CoCA: Fusing Position Embedding with Collinear Constrained Attention in Transformers for Long Context Window Extending

Shiyi Zhu[1*], **Jing Ye**[1*], **Wei Jiang**[1], **Siqiao Xue**[1], **Qi Zhang**[1], **Yifan Wu**[1,2], **Jianguo Li**[1†]

[1]Ant Group, Hangzhou, China

[2]Peking University, Beijing, China

zhushiyi.zsy@antgroup.com; yejing2022@ia.ac.cn;

{shouzhi.jw, siqiao.xsq, yuqi.zq, lijg.zero}@antgroup.com; yifanwu@pku.edu.cn

Abstract

Self-attention and position embedding are two crucial modules in transformer-based Large Language Models (LLMs). However, the potential relationship between them is far from well studied, especially for long context window extending. In fact, anomalous behaviors that hinder long context extrapolation exist between Rotary Position Embedding (RoPE) and vanilla self-attention. Incorrect initial angles between Q and K can cause misestimation in modeling rotary position embedding of the closest tokens. To address this issue, we propose **Co**llinear **C**onstrained **A**ttention mechanism, namely CoCA. Specifically, we enforce a collinear constraint between Q and K to seamlessly integrate RoPE and self-attention. While only adding minimal computational and spatial complexity, this integration significantly enhances long context window extrapolation ability. We provide an optimized implementation, making it a drop-in replacement for any existing transformer-based models. Extensive experiments demonstrate that CoCA excels in extending context windows. A CoCA-based GPT model, trained with a context length of 512, can extend the context window up to 32K (60×) without any fine-tuning. Additionally, incorporating CoCA into LLaMA-7B achieves extrapolation up to 32K within a training length of only 2K. Our code is publicly available at: https://github.com/codefuse-ai/Collinear-Constrained-Attention

1 Introduction

In the seminal work of Transformer (Vaswani et al., 2017), it claims the ability of "extrapolating to sequence length longer than the ones encountered during training". This is an ideal hypothesis, but actually not work in practice for vanilla Transformer. Several subsequent works, collectively known as long context extrapolation, have delved into exploring the capabilities of large language models (LLMs) trained within the range of $[1, N-1]$ to effectively extend the testing sequence $\geq N$.

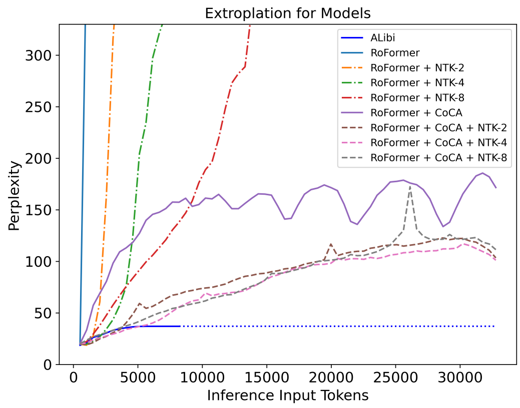


Figure 1: Perplexity evaluation on 100 PG-19 documents with a sliding window strategy (Stride = 512). The perplexity of RoFormer (Su et al., 2024) sharply exceeds 1000 beyond its training length, while CoCA maintains a low plateau even at 60 × its training length. ALibi (Press et al., 2022) encounters Out of Memory (OOM) issues for input $N_{max} > 8000$ due to flash-attention (Dao et al., 2022) incompatibility, we suppose it maintains perplexity for $N_{max} > 8000$.

Existing studies mainly focus on either the attention kernel (Beltagy et al., 2020; Ding et al., 2023; Han et al., 2023) or position embedding (Huang et al., 2023). However, they often neglect the intrinsic relationship between these two key components. Attention bias is an alternative to the explicit encoding of positional information. ALibi (Press et al., 2022) and KERPLE (Chi et al., 2022), incorporate heuristic and compositional triangle kernel-based negative causal attention bias, respectively. While these approaches effectively maintain low perplexity, they fall short in capturing long-range dependencies due to introducing local hypotheses to context tokens. Another branch of methods involve simply scaling Rotary Position Embedding (RoPE) (Su et al., 2024) to extrapolate the inference context length with minimal or

[*]Equal Contribution. This work was completed during Jing Ye's internship at Ant Group.

[†]Corresponding Author

no fine-tuning. For instance, Position Interpolation (PI) (Chen et al., 2023) employs linear scaling on each position number from n to n/k, where k is the extrapolation ratio. NTK-aware Scaled RoPE (bloc97, 2023) and dynamic-NTK (Emozilla, 2023) combine high-frequency extrapolation and low-frequency interpolation. They scale the basis in RoPE upon the sequence length to adapt to the unseen position indices. Furthermore, CLEX (Chen et al., 2024) optimizes extrapolation capability by modeling the continuous dynamics over different scaling factors. However, these methods primarily alleviate the problem of modeling the rotation angles in out-of-distribution positions, without recognizing the intrinsic correlation between attention matrices and rotation angles. Therefore, these methods still suffer from a limited context window extending ratio.

Here, we present a new perspective on the relationship between position embedding (with a focus on RoPE) and the self-attention mechanism. RoPE utilizes a rotation matrix to encode absolute positions while simultaneously incorporating explicit relative position dependencies within the self-attention formulation (Su et al., 2024). It is designed based on the relative angular difference between the queries (Q) and keys (K). However, there is an initial angle between Q and K in 2-D subspace. Incorrect initial angles between Q and K introduce additional rotary borders, resulting in misestimation in modeling the rotary position embedding of the closest tokens. This rotational anomaly severely compromises the performance of long context extrapolation methods.

To address this undesirable behavior, we propose an innovative architecture called **Co**llinear **C**onstrained **A**ttention (CoCA). Specifically, we enforce a collinear constraint between Q and K by initializing the angle between every two hidden dimensions in the Q and K vectors to 0. This allows for a seamless integration of RoPE and self-attention. The architecture of CoCA and its comparison with RoFormer (Su et al., 2024) is illustrated in Figure 2.

Extensive experiments show that a CoCA-based GPT model, trained within a 512 context length, can seamlessly extend the context window up to 32K (60×) without perplexity divergence. A comprehensive comparison between our method and existing methods is presented in Figure 1. Furthermore, it enhances long-context retrieval ability, achieving a passkey retrieval accuracy of 50%+ even when extrapolating to 16× longer than its training context length by applying dynamic-NTK (Emozilla, 2023). Additionally, by dropping CoCA in LLaMA-7B, we achieve extrapolation up to 32K within only a 2K training length.

Our main contributions can be summarized as follows:

- We unveil the undesirable context boundary behavior resulting from the lack of modeling the relationship between position embedding and self-attention.
- To tackle the undesirable context boundary behavior, we propose Collinear Constrained Attention (CoCA) to seamlessly integrate the position embedding and self-attention, achieving excellent long context window extrapolation performance.
- CoCA extends its context window from 512 to 32K without fine-tuning, achieving over 50% passkey accuracy even when 16 × longer than its training length. Using CoCA in LLaMA-7B, we achieve extrapolation up to 32K within just 2K training length.
- CoCA introduces minimal computational and spatial complexity compared to vanilla self-attention. We provide an optimized implementation of CoCA, making it able to be a drop-in replacement for existing transformer-based models.

2 Preliminary

2.1 Rotary Position Embedding

Position embedding is a crucial component in transformer-based models. Here we focus on Rotary Position Embedding (RoPE) (Su et al., 2024), which is widely used by LLMs including LLaMA (Touvron et al., 2023a), LLaMA-2 (Touvron et al., 2023b), GPT-NeoX (Black et al., 2022) and Qwen (Bai et al., 2023). Suppose the positional index is an integer $n \in [1, N]$, and the corresponding input vector $\boldsymbol{x} = [x_0, x_1, ..., x_{d-1}]^{\mathrm{T}}$, where N is the sequence length, d is the dimension of the attention head. RoPE defines a vector-valued complex function $f(\boldsymbol{x}, n)$:

$$\begin{aligned} f(\boldsymbol{x}, n) = [&(x_0 + ix_1)e^{in\theta_0}, (x_2 + ix_3)e^{in\theta_1}, \\ &\ldots, (x_{d-1} + ix_d)e^{in\theta_{d/2-1}}]^{\mathrm{T}}, \\ &where\ \ \theta_j = \mathrm{B}^{-2j/d}, \end{aligned} \tag{1}$$

here, the base $\mathrm{B} = 10,000$.

