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

Ahmed Murtadha
Zhuiyi Technology Co., Ltd.
Shenzhen
mengjiayi@wezhuiyi.com

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

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

August 10, 2022

ABSTRACT

Position encoding recently has shown effective in the transformer architecture. It enables valuable supervision for dependency modeling between elements at different positions of the sequence. In this paper, we first investigate various methods to integrate positional information into the learning process of transformer-based language models. Then, we propose a novel method named Rotary Position Embedding(RoPE) to effectively leverage the positional information. Specifically, the proposed RoPE encodes the absolute position with a rotation matrix and meanwhile incorporates the explicit relative position dependency in self-attention formulation. Notably, RoPE enables valuable properties, including the flexibility of sequence length, decaying inter-token dependency with increasing relative distances, and the capability of equipping the linear self-attention with relative position encoding. Finally, we evaluate the enhanced transformer with rotary position embedding, also called RoFormer, on various long text classification benchmark datasets. Our experiments show that it consistently overcomes its alternatives. Furthermore, we provide a theoretical analysis to explain some experimental results. RoFormer is already integrated into Huggingface: `https://huggingface.co/docs/transformers/model_doc/roformer`.

Keywords Pre-trained Language Models · Position Information Encoding · Pre-training · Natural Language Processing.

1 Introduction

The sequential order of words is of great value to natural language understanding. Recurrent neural networks (RRNs) based models encode tokens' order by recursively computing a hidden state along the time dimension. Convolution neural networks (CNNs) based models (CNNs) Gehring et al. [2017] were typically considered position-agnostic, but recent work Islam et al. [2020] has shown that the commonly used padding operation can implicitly learn position information. Recently, the pre-trained language models (PLMs), which were built upon the transformer Vaswani et al. [2017], have achieved the state-of-the-art performance of various natural language processing (NLP) tasks, including context representation learning Devlin et al. [2019], machine translation Vaswani et al. [2017], and language modeling Radford et al. [2019], to name a few. Unlike, RRNs and CNNs-based models, PLMs utilize the self-attention mechanism to semantically capture the contextual representation of a given corpus. As a consequence, PLMs achieve a significant improvement in terms of parallelization over RNNs and improve the modeling ability of longer intra-token relations compared to CNNs[1].

[1] A stack of multiple CNN layers can also capture longer intra-token relation, here we only consider single layer setting.

It is noteworthy that the self-attention architecture of the current PLMs has shown to be position-agnostic Yun et al. [2020]. Following this claim, various approaches have been proposed to encode the position information into the learning process. On one side, generated absolute position encoding through a pre-defined function Vaswani et al. [2017] was added to the contextual representations, while a trainable absolute position encoding Gehring et al. [2017], Devlin et al. [2019], Lan et al. [2020], Clark et al. [2020], Radford et al. [2019], Radford and Narasimhan [2018]. On the other side, the previous work Parikh et al. [2016], Shaw et al. [2018], Huang et al. [2018], Dai et al. [2019], Yang et al. [2019], Raffel et al. [2020], Ke et al. [2020], He et al. [2020], Huang et al. [2020] focuses on relative position encoding, which typically encodes the relative position information into the attention mechanism. In addition to these approaches, the authors of Liu et al. [2020] have proposed to model the dependency of position encoding from the perspective of Neural ODE Chen et al. [2018a], and the authors of Wang et al. [2020] have proposed to model the position information in complex space. Despite the effectiveness of these approaches, they commonly add the position information to the context representation and thus render them unsuitable for the linear self-attention architecture.

In this paper, we introduce a novel method, namely Rotary Position Embedding(RoPE), to leverage the positional information into the learning process of PLMS. Specifically, RoPE encodes the absolute position with a rotation matrix and meanwhile incorporates the explicit relative position dependency in self-attention formulation. Note that the proposed RoPE is prioritized over the existing methods through valuable properties, including the sequence length flexibility, decaying inter-token dependency with increasing relative distances, and the capability of equipping the linear self-attention with relative position encoding. Experimental results on various long text classification benchmark datasets show that the enhanced transformer with rotary position embedding, namely RoFormer, can give better performance compared to baseline alternatives and thus demonstrates the efficacy of the proposed RoPE.

In brief, our contributions are three-folds as follows:

- We investigated the existing approaches to the relative position encoding and found that they are mostly built based on the idea of the decomposition of adding position encoding to the context representations. We introduce a novel method, namely Rotary Position Embedding(RoPE), to leverage the positional information into the learning process of PLMS. The key idea is 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 kindly argue that previous relative position encoding-based approaches are not compatible with linear self-attention.
- We evaluate the proposed RoFormer on various long text benchmark datasets. Our experiments show that it consistently achieves better performance compared to its alternatives. Some experiments with pre-trained language models are available on GitHub: `https://github.com/ZhuiyiTechnology/roformer`.

