2015.

A Derivation of RoPE under 2D

Under the case of $d = 2$, we consider two word embedding vectors $\boldsymbol{x}_q$, $\boldsymbol{x}_k$ corresponds to query and key and their position m and n, respectively. According to eq. (1), their position-encoded counterparts are:

$$\begin{aligned} \boldsymbol{q}_m &= f_q(\boldsymbol{x}_q, m) \\ \boldsymbol{k}_n &= f_k(\boldsymbol{x}_k, n) \end{aligned} \tag{20}$$

Here the subscripts of $\boldsymbol{q}_m$, $\boldsymbol{k}_n$ indicates the encoded position information. Assume there exists function g that defines inner product between vectors produced by $f_{\{q,k\}}$:

$$\boldsymbol{q}_m^\intercal \boldsymbol{k}_n = \langle f_q(\boldsymbol{x}_m, m), f_k(\boldsymbol{x}_n, n) \rangle = g(\boldsymbol{x}_m, \boldsymbol{x}_n, n - m) \tag{21}$$

We further ask below initial condition to be satisfied:

$$\begin{aligned} \boldsymbol{q} &= f_q(\boldsymbol{x}_q, 0) \\ \boldsymbol{k} &= f_k(\boldsymbol{x}_k, 0) \end{aligned} \tag{22}$$

which denotes the vectors with empty position information encoded. With above settings, we manage to find a solution of f_q, f_k.

First, we take advantage of the geometric meaning of vector in 2D and its complex counter part, decompose functions in eqs. (20) and (21) into:

$$\begin{aligned} f_q(\boldsymbol{x}_q, m) &= R_q(\boldsymbol{x}_q, m) e^{i\Theta_q(\boldsymbol{x}_q, m)} \\ f_k(\boldsymbol{x}_k, n) &= R_k(\boldsymbol{x}_k, n) e^{i\Theta_k(\boldsymbol{x}_k, n)} \\ g(\boldsymbol{x}_q, \boldsymbol{x}_k, n - m) &= R_g(\boldsymbol{x}_q, \boldsymbol{x}_k, n - m) e^{i\Theta_g(\boldsymbol{x}_q, \boldsymbol{x}_k, n - m)} \end{aligned} \tag{23}$$

Where R_f, R_g and Θ_f, Θ_g are the radical and angular components for $f_{\{q,k\}}$ and g, respectively. Plug them into eq. (21), we get relation:

$$\begin{aligned} R_q(\boldsymbol{x}_q, m) R_k(\boldsymbol{x}_k, n) &= R_g(\boldsymbol{x}_q, \boldsymbol{x}_k, n - m) \\ \Theta_k(\boldsymbol{x}_k, n) - \Theta_q(\boldsymbol{x}_q, m) &= \Theta_g(\boldsymbol{x}_q, \boldsymbol{x}_k, n - m) \end{aligned} \tag{24}$$

with the corresponding initial condition as:

$$\begin{aligned} \boldsymbol{q} = \|\boldsymbol{q}\| e^{i\theta_q} &= R_q(\boldsymbol{x}_q, 0) e^{i\Theta_q(\boldsymbol{x}_q, 0)} \\ \boldsymbol{k} = \|\boldsymbol{k}\| e^{i\theta_k} &= R_k(\boldsymbol{x}_k, 0) e^{i\Theta_k(\boldsymbol{x}_k, 0)} \end{aligned} \tag{25}$$

Where $\|\boldsymbol{q}\|$, $\|\boldsymbol{k}\|$ and θ_q, θ_k are the radial and angular part of $\boldsymbol{q}$ and $\boldsymbol{k}$ on the 2D plane.

Next, we set $m = n$ in eq. (24) and take into account initial conditions in eq. (25):

$$R_q(\boldsymbol{x}_q, m) R_k(\boldsymbol{x}_k, m) = R_q(\boldsymbol{x}_q, \boldsymbol{x}_k, 0) = R_k(\boldsymbol{x}_q, 0) R_k(\boldsymbol{x}_k, 0) = \|\boldsymbol{q}\| \|\boldsymbol{k}\| \tag{26a}$$

$$\Theta_k(\boldsymbol{x}_k, m) - \Theta_q(\boldsymbol{x}_q, m) = \Theta_g(\boldsymbol{x}_q, \boldsymbol{x}_k, 0) = \|\Theta_k(\boldsymbol{x}_k, 0) - \Theta_q(\boldsymbol{x}_q, 0)\| = \|\theta_k - \theta_q\| \tag{26b}$$