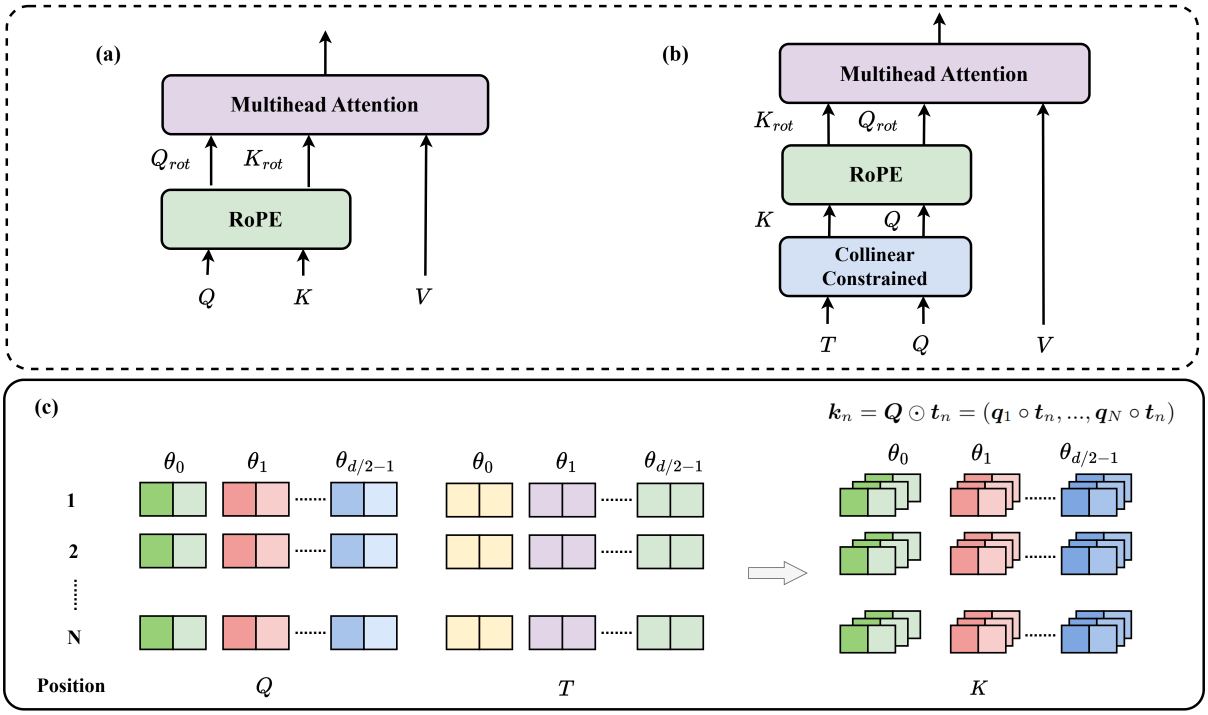


Figure 2: Architecture comparison between RoFormer and CoCA. (a) RoFormer; (b) CoCA; (c) The implementation detial of K in CoCA. Q, T, and V are produced using projection matrices identical to those employed in the vanilla self-attention. T undergoes a halving operation, with the other half being duplicated. K is then computed as the element-wise product of Q and T, adhering to a collinear constraint with Q. Note that $\boldsymbol{k}_n \in \mathbb{R}^{N\times d}$, where $n \in [1, N]$ is the positional index of key, d is the head dimension, N is the sequence length.

After the application of RoPE, the transformed vectors for query ($\boldsymbol{q}$) and key ($\boldsymbol{k}$) become $f(\boldsymbol{q}, m)$ and $f(\boldsymbol{k}, n)$, respectively. Here, $m, n \in [0, N]$ represent the positional indices of $\boldsymbol{q}$ and $\boldsymbol{k}$. The attention operation is computed as the inner product between $f(\boldsymbol{q}, m)$ and $f(\boldsymbol{k}, n)$, defined as follows:

$$
\begin{aligned}
a(m,n) &= \mathrm{Re}(\langle f(\boldsymbol{q},m), f(\boldsymbol{k},n)\rangle) \\
&= \mathrm{Re}\left[\sum_{j=0}^{d/2-1}(q_{2j}+iq_{2j+1})(k_{2j}-ik_{2j+1})e^{i(m-n)\theta_j}\right] \\
&= \sum_{j=0}^{d/2-1}[(q_{2j}k_{2j}+q_{2j+1}k_{2j+1})\cos((m-n)\theta_j) \\
&\quad + (q_{2j}k_{2j+1}-q_{2j+1}k_{2j})\sin((m-n)\theta_j)]
\end{aligned} \tag{2}
$$

The attention score $a(m-n)$ depends on the relative position $(m-n)$.

2.2 Anomalous Behavior between RoPE and Attention Matrices

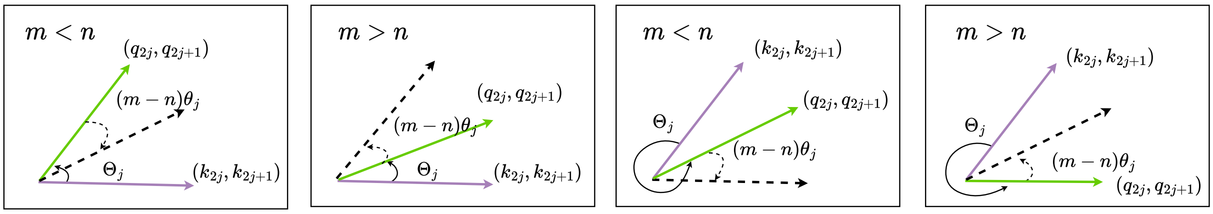
After applying RoPE, the attention score $a(m-n)$ can be interpreted as the sum of $d/2$ inner products of complex numbers, as illustrated in Equation (2). For any pair of $\boldsymbol{q}_j = (q_{2j}, q_{2j+1})$ and $\boldsymbol{k}_j = (k_{2j}, k_{2j+1})$, which is the 2-dimensional slicing of $\boldsymbol{q}$ (or $\boldsymbol{q}_m$) and $\boldsymbol{k}$ (or $\boldsymbol{k}_n$), we introduce the initial angle Θ_j between them, measured counterclockwise from $\boldsymbol{k}_j$ to $\boldsymbol{q}_j$ in the complex plane. Throughout our analysis, we keep the position of $\boldsymbol{k}_j$ fixed, systematically rotating $\boldsymbol{q}_j$ to comprehensively examine their relative positions. The final angle between $\boldsymbol{q}_j$ and $\boldsymbol{k}_j$ is represented as $\theta(\boldsymbol{q}_j, \boldsymbol{k}_j) = \Theta_j + (m-n)\theta_j$, where m and n are positional indices of $\boldsymbol{q}_j$ and $\boldsymbol{k}_j$.

In this concept, the attention score can be formulized as:

$$
a(m,n) = \sum_{j=0}^{d/2-1} |\boldsymbol{q}_j||\boldsymbol{k}_j|\cos(\theta(\boldsymbol{q}_j, \boldsymbol{k}_j)) \tag{3}
$$

Refer to Figure 3 for a visual representation of this concept for any individual $j \in [0, d/2]$ in the 2-D subspace. There are four distinct scenarios between $\boldsymbol{q}_j$ and $\boldsymbol{k}_j$ after rotation.

(1) **Scenario (b) and (c):** When $m > n$ and $\Theta_j \leq \pi$, or $m < n$ and $\Theta_j > \pi$, the value of $\cos(\theta(\boldsymbol{q}_j, \boldsymbol{k}_j))$ between $\boldsymbol{q}_j$ and $\boldsymbol{k}_j$ decreases with the expanding distance between m and n. In these 2 scenarios, no anomalous behavior is observed, as the attention score naturally decreases with the positional distance. This trend persists until the relative angle $\theta(\boldsymbol{q}_j, \boldsymbol{k}_j)$ rotates beyond the boundary of π.



(a) (b) (c) (d)

Figure 3: Anomalous behavior of RoPE in 2-D plane. The inner product of vectors $\boldsymbol{q}_j$ and $\boldsymbol{k}_j$ is contingent upon the relative angle $\theta(\boldsymbol{q}_j, \boldsymbol{k}_j)$, defined as $\Theta_j + (m-n)\theta_j$. Here, Θ_j represents the initial angle, and $(m-n)\theta_j$ signifies the position-dependent rotation angle. (a) $m < n$ and $\Theta_j \leq \pi$. (b) $m > n$ and $\Theta_j \leq \pi$. (c) $m < n$ and $\Theta_j > \pi$. (d) $m > n$ and $\Theta_j > \pi$.

(2) **Scenario (a) and (d):** When $m < n$ and $\Theta_j \leq \pi$, or $m > n$ and $\Theta_j > \pi$, intriguing phenomena emerge. As the distance between m and n grows, the value of $\cos(\theta(\boldsymbol{q}_j, \boldsymbol{k}_j))$ between $\boldsymbol{q}_j$ and $\boldsymbol{k}_j$ paradoxically increases. This anomaly has a notable impact on attention scores, particularly affecting the τ closest tokens. In this context, τ is defined as Θ_j/θ_j for scenario (a) and $(2\pi - \Theta_j)/\theta_j$ for scenario (d). Consequently, attention scores for these tokens are abnormally diminished.

For bidirectional language models, all four cases may occur. For causal models, only scenario (b) and (d) manifest, as m consistently exceeds n.

The attention score $a(m-n)$ is the sum of $d/2$ inner-products, one of them appearing anomalous might be insignificant, however, experimental results confirm its significance.