The remaining of the paper is organized as follows. We establish a formal description of the position encoding problem in self-attention architecture and revisit previous works in Section **(2)**. We then describe the rotary position encoding (RoPE) and study its properties in Section **(3)**. We report experiments in Section **(4)**. Finally, we conclude this paper in Section **(5)**.

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 $\mathbb{S}_N$ is 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 $\boldsymbol{q}_m, \boldsymbol{k}_n$ and $\boldsymbol{v}_n$ incorporate the m^{th} and n^{th} positions through f_q, f_k and f_v, respectively. The query and key values are then used to compute the attention weights, while the output is computed as the weighted sum over the value

representation.

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

The existing approaches of transformer-based position encoding mainly focus on choosing a suitable function to form Equation (1).

2.2 Absolute position embedding

A typical choice of Equation (1) is

Pos Embedding

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

where $\boldsymbol{p}_i \in \mathbb{R}^d$ is a d-dimensional vector depending of the position of token $\boldsymbol{x}_i$. Previous work Devlin et al. [2019], Lan et al. [2020], Clark et al. [2020], Radford et al. [2019], Radford and Narasimhan [2018] introduced the use of a set of trainable vectors $\boldsymbol{p}_i \in \{\boldsymbol{p}_t\}_{t=1}^{L}$, where L is the maximum sequence length. The authors of Vaswani et al. [2017] have proposed to generate $\boldsymbol{p}_i$ using the sinusoidal function.

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

in which $\boldsymbol{p}_{i,2t}$ is the $2t^{th}$ element of the d-dimensional vector $\boldsymbol{p}_i$. In the next section, we show that our proposed RoPE is related to this intuition from the sinusoidal function perspective. However, instead of directly adding the position to the context representation, RoPE proposes to incorporate the relative position information by multiplying with the sinusoidal functions.

2.3 Relative position embedding

The authors of Shaw et al. [2018] applied different settings of Equation (1) as following:

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

Linear layer, "M" is the position of Query

"n" is the position of whose effect we want to find. Distance is clipped

where $\tilde{\boldsymbol{p}}_r^k, \tilde{\boldsymbol{p}}_r^v \in \mathbb{R}^d$ are trainable relative position embeddings. Note that $r = \text{clip}(m-n, r_{\min}, r_{\max})$ 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. Keeping the form of Equation (3), the authors Dai et al. [2019] have proposed to decompose $\boldsymbol{q}_m^\intercal \boldsymbol{k}_n$ of Equation (2) as

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

the key idea is to replace the absolute position embedding $\boldsymbol{p}_n$ with its sinusoid-encoded relative counterpart $\tilde{\boldsymbol{p}}_{m-n}$, while the absolute position $\boldsymbol{p}_m$ in the third and fourth term with two trainable vectors $\mathbf{u}$ and $\mathbf{v}$ 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 noteworthy that the position information in the value term is removed by setting $f_v(\boldsymbol{x}_j) := \boldsymbol{W}_v \boldsymbol{x}_j$. Later work Raffel et al. [2020], He et al. [2020], Ke et al. [2020], Huang et al. [2020] followed these settings by only encoding the relative position information into the attention weights. However, the authors of Raffel et al. [2020] reformed Equation (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. The authors of Ke et al. [2020] investigated the middle two terms of Equation (6) and found little correlations between absolute positions and words. The authors of Raffel et al. [2020] proposed to model a pair of words or positions using different projection matrices.

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

The authors of He et al. [2020] argued that the relative positions of two tokens could only be fully modeled using the middle two terms of Equation (6). As a consequence, the absolute position embeddings $\boldsymbol{p}_m$ and $\boldsymbol{p}_n$ were simply replaced with the relative position embeddings $\tilde{\boldsymbol{p}}_{m-n}$:

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

A comparison of the four variants of the relative position embeddings Radford and Narasimhan [2018] has shown that the variant similar to Equation (10) is the most efficient among the other three. Generally speaking, all these approaches attempt to modify Equation (6) based on the decomposition of Equation (3) under the self-attention settings in Equation (2), which was originally proposed in Vaswani et al. [2017]. They commonly introduced to directly add the position information to the context representations. Unlikely, our approach aims to derive the relative position encoding from Equation (1) under some constraints. Next, we show that the derived approach is more interpretable by incorporating relative position information with the rotation of context representations.