On one hand, from eq. (26a), we have a straightforward solution of R_f:

$$\begin{aligned} R_q(\boldsymbol{x}_q, m) &= R_q(\boldsymbol{x}_q, 0) = \|\boldsymbol{q}\| \\ R_k(\boldsymbol{x}_k, n) &= R_k(\boldsymbol{x}_k, 0) = \|\boldsymbol{k}\| \\ R_g(\boldsymbol{x}_q, \boldsymbol{x}_k, n - m) &= R_g(\boldsymbol{x}_q, \boldsymbol{x}_k, 0) = \|\boldsymbol{q}\| \|\boldsymbol{k}\| \end{aligned} \tag{27}$$

Which means the radial functions R_q, R_k and R_g are functions independent to position information.

On the other hand, according to eq. (26b), notice $\Theta_q(\boldsymbol{x}_q, m) - \theta_q = \Theta_k(\boldsymbol{x}_k, m) - \theta_k$ indicates that the angular functions does not dependent on query and key, we set them to $\Theta_f := \Theta_q = \Theta_k$ and term $\Theta_f(\boldsymbol{x}_{\{q,k\}}, m) - \theta_{\{q,k\}}$ is a function of position m and is independent of word embedding $\boldsymbol{x}_{\{q,k\}}$, we denote it as $\phi(m)$, yielding:

$$\Theta_f(\boldsymbol{x}_{\{q,k\}}, m) = \phi(m) + \theta_{\{q,k\}} \tag{28}$$

Further, by plugging in $n = m + 1$ in eq. (24) and consider above equation, we have:

$$\phi(m+1) - \phi(m) = \Theta_g(\boldsymbol{x}_q, \boldsymbol{x}_k, 1) + \theta_q - \theta_k \tag{29}$$

Since the RHS of the equation is a constant irrelevant to m, function $\phi(m)$ with continuous integer inputs produce an arithmetic progression. Thus, it is straightforward to get:

$$\phi(m) = m\theta + \gamma \tag{30}$$

Where $\theta, \gamma \in \mathbb{R}$ are constants and θ is non-zero.

To summarize our solutions from Equations (27) to (30):

$$\begin{aligned} f_q(\boldsymbol{x}_q, m) &= \|\boldsymbol{q}\| e^{i\theta_q + m\theta + \gamma} = \boldsymbol{q} e^{i(m\theta+\gamma)} \\ f_k(\boldsymbol{x}_k, n) &= \|\boldsymbol{k}\| e^{i\theta_k + n\theta + \gamma} = \boldsymbol{k} e^{i(n\theta+\gamma)} \end{aligned} \tag{31}$$

Finally, notice that we haven't set any constrains to functions in eq. (22), thus $f_q(\boldsymbol{x}_m, 0)$ and $f_k(\boldsymbol{x}_n, 0)$ are left to choose freely. To make our result be comparable to eq. (3), here we simply set:

$$\begin{aligned} \boldsymbol{q} &= f_q(\boldsymbol{x}_m, 0) = \boldsymbol{W}_q \boldsymbol{x}_n \\ \boldsymbol{k} &= f_k(\boldsymbol{x}_n, 0) = \boldsymbol{W}_k \boldsymbol{x}_n \end{aligned} \tag{32}$$

With above and simply set $\gamma = 0$ in eq. (31), the ultimate solution is:

$$\begin{aligned} f_q(\boldsymbol{x}_m, m) &= (\boldsymbol{W}_q \boldsymbol{x}_m) e^{im\theta} \\ f_k(\boldsymbol{x}_n, n) &= (\boldsymbol{W}_k \boldsymbol{x}_n) e^{in\theta} \end{aligned} \tag{33}$$