2.3 Rotary Borders Analysis

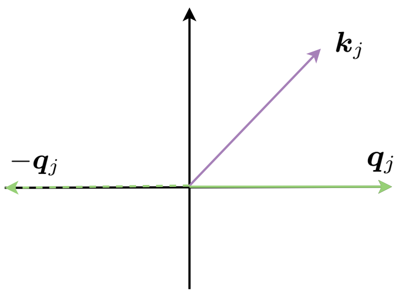


Figure 4: Rotary Borders Analysis. Regarding $\boldsymbol{q}_j$ as x-axis, 3 distinct boundaries correspond to $\boldsymbol{k}_j$, $-\boldsymbol{q}_j$, and $\boldsymbol{q}_j$

Without lose of generality, in this section, we focus on case (d) to further analyse the anomalous behavior during context extrapolation. As shown in Figure 4, three distinct boundaries appear during the rotation process. These boundaries are $\boldsymbol{k}_j$, $-\boldsymbol{q}_j$, and $\boldsymbol{q}_j$, respectively. Once the relative angle between $\boldsymbol{q}_j$ and $\boldsymbol{k}_j$ crosses these boundaries, the monotonicity of $< \boldsymbol{q}_j, \boldsymbol{k}_j >$ will reverse. Specifically, the monotonicity of $< \boldsymbol{q}_j, \boldsymbol{k}_j >$ learns the pattern described as follows:

$$< \boldsymbol{q}_j, \boldsymbol{k}_j >= \begin{cases} \uparrow (m-n), \forall -(2\pi - \Theta_j) \leq \theta(\boldsymbol{q}_j, \boldsymbol{k}_j) < 0 \\ \downarrow (m-n), \forall 0 \leq \theta(\boldsymbol{q}_j, \boldsymbol{k}_j) < \pi \\ \uparrow (m-n), \forall \pi \leq \theta(\boldsymbol{q}_j, \boldsymbol{k}_j) < 2\pi \\ \dots \\ \uparrow (m-n), \forall (2k-1)\pi \leq \theta(\boldsymbol{q}_j, \boldsymbol{k}_j) < (2k)\pi \\ \downarrow (m-n), \forall (2k)\pi \leq \theta(\boldsymbol{q}_j, \boldsymbol{k}_j) < (2k+1)\pi \end{cases} \tag{4}$$

where $\theta(\boldsymbol{q}_j, \boldsymbol{k}_j) = \Theta_j + (m-n)\theta_j$ defined in Section 2.2. This introduces confusion into the model during direct context extrapolation.

Apart from the interval $-(2\pi - \Theta_j) \leq \theta(\boldsymbol{q}_j, \boldsymbol{k}_j) < 0$ in Equation (4), the boundaries introduced by $-\boldsymbol{q}_j$ and $\boldsymbol{q}_j$ exhibit regularity with a periodicity of 2π. RoPE Scaling methods, such as PI and NTK, can effectively address these two boundary issues.

Conversely, the boundaries introduced by $\boldsymbol{k}_j$ present significant challenges. These boundaries disrupt the 2π periodicity, increasing the complexity of the extrapolation. Additionally, there are a total of $d/2 * h * L$ different such boundaries, where d, h, and L represent the head dimension, number of heads, and number of layers, respectively. Methods like PI and NTK are ineffective at these boundaries, even causing more positions to fall into this abnormal region. For PI, this increases by a factor of k (where k is the scaling factor), and for NTK, it increases by a factor of $\lambda^{2j/d}$ (where λ is the scaling factor). From this perspective, the positional concentration of PI results in more issues than NTK, with additional positions falling into the abnormal area during context extrapolation. This partially explains why NTK can be directly used in standard self-attention mechanisms without fine-tuning, while PI requires fine-tuning.

We resolve this third border-related challenge associated with $\boldsymbol{k}_j$ by constraining $\boldsymbol{k}_j$ to be collinear with $\boldsymbol{q}_j$, making it possible for extrapolation with regular periodicity.

3 Method

3.1 Collinear Constrained Attention

To tackle the anomalous behavior between RoPE and attention matrices, we propose a new attention mechanism, **Co**llinear **C**onstrained **A**ttention, namely CoCA. Specifically, by applying a collinear constraint to any pair of $\boldsymbol{q}_j = (q_{2j}, q_{2j+1})$ and $\boldsymbol{k}_j = (k_{2j}, k_{2j+1})$, we seamlessly integrate RoPE into self-attention mechanism, achieving long context extrapolation.

To formalize this, considering a sequence of N input tokens $\mathbb{S}_N = \{w_n\}_{n=1}^N$, with corresponding word embeddings $\mathbb{E}_N = \{\boldsymbol{x}_n\}_{n=1}^N$, where $\boldsymbol{x}_n \in \mathbb{R}^d$ is the d-dimensional word embedding vector of token w_n without position information. First, the queries $\boldsymbol{q}_m$ are obtained:

$$\boldsymbol{q}_m = \mathbf{W}_Q \boldsymbol{x}_m, \forall m \in [1, N] \tag{5}$$

Next, we derive the keys $\boldsymbol{k}_n$ with collinear constraints. This begins with the introducing of the constraint coefficient $\boldsymbol{t}_n$ for each token position n, as depicted in Equation (6).

$$\boldsymbol{t}_n = \mathbf{W}_T \boldsymbol{x}_n, \forall n \in [1, N] \tag{6}$$

Next, Equation (7) imposes the collinearity condition on the coefficients t_{2j} and t_{2j+1}, where $\boldsymbol{t}_n = [t_0, t_1, ..., t_{d-1}]^{\mathrm{T}}$, ensuring that each pair is identical. This step effectively duplicates each 2-dimensional segment of the tensor.

$$\begin{aligned} &t_{2j} = t_{2j+1}, \forall j \in [0, d/2-1] \\ &\boldsymbol{t}_n = \mathrm{Relu}(\boldsymbol{t}_n) \end{aligned} \tag{7}$$

Subsequently, the keys are calculated as shown in Equation (8), where $\boldsymbol{k}_n$ are represented by the element-wise multiplication of $\boldsymbol{Q} = (\boldsymbol{q}_1, ..., \boldsymbol{q}_N)$ and $\boldsymbol{t}_n$. This results in an expansion of dimensionality, as $\boldsymbol{k}_n \in \mathbb{R}^{N \times d}$ now includes an additional sequence length dimension. We address potential memory pressure by optimizing tensor contractions, ensuring no net increase in memory consumption. For an in-depth analysis, please refer to Appendix C.

$$\boldsymbol{k}_n = \boldsymbol{Q} \odot \boldsymbol{t}_n = (\boldsymbol{q}_1 \circ \boldsymbol{t}_n, ..., \boldsymbol{q}_N \circ \boldsymbol{t}_n) \tag{8}$$

After that, we apply RoPE on Q and K, with the function f detailed in Equation (1).

$$\begin{aligned} &f(\boldsymbol{q}_m) = f(\boldsymbol{q}_m, m) \\ &f(\boldsymbol{k}_n) = f(\boldsymbol{Q} \odot \boldsymbol{t}_n, n) = f(\boldsymbol{Q}, n) \odot \boldsymbol{t}_n \end{aligned} \tag{9}$$

Finally, the attention score of CoCA would be:

$$a(m, n) = \mathrm{Re}(\langle f(\boldsymbol{q}_m, m), f(\boldsymbol{q}_m, n) \circ \boldsymbol{t}_n \rangle) \tag{10}$$

Equation (10) illustrates the additional dimension of the keys in our CoCA mechanism. Specifically, it maps the index of each query to the additional dimension, establishing a collinear relationship between the n-th key and the m-th query. This is a critical aspect of our method.

3.2 Slacking the Constraint on Query

In Section 3.1, we present a theoretically precise solution for CoCA. However, practical implementation faces challenges due to the complexity of $O(N^2 d)$ when storing $f(\boldsymbol{Q}, n)$. To address this issue, we provide a dual implementation with $O(Nd)$ complexity in this section and prove their equivalence.

Theorem 1. *(Dual implementation of CoCA) For any attention score defined in Equation (10), there exists an equivalent form as follows:*

$$a(m, n) = \mathrm{Re}(\langle f(\boldsymbol{q}_m, m), \boldsymbol{q}_m \circ f(\boldsymbol{t}_n, n) \rangle) \tag{11}$$

with constraint:

$$q_{2j} = q_{2j+1}, \forall j \in [0, d/2-1] \tag{12}$$

Proof: The proof consists of two steps.

Step 1. We prove that, by imposing the constraint $q_{2j} = q_{2j+1}$, $\forall j \in [0, d/2-1]$, $\mathrm{Re}(\langle f(\boldsymbol{q}_m, m), \boldsymbol{q}_m \circ f(\boldsymbol{t}_n, n) \rangle)$ is equivalent to $\mathrm{Re}(\langle f(\boldsymbol{q}_m, m), f(\boldsymbol{q}_m, n) \circ \boldsymbol{t}_n \rangle)$.