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

Transformer-based language modeling usually leverages the position information of individual tokens through a self-attention mechanism. As can be observed in Equation (2), $\boldsymbol{q}_m^\intercal \boldsymbol{k}_n$ typically enables knowledge conveyance between tokens at different positions. In order to incorporate relative position information, we require the inner product of query $\boldsymbol{q}_m$ and key $\boldsymbol{k}_n$ to 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 that the inner product encodes position information only in the relative form:

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

The ultimate goal is to find an equivalent encoding mechanism to solve the functions $f_q(\boldsymbol{x}_m, m)$ and $f_k(\boldsymbol{x}_n, n)$ to conform the aforementioned relation.

3.2 Rotary position embedding

3.2.1 A 2D case

We begin with a simple case with a dimension $d = 2$. Under these settings, we make use of the geometric property of vectors on a 2D plane and its complex form to prove (refer Section **(3.4.1)** for more details) that a solution to our formulation Equation (11) is:

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

Rotation is done BEFORE the dot product

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 a multiplication matrix:

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

where $(x_m^{(1)}, x_m^{(2)})$ is $\boldsymbol{x}_m$ expressed in the 2D coordinates. Similarly, g can be viewed as a matrix and thus enables the solution of formulation in Section **(3.1)** under the 2D case. Specifically, incorporating the relative position embedding is straightforward: simply rotate the affine-transformed word embedding vector by amount of angle multiples of its position index and thus interprets the intuition behind *Rotary Position Embedding*.

3.2.2 General form

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

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

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 $\Theta = \{\theta_i = 10000^{-2(i-1)/d}, i \in [1, 2, ..., d/2]\}$. A graphic illustration of RoPE is shown in Figure **(1)**. Applying our RoPE to self-attention in Equation (2), we obtain:

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

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

In contrast to the additive nature of position embedding method adopted in the previous works, i.e., Equations (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.

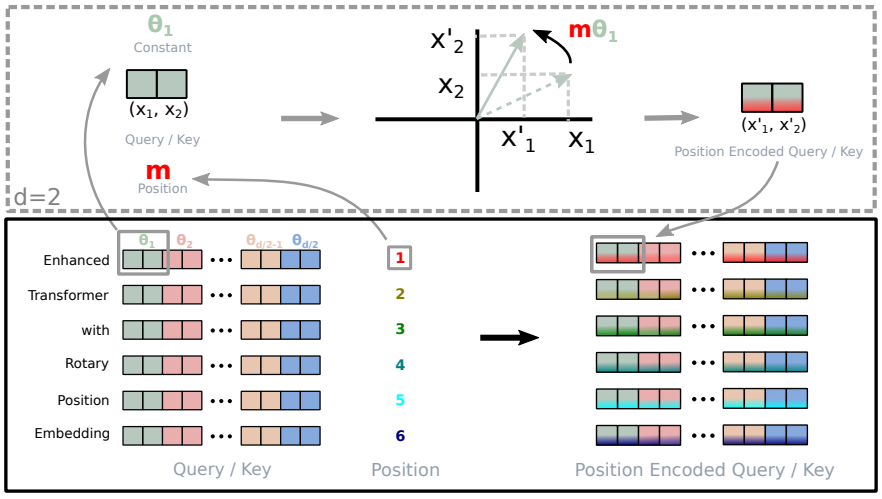

Figure 1: Implementation of Rotary Position Embedding(RoPE).

3.3 Properties of RoPE

Long-term decay: Following Vaswani et al. [2017], we set $\theta_i = 10000^{-2i/d}$. One can prove that this setting provides a long-term decay property (refer to Section **(3.4.3)** 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 a long relative distance should have less connection.

RoPE with linear attention: The self-attention can be rewritten in a more 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 $\mathrm{sim}(\boldsymbol{q}_m, \boldsymbol{k}_n) = \exp(\boldsymbol{q}_m^\intercal \boldsymbol{k}_n / \sqrt{d})$. Note that the original self-attention should compute the inner product of query and key for every pair of tokens, which has a quadratic complexity $\mathbb{O}(N^2)$. Follow Katharopoulos et al. [2020], the 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. The authors of Katharopoulos et al. [2020] have 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. A softmax function is used in Shen et al. [2021] 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 attention, 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 attention by multiplying the rotation matrix with the outputs of the non-negative functions.

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

Norm is unchanged as everything was rotated by the same amount

It is noteworthy 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 kindly argue that the computation can still model the importance of values.

3.4 Theoretical Explanation

3.4.1 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:

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

where the subscripts of $\boldsymbol{q}_m$ and $\boldsymbol{k}_n$ indicate the encoded positions information. Assume that there exists a function g that defines the 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 require 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 can be read as the vectors with empty position information encoded. Given these settings, we attempt 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 Equations (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 Equation (21), we get the 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:

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