B Computational efficient realization of rotary matrix multiplication

Taking the advantage of the sparsity of $\boldsymbol{R}^d_{\Theta,m}$ in eq. (15), a more computational efficient realization of multiplication operation between matrix R^d_Θ and vector $\boldsymbol{x} \in \mathbb{R}^d$ is:

$$\boldsymbol{R}^d_{\Theta,m}\boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_{d-1} \\ x_d \end{pmatrix} \otimes \begin{pmatrix} \cos m\theta_1 \\ \cos m\theta_1 \\ \cos m\theta_2 \\ \cos m\theta_2 \\ \vdots \\ \cos m\theta_{d/2} \\ \cos m\theta_{d/2} \end{pmatrix} + \begin{pmatrix} -x_2 \\ x_1 \\ -x_4 \\ x_3 \\ \vdots \\ -x_{d-1} \\ x_d \end{pmatrix} \otimes \begin{pmatrix} \sin m\theta_1 \\ \sin m\theta_1 \\ \sin m\theta_2 \\ \sin m\theta_2 \\ \vdots \\ \sin m\theta_{d/2} \\ \sin m\theta_{d/2} \end{pmatrix} \tag{34}$$

C Long-term decay of RoPE

We can group entries of vectors $\boldsymbol{q} = \boldsymbol{W}_q \boldsymbol{x}_m$ and $\boldsymbol{k} = \boldsymbol{W}_k \boldsymbol{x}_n$ in pairs, and the inner product of RoPE in 16 can be written as complex number multiplication.

$$(\boldsymbol{R}^d_{\Theta,m}\boldsymbol{W}_q\boldsymbol{x}_m)^\intercal (\boldsymbol{R}^d_{\Theta,n}\boldsymbol{W}_k\boldsymbol{x}_n) = \mathrm{Re}\left[\sum_{i=0}^{d/2-1} \boldsymbol{q}_{[2i:2i+1]} \boldsymbol{k}^*_{[2i:2i+1]} e^{i(m-n)\theta_i}\right] \tag{35}$$

where $\boldsymbol{q}_{[2i:2i+1]}$ represents the $2i^{th}$ to $(2i+1)^{th}$ entries of $\boldsymbol{q}$. Denote $h_i = \boldsymbol{q}_{[2i:2i+1]} \boldsymbol{k}^*_{[2i:2i+1]}$ and $S_j = \sum_{i=0}^{j-1} e^{i(m-n)\theta_i}$, and let $h_{d/2} = 0$ and $S_0 = 0$, we can rewrite the summation using Abel transformation

$$\sum_{i=0}^{d/2-1} \boldsymbol{q}_{[2i:2i+1]} \boldsymbol{k}^*_{[2i:2i+1]} e^{i(m-n)\theta_i} = \sum_{i=0}^{d/2-1} h_i (S_{i+1} - S_i) = -\sum_{i=0}^{d/2-1} S_{i+1}(h_{i+1} - h_i) \tag{36}$$

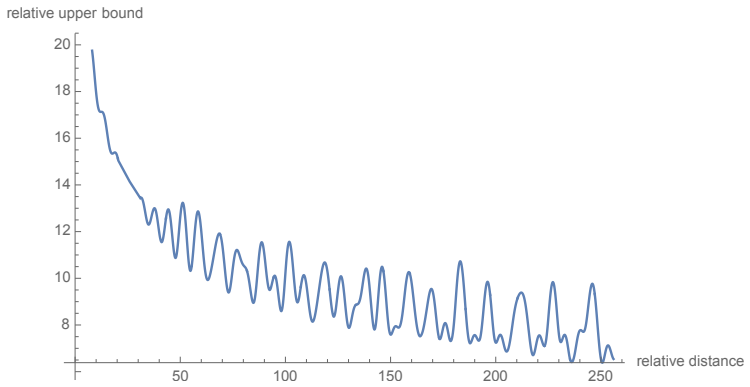

Figure 3: Long-term decay of RoPE.

So we have

$$\begin{aligned}\left|\sum_{i=0}^{d/2-1} \boldsymbol{q}_{[2i:2i+1]}\boldsymbol{k}^*_{[2i:2i+1]}e^{i(m-n)\theta_i}\right| &= \left|\sum_{i=0}^{d/2-1} S_{i+1}(h_{i+1}-h_i)\right| \\ &\leq \sum_{i=0}^{d/2-1} |S_{i+1}||(h_{i+1}-h_i)| \\ &\leq \left(\max_i |h_{i+1}-h_i|\right)\sum_{i=0}^{d/2-1}|S_{i+1}|\end{aligned} \tag{37}$$

the value of $\frac{1}{d/2}\sum_{i=1}^{d/2}|S_i|$ decay with the relative distance $m-n$ increases by setting $\theta_i = 10000^{-2i/d}$, as shown in fig. 3.