To see this, we calculate the difference between $f(\boldsymbol{q}_m, n) \circ \boldsymbol{t}_n$ and $\boldsymbol{q}_m \circ f(\boldsymbol{t}_n, n)$:

$$\begin{aligned} &f(\boldsymbol{q}_m, n) \circ \boldsymbol{t}_n - \boldsymbol{q}_m \circ f(\boldsymbol{t}_n, n) \\ &= \begin{pmatrix} t_0(q_0 \cos n\theta_0 - q_1 \sin n\theta_0) \\ t_1(q_0 \sin n\theta_0 + q_1 \cos n\theta_0) \\ \cdots \\ t_{d-2}(q_{d-2} \cos n\theta_{d/2-1} - q_{d-1} \sin n\theta_{d/2-1}) \\ t_{d-1}(q_{d-2} \sin n\theta_{d/2-1} + q_{d-1} \cos n\theta_{d/2-1}) \end{pmatrix} \\ &- \begin{pmatrix} q_0(t_0 \cos n\theta_0 - t_1 \sin n\theta_0) \\ q_1(t_0 \sin n\theta_0 + t_1 \cos n\theta_0) \\ \cdots \\ q_{d-2}(t_{d-2} \cos n\theta_{d/2-1} - t_{d-1} \sin n\theta_{d/2-1}) \\ q_{d-1}(t_{d-2} \sin n\theta_{d/2-1} + t_{d-1} \cos n\theta_{d/2-1}) \end{pmatrix} \end{aligned} \tag{13}$$

Recall that $t_{2j} = t_{2j+1}, \forall j \in [0, d/2-1]$ (see Equation (7)), Equation (13) is equivalent to:

$$\begin{aligned} &f(\boldsymbol{q}_m, n) \circ \boldsymbol{t}_n - \boldsymbol{q}_m \circ f(\boldsymbol{t}_n, n) \\ &= \begin{pmatrix} t_0(q_0 - q_1) \sin n\theta_0 \\ t_1(q_0 - q_1) \sin n\theta_0 \\ \cdots \\ t_{d-2}(q_{d-2} - q_{d-1}) \sin n\theta_{d/2-1} \\ t_{d-1}(q_{d-2} - q_{d-1}) \sin n\theta_{d/2-1} \end{pmatrix} \end{aligned} \tag{14}$$

Clearly, if we impose the constraint $q_{2j} = q_{2j+1}, \forall j \in [0, d/2-1]$, the vector in Equation (14) becomes null and we deduce that:

$$f(\boldsymbol{q}_m, n) \circ \boldsymbol{t}_n - \boldsymbol{q}_m \circ f(\boldsymbol{t}_n, n) = \boldsymbol{0} \tag{15}$$

Consequently, with the constraint $q_{2j} = q_{2j+1}, \forall j \in [0, d/2-1]$, we have:

$$\begin{aligned} &\text{Re}(\langle f(\boldsymbol{q}_m, m), \boldsymbol{q}_m \circ f(\boldsymbol{t}_n, n)\rangle) \\ &= \text{Re}(\langle f(\boldsymbol{q}_m, m), f(\boldsymbol{q}_m, n) \circ \boldsymbol{t}_n\rangle) \end{aligned} \tag{16}$$

Step 2. We further demonstrate that, $q_{2j} = q_{2j+1}, \forall j \in [0, d/2-1]$ is in fact a redundant constraint when calculating $\text{Re}(\langle f(\boldsymbol{q}_m, m), f(\boldsymbol{q}_m, n) \circ \boldsymbol{t}_n\rangle)$. To verify this, we expand the inner product:

$$\begin{aligned} &\text{Re}(\langle f(\boldsymbol{q}_m, m), f(\boldsymbol{q}_m, n) \circ \boldsymbol{t}_n\rangle) \\ &= \sum_{j=0}^{d/2-1} [(q_{2j}^2 t_{2j} + q_{2j+1}^2 t_{2j+1}) \cos((m-n)\theta_j) \\ &\quad + (q_{2j} q_{2j+1} t_{2j} - q_{2j+1} q_{2j} t_{2j+1}) \sin((m-n)\theta_j)] \end{aligned} \tag{17}$$

Recall again $t_{2j} = t_{2j+1}, \forall j \in [0, d/2-1]$, we have

$$\begin{aligned} &\text{Re}(\langle f(\boldsymbol{q}_m, m), f(\boldsymbol{q}_m, n) \circ \boldsymbol{t}_n\rangle) \\ &= \sum_{j=0}^{d/2-1} t_{2j} [(q_{2j}^2 + q_{2j+1}^2) \cos((m-n)\theta_j)] \\ &= \sum_{j=0}^{d/2-1} t_{2j} |\boldsymbol{q}_j|^2 \cos((m-n)\theta_j) \end{aligned} \tag{18}$$

This implies that $\text{Re}(\langle f(\boldsymbol{q}_m, m), f(\boldsymbol{q}_m, n) \circ \boldsymbol{t}_n\rangle)$ depends solely on the magnitude of $\boldsymbol{q}_j = (q_{2j}, q_{2j+1})$ in 2-D subspace, demonstrating the independence of the relationship between q_{2j} and q_{2j+1}. Refer to Appendix D.2 for the rigorous proof.

Now we conclude that, with the constraint $q_{2j} = q_{2j+1}, \forall j \in [0, d/2-1]$, $\text{Re}(\langle f(\boldsymbol{q}_m, m), \boldsymbol{q}_m \circ f(\boldsymbol{t}_n, n)\rangle)$ is equivalent to $\text{Re}(\langle f(\boldsymbol{q}_m, m), f(\boldsymbol{q}_m, n) \circ \boldsymbol{t}_n\rangle)$ with no constraint on query. □

By removing $q_{2j} = q_{2j+1}$ constraint, we designate this modified version as CoCA-Slack. The mathematical definition is provided in Appendix D.3.

4 Experimental Setting

This section provides an overview of the experimental setup, including details regarding the training data utilized and the baseline models employed to evaluate the effectiveness of the proposed method.

4.1 Training Data

Our model undergoes training on a combination of datasets, including the Pile training dataset (Gao et al., 2020), BookCorpus (Zhu et al., 2015), and the Wikipedia Corpus (Foundation, 2021). Additionally, we integrate manually collected open-source code from GitHub repositories with at least 1 star. From these datasets, we derive a sample of approximately 50B tokens, maintaining a composition of 75% text and 25% code.

4.2 Model Variants

To evaluate the effectiveness of our proposed approach, we train 3 models from scratch under identical experimental settings, including ALibi (Press et al., 2022), RoFomer (Su et al., 2024), and RoFormer+CoCA. All models share common specifications, featuring a size of 350M, 24 layers, a hidden dimension of 1024, 16 attention heads, and a maximum sequence length of 512. The key distinctions among them lie in variations in self-attention mechanisms and position embeddings. The implementation is optimized based on EleutherAI GPT-NeoX[1]. Training a model from scratch demands substantial computational resources. Therefore, we also conduct experiments involving fine-tuning existing LLMs integrated with CoCA module. For this purpose, we utilize the LLaMA-7B model (Touvron et al., 2023a), which was trained with a context length of 2,048. Additionally, we employ dynamic-NTK for all the above models.

In summary, our comparison models are categorized as follows:

- **Training from scratch**: ALibi, RoFormer, RoFormer+CoCA, RoFormer+dynamic NTK, and RoFormer+dynamic NTK & CoCA.
- **Fine-tuning LLM**: LLaMA-7B, LLaMA-7B+CoCA, LLaMA-7B+dynamic NTK, and LLaMA-7B+dynamic NTK & CoCA.

4.3 Implementation Detials

Pre-training Procedure We train all models using the next token prediction objective. We use AdamW (Loshchilov and Hutter, 2017) with β_1 = 0.9 and β_2 = 0.95. The learning rate follows a linear warm-up of 1% of total steps, starting from 1e-7. Subsequently, the learning rate is adjusted to 1e-4 with linear decay, eventually reaching 1e-5. The training utilizes 8 A100 GPUs, with a global

[1] https://github.com/EleutherAI/gpt-neox/tree/v2.0

batch size of 256 and 2 gradient steps accumulation, taking approximately 96 hours for 2 epochs.

Fine-tuning Procedure To integrate CoCA in LLaMA, we employ a three-stage fine-tuning strategy: (1) only updating the K projection (7% of parameters). This stage aims to reconstruct the K projection in CoCA. By freezing the other parameters, we maintain attention scores as closely as possible to those of vanilla self-attention. (2) updating the QKV projection (21% of parameters). This stage aims to address intrinsic over-fitting in vanilla self-attention caused by undesired behaviors between RoPE and attention matrices. (3) fine-tuning all parameters. Each stage involves 15K steps, totaling 7.5B tokens (22B tokens overall), using the next token prediction objective. The training length of LLaMA-7B + CoCA remains at 2,048 as in the original model. All experiments are conducted with 32 A100 GPUs, setting a per-device batch size to 8 without gradient accumulation.

5 Experiment Results

We conducted experiments to shed light on the following reasonable doubts:

- Can CoCA improve the long context extrapolation performance of existing models?
- Can combining CoCA with other extending methods for RoPE effectively solve the three types of rotational boundary problems?
- Can CoCA maintain its performance in short context windows? (Detailed in Appendix B.4.)

5.1 Long Sequence Language Modeling

We evaluate the long sequence language modeling performance of both our model and baseline models on the test splits of the PG-19 dataset (Rae et al., 2020). For this evaluation, we randomly select a subsample comprising 100 documents, each containing at least 32,768 SentencePiece (Kudo and Richardson, 2018) tokens. We then truncate each test document to its initial 32,768 tokens. The evaluation involves calculating perplexity across different context window sizes using a sliding window approach, as described by (Press et al., 2022), with a stride of 512. The perplexity results for both our models and baselines are presented in Table 1 and Figure 1.

Based on our experiments, the evaluation results indicate that models integrated with CoCA exhibit significantly improved perplexity with longer inference sequence length. For pre-trained models, by increasing the context window size from 512 (training context window size) to 32k, the perplexity of CoCA only increases from 20.11 to 171.63, whereas the perplexity of RoFormer becomes $\inf$. Additionally, by increasing the context window size from 2K to 32K, the perplexity of fine-tuned LLaMA-7B+CoCA only increases 21.68, while LLaMA-7B with other extending methods increases more than 100. In general, we observe a consistent trend of CoCA achieving better perplexity with longer context windows. This suggests that CoCA has a more robust position embedding, enabling it to handle long context more effectively.

In contrast, we observe that models extended through the direct application of dynamic NTK-aware Scaled RoPE exhibit a larger increase in perplexity at longer sequences. The perplexity of both RoFormer+dynamic NTK and LLaMA-7B+dynamic NTK remains significantly higher than that combining CoCA. This difference becomes more pronounced as the sequence length increases. When the inference sequence length reaches 32k, the perplexity of RoFormer+dynamic NTK increases to 380.75, while the result for RoFormer+CoCA is only 171.63. Similarly, the perplexity of LLaMA-7B+dynamic NTK reaches 133.87, whereas LLaMA-7B+CoCA is only 29.95.

It is worth noting that the model achieves the best performance when both dynamic NTK and CoCA are combined. Particularly, LLaMA-7B+dynamic NTK & CoCA consistently maintains a very low perplexity. Even when the inference sequence length has reached 32k ($16 \times$ longer than the training length), the perplexity is only 13.89. This indicates that combining CoCA with other extending methods for RoPE can effectively address the three types of rotational boundary problems, achieving robust long-text extrapolation modeling capabilities.

5.2 Long Context Retrieval

Perplexity evaluates the performance of language model in predicting the next token. However, it is insufficient for a comprehensive assessment of the effective context window size. To address this, we conducted experiments using a passkey retrieval task (Mohtashami and Jaggi, 2023) to evaluate our method and baselines. The task involves identifying and retrieving a randomly hidden passkey within a lengthy document. More details of task definition and test sample generation settings can

Method	Evaluation Context Window Size (Perplexity ↓)						
	512	1024	2048	4096	8192	16k	32k
Training model from scratch							
ALibi	**18.69**	21.27	28.20	35.66	**37.03**	OOM	OOM
RoFomer	19.66	411.50	3276.00	3026.00	3028.00	inf	inf
+ dynamic NTK	19.66	22.30	38.00	75.75	138.13	370.75	380.75
+ CoCA	20.11	33.47	69.06	113.19	157.38	141.00	171.63
+ dynamic NTK & CoCA	20.11	**20.81**	**25.88**	**34.16**	55.75	**89.31**	**101.13**
Fine-tuning LLM							
LLaMA-7B	**9.25**	**7.56**	**7.30**	9673.14	inf	inf	inf
+ dynamic NTK	9.25	7.56	7.30	9.40	14.40	63.62	133.87
+ CoCA	9.91	8.49	8.27	24.23	42.00	23.83	29.95
+ dynamic NTK & CoCA	9.91	8.49	8.27	**8.61**	**9.56**	**11.10**	**13.98**

Table 1: Evaluation perplexity on 100 PG-19 documents using sliding window (S = 512) strategy. Dynamic-NTK is employed without fine-tuning. The best result is highlighted in bold.

Method	Evaluation Context Window Size (Accuracy↑)						
	512	1024	2048	4096	8192	16k	32k
Traning model from scratch							
ALibi	0.82	0.65	0.28	0.18	0.12	OOM	OOM
RoFomer	0.99	0.53	0.30	0.18	0.04	0.02	0.04
+ dynamic NTK	0.99	1.00	0.95	0.70	0.41	0.16	0.06
+ CoCA	1.00	0.64	0.33	0.19	0.06	0.02	0.04
+ dynamic NTK & CoCA	**1.00**	**1.00**	**0.96**	**0.89**	**0.50**	**0.23**	**0.08**
Fine-tuning LLM							
LLaMA-7B	1.00	1.00	1.00	0.61	0.21	0.07	0.09
+ dynamic NTK	1.00	1.00	1.00	0.81	0.26	0.06	0.03
+ CoCA	1.00	1.00	1.00	0.71	0.28	0.11	0.10
+ dynamic NTK & CoCA	**1.00**	**1.00**	**1.00**	**1.00**	**0.85**	**0.51**	**0.30**

Table 2: Long context retrieval performance on passkey retrieval task. The best result is highlighted in bold.

be found in Appendix B.1. Table 2 illustrates the accuracy of all tested models and their variants.

It is evident that ALibi exhibits failures when tested on sequences that were 1× longer than its training length, attributed to its local hypothesis. In contrast, our model consistently demonstrated superior accuracy. RoFormer+dynamic NTK & CoCA maintained a 50% accuracy, even with the test sequence length expanded to 16× its training length. Similarly, LLaMA-7B+dynamic NTK & CoCA still maintained a 30% accuracy when the test length was up to 32K.

5.3 Impact of Strict and Slack Constraint on Q

As mentioned in Section 3.2, we implement a slack version of CoCA, referred to as CoCA-Slack. In this section, under the same experimental settings, we implement two versions of CoCA based on RoFormer-350M, labeled as CoCA-Slack and CoCA-Strict. The comparison results between them are shown in Table 3.

We observe that the CoCA-Strict and CoCA-Slack exhibit similar performance in long sequence language modeling, as evidenced by comparable perplexity results. However, in the passkey retrieval task, contrary to our initial expectations, the CoCA-Strict model produces significantly lower results. This unexpected outcome suggests that models with a slack constraint may offer additional performance advantages, such as a larger effective context window size.

Understanding the reasons behind the superiority of slack constraints will be a key focus of our future work. In this regard, we provide some theoretical insights in Appendices D.2 and D.3. These insights aim to shed light on the underlying mechanisms that contribute to the observed differences and lay the groundwork for a more comprehensive analysis in subsequent research.

Method		512	1024	2048	4096	8192	16384	32768
Performance on Long Sequence Modeling (Perplexity)								
ntk-2	CoCA-Slack	20.11	19.02	24.92	40.53	68.38	92.75	103.44
	CoCA-Strict	+0.07	+0.61	-1.58	-4.03	+15.37	+12.38	+1.94
ntk-4	CoCA-Slack	20.11	20.81	25.88	34.16	55.75	89.31	101.13
	CoCA-Strict	+0.07	-0.49	-0.66	-0.88	+3.16	-18.25	-2.57
ntk-8	CoCA-Slack	20.11	23.66	29.05	37.47	55.5	88.88	111.38
	CoCA-Strict	+0.07	-1.74	-0.64	+1.16	+0.03	+0.5	+0.31
Performance on Long Context Retrieval (Passkey Accuracy)								
ntk-2	CoCA-Slack	1.0	0.99	0.94	0.77	0.47	0.27	0.15
	CoCA-Strict	+0.0	-0.12	-0.3	-0.42	-0.34	-0.22	-0.07
ntk-4	CoCA-Slack	1.0	1.0	0.96	0.89	0.5	0.23	0.08
	CoCA-Strict	+0.0	-0.11	-0.38	-0.46	-0.38	-0.19	-0.02
ntk-8	CoCA-Slack	1.0	0.98	0.99	0.85	0.5	0.11	0.02
	CoCA-Strict	+0.0	-0.05	-0.34	-0.51	-0.4	-0.07	-0.01

Table 3: Comparison results for the Strict and Slack Constraints of $\boldsymbol{Q}$ in our proposed CoCA module. Superior performance to CoCA-Slack is indicated by the green color, while inferior performance is signified by the red color. The perplexity of the strict and slack models is comparable, whereas the strict model achieved lower accuracy in the passkey retrieval task.

6 Conclusion

In this paper, we introduce Collinear Constrained Attention (CoCA), a novel approach that integrates position embedding with the self-attention mechanism. This innovation addresses undesired behaviors occurring around the context window boundary, which stem from discrepancies between RoPE and attention matrices. To the best of our knowledge, we are the first to analyze the initial angles between queries and keys in the self-attention mechanism, which give rise to anomalous phenomena in RoPE. Furthermore, we deduce a slack constraint for our implementation of CoCA. Extensive experiments demonstrate that incorporating CoCA into existing models significantly enhances performance in both long sequence language modeling and long context retrieval tasks. Additionally, the simultaneous integration of CoCA with other extended RoPE methods (e.g., dynamic-NTK) effectively mitigates three types of rotational boundary issues, resulting in remarkably improved capabilities for long context extrapolation.

Limitations

Our current approach, CoCA, has so far been validated exclusively on RoPE. Experimental results demonstrate that CoCA enhances the long-context extrapolation performance of LLMs and further augments other extension methods by addressing rotational boundary issues. However, questions arise regarding its applicability to more general methods. While the effectiveness of slack position embedding (SPE) is evident, a deeper understanding of the underlying reasons for its superior performance necessitates further investigation.

Acknowledgement

Thanks to Jianlin Su from Moonshot AI Ltd and Zhongqiang Huang from Sangfor Technology for their valuable suggestions during the revision of the paper. They pointed out that CoCA is equivalent to relaxing the constraint on the query.

References

Daniel G. a. Smith and Johnnie Gray. 2018. opt_einsum - a python package for optimizing contraction order for einsum-like expressions. *Journal of Open Source Software*, 3(26):753.

Jinze Bai, Shuai Bai, Yunfei Chu, et al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Sidney Black, Stella Biderman, Eric Hallahan, et al. 2022. GPT-NeoX-20B: An open-source autoregressive language model. In *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 95–136, virtual+Dublin. Association for Computational Linguistics.

bloc97. 2023. Ntk-aware scaled rope allows llama models to have extended (8k+) context size without any fine-tuning and minimal perplexity degradation.

Guanzheng Chen, Xin Li, Zaiqiao Meng, Shangsong Liang, and Lidong Bing. 2024. CLEX: Continuous length extrapolation for large language models. In *The Twelfth International Conference on Learning Representations*.

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

Ta-Chung Chi, Ting-Han Fan, Peter J. Ramadge, and Alexander Rudnicky. 2022. KERPLE: kernelized relative positional embedding for length extrapolation. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. `https://github.com/open-compass/opencompass`.

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

Jiayu Ding, Shuming Ma, Li Dong, et al. 2023. Longnet: Scaling transformers to 1,000,000,000 tokens. *arXiv preprint arXiv:2307.02486*.

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

Wikimedia Foundation. 2021. Wikimedia downloads.

Leo Gao, Stella Rose Biderman, Sid Black, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *ArXiv*, abs/2101.00027.

Chi Han, Qifan Wang, Wenhan Xiong, et al. 2023. Lm-infinite: Simple on-the-fly length generalization for large language models. *arXiv preprint arXiv:2308.16137*.

Yunpeng Huang, Jingwei Xu, Zixu Jiang, et al. 2023. Advancing transformer architecture in long-context large language models: A comprehensive survey. *arXiv preprint arXiv:2311.12351*.

Gangwei Jiang, Caigao Jiang, Siqiao Xue, James Zhang, Jun Zhou, Defu Lian, and Ying Wei. 2023. Towards anytime fine-tuning: Continually pre-trained language models with hypernetwork prompts. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12081–12095, Singapore. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*, pages 66–71. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *ArXiv*, abs/1711.05101.

Amirkeivan Mohtashami and Martin Jaggi. 2023. Landmark attention: Random-access infinite context length for transformers. *CoRR*, abs/2305.16300.

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

Ofir Press, Noah A. Smith, and Mike Lewis. 2022. Train short, test long: Attention with linear biases enables input length extrapolation. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. 2020. Compressive transformers for long-range sequence modelling. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Sebastian Ruder, Matthew E. Peters, Swabha Swayamdipta, and Thomas Wolf. 2019. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2, 2019, Tutorial Abstracts*, pages 15–18. Association for Computational Linguistics.

Jianlin Su, Murtadha H. M. Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Yutao Sun, Li Dong, Barun Patra, Shuming Ma, Shaohan Huang, Alon Benhaim, Vishrav Chaudhary, Xia Song, and Furu Wei. 2023. A length-extrapolatable transformer. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 14590–14604. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, et al. 2023a. Llama: Open and efficient foundation language models. *ArXiv*, abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin R. Stone, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, et al. 2023. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 19–27. IEEE Computer Society.

A Related Work

Existing researches are mainly focused on the submodule of attention kernel or position embedding (Huang et al., 2023). In the following sections, we will separately introduce works on these two aspects: Section A.1 primarily addresses the former, while Section A.2 delves into the latter.

A.1 Efficient Attention Mechanisms

Several works aim to implement efficient attention mechanisms with reduced computational demands, even achieving linear complexity. This enables extending the effective context length boundary of LLMs during inference by directly increasing L_{max} in the pre-training stage (Ding et al., 2023; Mohtashami and Jaggi, 2023). Noteworthy approaches include Longformer (Beltagy et al., 2020), utilizing slide window attention, and models such as StreamingLLM (Xiao et al., 2023) and LM-Infinite (Han et al., 2023), which leverage a global-local attention mechanism. These variants have achieved success to a certain extent, but still face issues we unveiled in this work when using RoPE as their positional encoding method.

A.2 Extrapolative Position Embedding Methods

Extrapolative position embedding methods aim to enhance the length generalization capability of LLMs.

A.2.1 Attention Bias

In seeking alternatives to the explicit encoding of positional information, researchers have explored the integration of attention bias to capture the sequential and temporal nuances inherent in natural language. Early approaches, such as T5 (Ruder et al., 2019), incorporate learnable attention bias. However, these methods do not explicitly address the challenge of length extrapolation. ALibi (Press et al., 2022) introduces a negative causal attention bias in a heuristic manner. Extending the ALiBi-style attention bias, KERPLE (Chi et al., 2022) treats it as a composition triangle kernel for self-attention and modifies style Xpos (Sun et al., 2023) by integrating it with RoPE. While these approaches effectively manage to maintain low perplexity levels, they fall short in capturing long-range dependencies due to introducing local hypotheses to context tokens.

A.2.2 Extend RoPE

Besides, various strategies have been explored to extend RoPE (Su et al., 2024), a commonly employed positional encoding method in popular LLMs. Recent approaches involve simply scaling it to extrapolate the inference context length with minimal or no fine-tuning. For instance, Position Interpolation (PI) (Chen et al., 2023) applies linear scaling on each position number from n to n/k, densifying the representation space to extend the farthest length boundary by k times. Other approaches, such as NTK-aware Scaled RoPE (bloc97, 2023) and dynamic-NTK (Emozilla, 2023), combine high-frequency extrapolation and low-frequency interpolation. These training-free methods require limited code changes during inference (Peng et al., 2023). Furthermore, CLEX (Chen et al., 2024) optimizes extrapolation capability by modeling the continuous dynamics over different scaling factors. However, these methods aim solely at alleviating the problem of modeling the rotation angles in out-of-distribution (OOD) positions without recognizing the intrinsic correlation between attention matrices and rotation angles. Therefore, these methods still suffer from a limited context window extending ratio.

Previous methods independently investigate self-attention and position embedding without considering their intrinsic relationship, especially for the widely used RoPE method.

B Additional Experiment

B.1 Passkey Retrieval Task Definition

The passkey retrieval task, as proposed by Mohtashami and Jaggi (2023), involves the model recovering a randomly generated passkey hidden in a long document (see Listing 1 for the task prompt format). Given a language model, we can determine the effective context window by assessing the upper and lower bounds. We assume a random passkey is k tokens away from the end of the input. If a model consistently fails to recover the passkey in multiple attempts, it suggests a context window size smaller than k. Conversely, successful retrievals indicate an effective context window size of at least k tokens (Chen et al., 2023).

```
There is an important info hidden inside a
lot of irrelevant text. Find it and
memorize them. I will quiz you about the
important information there.

The grass is green. The sky is blue. The
sun is yellow. Here we go. There and back
again.
⋮  // Repeat x times.

// Passkey is 5 randomly generated numbers.
The passkey is 12345. Remeber it. 12345 is
the passkey.

The grass is green. The sky is blue. The
sun is yellow. Here we go. There and back
again.
⋮  // Repeat y times.

What is the passkey?
```

Listing 1: Prompt format for passkey retrieval (Mohtashami and Jaggi, 2023). The passkey is randomly generated from 10,000 to 99,999.

In our experiments, we generate test samples based on the prompt template in Listing 1, with lengths ranging from 512 to 32k. There are 100 test cases for each length. Given a language model, we input the passkey task prompt, examine the model's output for the new 64 tokens, and calculate the accuracy.

B.2 Analysis I : Consistency of Optimization in Position Embedding

The passkey retrieval results are presented in Section 5.2. Our model demonstrates superior passkey retrieval accuracy compared to baseline models under various conditions. However, we remain intrigued about its optimization, specifically whether it occurs within or beyond the confines of the training context window. To probe this further, we categorize the experimental data into two segments: passkey distance shorter and farther than the training context window length.

Figure 5 (a) illustrates the comparison results when the passkey is inserted less than 512 tokens away from the end token, while Figure 5 (b) illustrates that outside this range. When the passkey is inserted outside the 512 window, RoFormer+NTK & CoCA consistently outperforms Roformer+NTK across various lengths of inference sequences. This superiority persists when the passkey is inserted inside the 512 window. Notably, with an increase in the length of the inference sequence, RoFormer + NTK & CoCA demonstrates increasingly superior performance compared to RoFormer + NTK. These experiments suggest that our model can consistently optimize the position embedding and extend the effective context window.

B.3 Analysis II : Impact of Dynamic-NTK in CoCA

We utilize the dynamic NTK method (Emozilla, 2023) during the inference process, applying it separately to both our model and the baseline model. To comprehensively assess the robustness of these models, we conduct a thorough validation by varying scaling factors (2, 4, and 8).

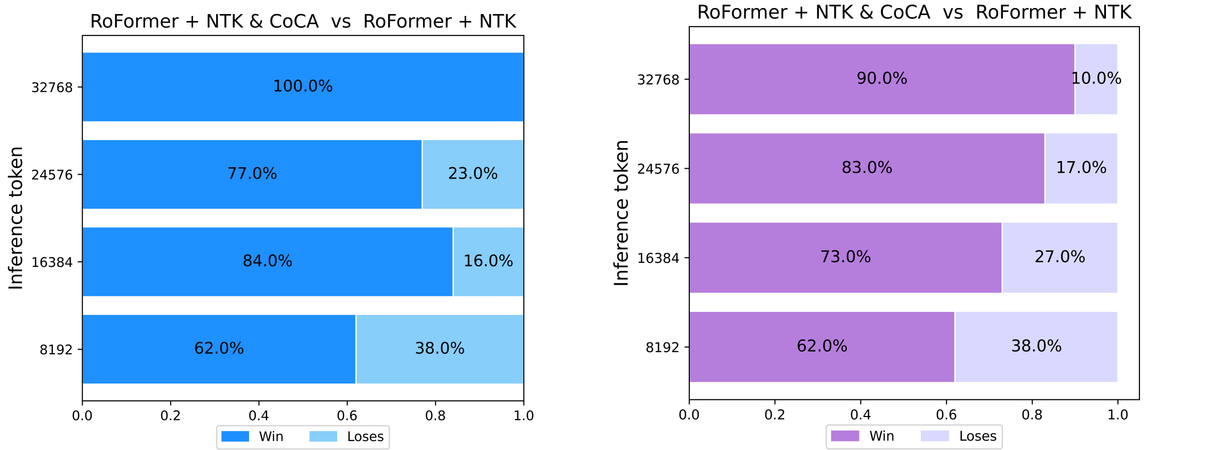
The results in Figures 1 and 6 demonstrate that, with the integration of the dynamic NTK method, our model achieves higher passkey accuracy and lower perplexity. Additionally, when the scaling factor varies between 2, 4, and 8, the vanilla RoFormer model fails to maintain stable performance. In contrast, CoCA consistently outperforms RoFormer at different scaling rates. This consistent trend indicates that our model is more robust, showing minimal performance fluctuations with changes in the scaling factor.

Furthermore, it suggests that by implementing collinear constraints, we can cleverly address anomalous behavior in RoPE, allowing RoPE to better leverage other extrapolation techniques.

B.4 Analysis III : Compatibility of CoCA with PI

B.4.1 Experiment Setup

We conduct experiments utilizing the LLaMA-7B model (Touvron et al., 2023a) and its LLaMA-7B + CoCA variant, as detialed in Section 4.2. To apply PI, we follow the same settings as Chen et al. (2023). All parameters are fine-tuned, and the fine-tuning sequence length is set to 32k. The learning rate is adjusted to $2e-5$ with no decay. The experiments are conducted on 32 A100 GPUs, with a per-device batch size of 1 and no gradient accumulation. Each experiment is carried out for 6,000 steps.



(a) Inserting passkey inside 512 tokens away from end tokens (b) Inserting passkey outside 512 tokens away from end tokens

Figure 5: Comparison of effective context window between RoFormer + NTK and RoFormer + NTK & CoCA.

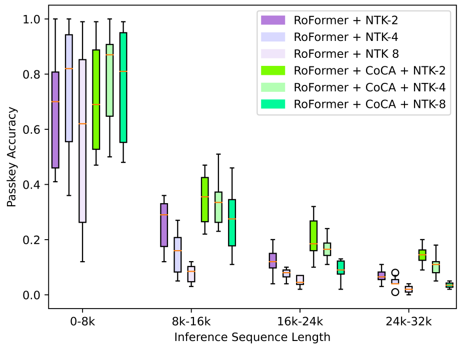


Figure 6: Passkey accuracy distribution on 4 range of distances. CoCA outperforms RoFormer for all distances and scaling factors of NTK.

B.4.2 Long Context Validation

The fine-tuning results with PI are shown in Table 4. For long sequence modeling, both LLaMA-7B+PI and LLaMA-7B+CoCA & PI exhibit competitive performance across sequence lengths from 512 to 32768. Both methods achieve near-perfect accuracy in long context retrieval across all sequence lengths, with scores close to 1.0.

Overall, these findings suggest that the integration of PI and the CoCA module with the LLaMA-7B model yields robust performance in both long sequence modeling and long context retrieval tasks. Additionally, the CoCA module demonstrates the ability to maintain performance levels comparable to PI, particularly evident at longer sequence lengths.

B.4.3 Short Context Validation

In addition to enhancing long-context extrapolation, it is imperative to consider the practicality and scalability of CoCA in short contexts. Hence, we evaluate our model on OpenCompass (Contributors, 2023), which comprises various dimensions, including reasoning, understanding, knowledge, language, and examination. The results are presented in Table 5.

The results indicate that LLaMA-7B with CoCA integration achieves performance comparable to the original LLaMA-7B across all evaluated dimensions. Specifically, CoCA integration does not significantly decrease performance in reasoning, understanding, and language tasks, and even surpasses the original LLaMA in some aspects. The observed decline in the knowledge dimension may be due to the distribution bias between the fine-tuning data and the original pre-training data (Jiang et al., 2023). The abnormal score in the CHID task could be attributed to a bias in our training data, which includes some Chinese documents. Overall, the integrated CoCA outperforms LLaMA-7B in comprehensive scores, demonstrating its effectiveness in both long-context and short-context scenarios, and highlighting its practical value.

C Computational and Spatial Complexity Analysis

In this section, we analyze the computational and spatial complexities of CoCA. Table 6 provides a detailed comparison between the vanilla self-attention mechanism and CoCA.

When using the operation $\boldsymbol{K}_{rot} = \boldsymbol{Q} \circ \boldsymbol{T}_{rot}$, the computational complexity of CoCA does not exceed twice that of the vanilla self-attention. In practice, the training and inference speed of CoCA are comparable to the vanilla self-attention mech-

Method	512	1024	2048	4096	8192	16384	32768
			Performance on Long Sequence Modeling (Perplexity)				
LLaMA-7B+PI	9.06	7.55	7.74	7.16	7.04	6.93	7.11
+ CoCA & PI	9.65	8.19	8.37	7.87	7.84	7.83	7.96
			Performance on Long Context Retrieval (Passkey Accuracy)				
LLaMA-7B+PI	1.0	1.0	1.0	1.0	1.0	1.0	0.99
+ CoCA & PI	1.0	1.0	1.0	1.0	1.0	0.99	0.99

Table 4: Comparison results for LLaMA-7B+PI and LLaMA-7B+CoCA & PI after fine-tuning with sequence length of 32,768. CoCA succeeds in maintaining the performance of PI within fine-tuning window size.

Dataset	LLaMA-7B	LLaMA+CoCA	LLaMA+PI	LLaMA+CoCA & PI
		Reasoning		
COPA	**72.0**	66.0	60.0	59.0
AX_b	**57.5**	41.9	46.7	54.6
AX_g	50.0	50.0	51.4	**52.0**
RTE	46.9	52.4	48.7	**52.7**
OCNLI	30.0	**30.1**	30.0	30.0
CMNLI	**33.1**	33.0	33.0	33.0
		Understanding		
C3	40.3	44.9	46.4	**47.3**
CSL	52.4	**58.5**	58.1	58.2
EPRSTMT	50.0	50.0	**50.2**	50.0
		Knowledge		
BoolQ	**69.4**	62.4	66.5	62.5
		Language		
WiC	**50.2**	50.0	50.0	50.0
WSC	40.4	37.5	**45.2**	37.5
CHID	26.2	64.6	55.0	**65.7**
AFQMC	68.9	**69.0**	**69.0**	**69.0**
		Examination		
MMLU	**35.0**	25.7	28.4	25.8
CMMLU	**26.8**	25.1	26.5	25.4
C-Eval	**27.1**	24.6	26.2	24.8
Total	776.2	785.7	791.2	**797.4**

Table 5: OpenCompass results of LLaMA-7B and its variants. Models integrated with CoCA achieved comparable performance to LLaMA-7B, leading no harm to the expression ability of the model.

anism, with only a slight increase of about 5% to 10% , as depicted in Figure 7. However, there is a significant increase in spatial complexity when expanding $\boldsymbol{K}_{rot} = \boldsymbol{Q} \circ \boldsymbol{T}_{rot}$, becoming d times that of the vanilla self-attention. This level of spatial complexity is not practical for applications.

To address this problem, we can draw inspiration from the computation of $\boldsymbol{Q}_{rot}\boldsymbol{K}_{rot}^{\mathrm{T}}$, which involves two steps: element-wise multiplication between $\boldsymbol{Q}_{rot}$ and $\boldsymbol{K}_{rot}$ followed by summation along the hidden dimension. Optimization is attainable by condensing the hidden dimension before fully expanding the sequence length dimension. Consequently, the spatial complexity is effectively reduced from N^2d to N^2. This optimization strategy is equally applicable to $\boldsymbol{K}_{rot} = \boldsymbol{Q} \circ \boldsymbol{T}_{rot}$. These two components can be unified as articulated in Equation (19):

$$\boldsymbol{Q}_{rot}\boldsymbol{K}_{rot}^{\mathrm{T}} = \boldsymbol{Q}_{rot}(\boldsymbol{Q} \circ \boldsymbol{T}_{rot})^{\mathrm{T}} \tag{19}$$

The commendable work accomplished by opt_einsum (a. Smith and Gray, 2018) facilitates the optimization of Equation (19). Experimental results indicate that Roformer+CoCA only demands approximately 60GB of GPU memory during inference with a sequence length of 32k, aligning closely with the memory consumption of the vanilla self-attention mechanism.

Module	vanilla self-attention		CoCA	
	Computational	Spatial	Computational	Spatial
$\boldsymbol{W}_{QK(T)V}$	$3Nd^2h$	Nd	$3Nd^2h$	Nd
$\boldsymbol{T}$ half	—	—	Ndh	Nd
$\boldsymbol{T}$ Relu	—	—	Ndh	Nd
$\boldsymbol{QK}(\boldsymbol{T})$ rotation	$2Ndh$	Nd	$2Ndh$	Nd
$\boldsymbol{K}_{rot} = \boldsymbol{Q} \circ \boldsymbol{T}_{rot}$	—	—	N^2dh	N^2d
$\boldsymbol{Q}_{rot}\boldsymbol{K}_{rot}^{\mathrm{T}}$	N^2dh	N^2	N^2dh	N^2
Mask	N^2	N^2	N^2	N^2
Softmax	N^2	N^2	N^2	N^2

Table 6: The comparison of computational and spatial complexity between vanilla self-attention block and CoCA. Here, N represents the sequence length, h denotes the number of heads, and d signifies the dimension of each head.

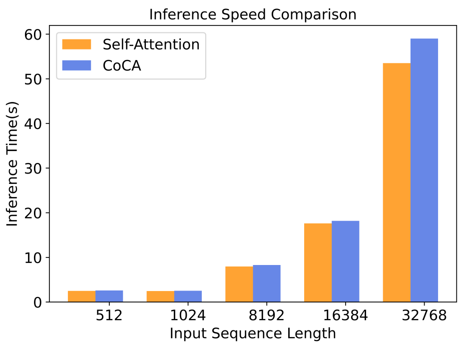


Figure 7: Inference speed comparison between CoCA and vanilla self-attention.

D Theoretical Proof

D.1 Strong Form of Long-term Decay with CoCA

We have introduced the basic theory of Rotary Position Embedding in Section 2.1. In fact, (Su et al., 2024) shows that RoPE has the characteristic of long-term decay:

$$\begin{aligned} |a(s)| &= \left| \mathrm{Re}\left[\sum_{j=0}^{d/2-1} h_j e^{is\theta_j} \right] \right| \\ &\leq (\max_i |h_{i+1} - h_i|) \sum_{j=0}^{d/2-1} |S_{j+1}| \end{aligned} \tag{20}$$

where $h_j := (q_{2j} + iq_{2j+1})(k_{2j} - ik_{2j+1})$ and $S_j := \sum_{k=0}^{j-1} e^{is\theta_k}$, $s = (m - n)$, m for the index of query, n for the index of key. Since the value of $\sum_{j=0}^{d/2-1} |S_{j+1}|$ decays with the relative distance s, the attention score decays either.

This characteristic ensures the stability of RoPE during extrapolation to some extent by preventing outliers. For CoCA, a stronger deduction can be formulated as follows:

$$|a(s)| \leq (\max_i |l_{i+1} - l_i|) \sum_{j=0}^{d/2-1} |C_{j+1}| \tag{21}$$

where $l_j := |q_{2j} + iq_{2j+1}||k_{2j} + ik_{2j+1}|$, and $C_j := \sum_{k=0}^{j-1} \cos(s\theta_k)$. Furthermore, it holds that:

$$|l_{i+1} - l_i| \leq |h_{i+1} - h_i| \tag{22}$$

Proof: Notice that when the initial angle Θ_j between $\boldsymbol{q}_j$ and $\boldsymbol{k}_j$ is 0, from Equation (18), the attention score can be simplified as:

$$\begin{aligned} a(s) &= \mathrm{Re}\left[\sum_{j=0}^{d/2-1} h_j e^{is\theta_j} \right] \\ &= \sum_{j=0}^{d/2-1} l_j \cos(s\theta_j) \end{aligned} \tag{23}$$

By following the study of (Su et al., 2024), we can easily derive the estimation in Equation (21).

For Equation (22), applying the triangle inequality, we get:

$$|h_{i+1} - h_i| \geq ||h_{i+1}| - |h_i|| \tag{24}$$

Reviewing the definition of $h_i = (q_{2j} + iq_{2j+1})(k_{2j} - ik_{2j+1})$, we will find:

$$\begin{aligned} |h_{i+1} - h_i| &\geq ||h_{i+1}| - |h_i|| \\ &= ||\boldsymbol{q}_{i+1}\boldsymbol{k}_{i+1}^*| - |\boldsymbol{q}_i\boldsymbol{k}_i^*|| \\ &= ||\boldsymbol{q}_{i+1}\boldsymbol{k}_{i+1}| - |\boldsymbol{q}_i\boldsymbol{k}_i|| \\ &= |l_{i+1} - l_i| \end{aligned} \tag{25}$$

D.2 Homeomorphism of Representation Space

Theorem 2. *(Homeomorphism of representation space) For any attention score defined as follows:*

$$a(m, n) = \mathrm{Re}(\langle f(\boldsymbol{q}_m, m), f(\boldsymbol{q}_m, n) \circ \boldsymbol{t}_n \rangle) \tag{26}$$

where $\boldsymbol{q}_m$ is the query, m is the index number of query, $\boldsymbol{t}_n$ is the collinear coefficient of CoCA, n is the index number of key, f is the rotation operator.

Denote its representation space with respect to $\boldsymbol{q}_m$ as:

$$F(Q) = \{a(m, n) | \forall \boldsymbol{q}_m \in Q \subset \mathbb{R}^d\} \tag{27}$$

where $\boldsymbol{q}_m = \boldsymbol{W}_Q \boldsymbol{x}_m$, $\boldsymbol{x}_m \in \mathbb{E}_N$, $m \in [1, N]$ and $\mathbb{E}_N$ is the word embedding space, $\boldsymbol{W}_Q$ is the projection matrix.

Then we have the following homeomorphism:

$$F(Q) \cong F(Q_{half}) \tag{28}$$

where $Q_{half} = Q|_{q_{2j} = q_{2j+1}, \forall j \in [0, d/2-1]}$.

Proof: We prove it by demonstrating the homeomorphism mapping $\mathcal{G}$:

$$\mathcal{G} : F(Q) \to F(Q_{half})$$
$$F((q_0, ..., q_{d-1}) \mapsto F((\sqrt{\frac{q_0^2 + q_1^2}{2}}, ..., \sqrt{\frac{q_{d-2}^2 + q_{d-1}^2}{2}}) \tag{29}$$

It consists of three parts:

Part I ($\mathcal{G}$ is a bijection): recall Equation (18), we have:

$$\mathcal{G}(X) = X, \forall X \in F(Q) \tag{30}$$

which implies that $\mathcal{G}$ is an identity mapping, naturally injective.

Next, we prove that $\mathcal{G}$ is also surjective: for any $Y = F((q_0, ..., q_{d-1})|_{q_{2j}=q_{2j+1}}) \in F(Q_{half})$, there exists $\widetilde{Y} \in F(Q)$ such that $\mathcal{G}(\widetilde{Y}) = Y$. Let

$$\widetilde{Y} = F((q_0, ..., q_{d-1})|_{q_{2j}=q_{2j+1}}) \in F(Q) \tag{31}$$

obviously we have $\mathcal{G}(\widetilde{Y}) = Y$.

Part II ($\mathcal{G}$ is continuous): For any $X_0 \in F(Q)$, $\epsilon > 0$, there exists δ, such that if $|X - X_0| < \delta$, then $|\mathcal{G}(X) - \mathcal{G}(X_0)| < \epsilon$.

From *Part I*, $\mathcal{G}$ is an identity mapping, let $\delta = \epsilon$, then the continuity of $\mathcal{G}$ holds.

Part III ($\mathcal{G}^{-1}$ is continuous): $\mathcal{G}$ is an identity mapping, so is $\mathcal{G}^{-1}$. Following Part II, we immediately deduce that $\mathcal{G}^{-1}$ is continuous. □

D.3 Slack Position Embedding

Let $\mathcal{H}$ be a Hilbert space, and $\{\mathcal{T}(n)|n \geq 0\} \subset \mathcal{L}(\mathcal{H})$ is a family of bounded linear operator on $\mathcal{H}$. $\mathcal{A}$ is the inner-product defined on $\mathcal{H}$.

If it satisfies the following property, then we call $\{\mathcal{T}(n)|n \geq 0\}$ is a relative (bounded linear) operator on $\mathcal{H}$:

$$\begin{aligned} &\exists\, \{\mathcal{S}(m)|m \in \mathbb{Z}\} : \mathcal{H} \times \mathcal{H} \to \mathbb{C} \\ &\qquad\qquad (X, Y) \mapsto \mathcal{S}(m)(X, Y) \\ &\text{is a family of semi-bilinear operator on } \mathcal{H} \\ &s.t.\ \mathcal{S}(p - q)(X, Y) = \mathcal{A}(\mathcal{T}(p)(X), \mathcal{T}(q)(Y)) \\ &\forall\, p, q \in [0, N], X, Y \in \mathcal{H}, \end{aligned} \tag{32}$$

Additionally, if it satisfies the following property, then we call $\{\mathcal{T}(n)|n \geq 0\}$ is a slack relative (bounded linear) operator on $\mathcal{H}$:

$$\begin{aligned} &\exists\, \{\mathcal{S}(m)|m \in \mathbb{Z}\} : \mathcal{H} \times \mathcal{H} \to \mathbb{C} \\ &\qquad\qquad (X, Y) \mapsto \mathcal{S}(m)(X, Y) \\ &\text{is a family of semi-bilinear operator on } \mathcal{H} \\ &and\ \mathcal{H}' \subset \mathcal{H}, \mathcal{H}' \neq \varnothing \\ &s.t.\ \mathcal{S}(p - q)(X, Y) = \mathcal{A}(\mathcal{T}(p)(X), \mathcal{T}(q)(Y)) \\ &\forall\, p, q \in [0, N], X, Y \in \mathcal{H}', \end{aligned} \tag{33}$$

Specifically, when $\mathcal{H}$ represents our projection space in self-attention, and $\{\mathcal{T}(n)|n \geq 0\}$ is a position embedding on it, such as the Rotary Position Embedding (RoPE), we refer to it as a Slack Position Embedding (SPE) if it satisfies the property described in Equation (33).