Extending Context Length in Attention Mechanisms with Fourier Features

Anonymous authors
Paper under double-blind review

Abstract

This paper explores the use of Fourier features to extend the context length in attention mechanisms within deep learning models. By leveraging the properties of Fourier transforms, we provide a framework for understanding and implementing context length extension through linear interpolation and change of base techniques. This approach offers insights into model behavior and guides the design of more effective architectures.

1 Introduction

Fourier features have shown promise in improving model performance, particularly in the context of extending the effective context length in attention mechanisms. This paper explores how Fourier features can be utilized to achieve context length extension, revealing a perspective that enhances our understanding of modern deep learning models.

1.1 Background: Fourier Features in Attention Mechanisms

The challenge of modeling long contexts in Transformers is primarily due to the limitations of the attention mechanism. Attention operates over C embeddings $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_C]^\top \in \mathbb{R}^{C \times d}$, where d is the model dimension. Fourier features offer a powerful tool for enhancing these mechanisms by encoding positional information through frequency-based transformations. Learned weight matrices $\mathbf{W}_v \in \mathbb{R}^{d \times d_k}$, $\mathbf{W}_q \in \mathbb{R}^{d \times d_k}$, and $\mathbf{W}_k \in \mathbb{R}^{d \times d_k}$ are used to transform these inputs, where d_k is the projected hidden dimension. The attention mechanism computes the attention matrix and applies it to produce a weighted sum of the value vectors:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathbf{A}\mathbf{V} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}. \tag{1}$$

Basic attention was originally defined with: $\mathbf{Q} = \mathbf{X}\mathbf{W}_q, \mathbf{K} = \mathbf{X}\mathbf{W}_k, \mathbf{V} = \mathbf{X}\mathbf{W}_v$. However, this approach does not directly encode the relative position of keys and values, which is where Fourier features come into play. By leveraging the properties of Fourier transforms, we can encode positional information more effectively.

Rotary Position Embeddings (RoPE) (Su et al., 2024) utilize Fourier features by applying a phase rotation to each element of the embedding vectors. Formally, we define a general transformation $\mathbf{f}$:

$$\mathbf{f}_{\mathbf{W}}(\mathbf{x}_i, \boldsymbol{\theta}) = \mathbf{R}(\boldsymbol{\theta}, i)\mathbf{W}^\top \mathbf{x}_i \tag{2}$$

Here $\mathbf{x}_i \in \mathbb{R}^{d_k}$ is an embedding for position i, $\mathbf{W}$ is a projection matrix, and $\boldsymbol{\theta} \in \mathbb{R}^{d_k/2}$ is a frequency basis. The function is defined based on the rotary position matrix:

$$\mathbf{R}(\boldsymbol{\theta}, i) = \begin{pmatrix} e^{ii\theta_1} & 0 & \cdots & 0 & 0 \\ 0 & e^{ii\theta_1} & \cdots & 0 & 0 \\ \vdots & & & & \\ 0 & 0 & \cdots & e^{ii\theta_{\frac{d_k}{2}}} & 0 \\ 0 & 0 & \cdots & 0 & e^{ii\theta_{\frac{d_k}{2}}} \end{pmatrix} \tag{3}$$

Additionally, using Euler's theorem, we can express this as:

$$\mathbf{R}(\boldsymbol{\theta}, i) = \begin{pmatrix} e^{ii\theta_1} & 0 & \cdots & 0 & 0 \\ 0 & e^{ii\theta_1} & \cdots & 0 & 0 \\ \vdots & & & & \\ 0 & 0 & \cdots & e^{ii\theta_{\frac{d_k}{2}}} & 0 \\ 0 & 0 & \cdots & 0 & e^{ii\theta_{\frac{d_k}{2}}} \end{pmatrix} \tag{4}$$

Due to the arrangement of frequencies, this matrix has the property that $\mathbf{R}(\boldsymbol{\theta}, n - m) = \mathbf{R}(\boldsymbol{\theta}, m)^\top \mathbf{R}(\boldsymbol{\theta}, n)$. We redefine the query-key product between two positions m and n as,

$$\begin{aligned} \mathbf{q}_m^\top \mathbf{k}_n &= \mathbf{f}_{\mathbf{W}_q}(\mathbf{x}_m, m, \boldsymbol{\theta})^\top \mathbf{f}_{\mathbf{W}_k}(\mathbf{x}_n, n, \boldsymbol{\theta}) && (5) \\ &= \left(\mathbf{R}(\boldsymbol{\theta}, m)\mathbf{W}_q^\top \mathbf{x}_m\right)^\top \left(\mathbf{R}(\boldsymbol{\theta}, n)\mathbf{W}_k^\top \mathbf{x}_n\right) && (6) \\ &= \mathbf{x}_m^\top \mathbf{W}_q \mathbf{R}(\boldsymbol{\theta}, n - m)\mathbf{W}_k^\top \mathbf{x}_n && (7) \end{aligned}$$

In this way, the relative positional information $n - m$ is implicitly injected into the query and key product, thus the attention score.

The standard RoPE transformation utilizes:

$$\mathbf{f}_{\mathbf{W}}^{\mathrm{RoPE}}(x_i) = \mathbf{f}(x_i, \boldsymbol{\theta}^d). \tag{8}$$

where $\boldsymbol{\theta}_j^d = b^{-\frac{2j}{d_k}}$ and base $b = 10000$.

1.2 Adjusting Frequency Base of RoPE for Long Context Extensions

Recent papers focus on modifying the RoPE equation to extend length. We consider four methods: Position Interpolation (PI) (Chen et al., 2023a), NTK-RoPE (emozilla, 2023), YaRN (Peng et al., 2023) and CLEX (Chen et al., 2024). In this section we assume that the goal is to extend a method trained to work with C positions to instead work with $C' >> C$. We refer to the ratio $\alpha = \frac{C'}{C}$ as the *scale factor*.

Linear Position Interpolation (PI) involves down-scaling the frequency coefficient i to align with the original context size. PI has been integrated into many open-source models such as LLaMA-2-7B-32K (Together.AI, 2023), Vicuna-7B-v1.5 (Chiang et al., 2023), and LongAlpaca (Chen et al., 2023b). PI works by interpolating the position for a token originally at position i by a scale factor to be at pseudo-position $\frac{i}{\alpha}$. PI can be implemented by changing the frequency base $\boldsymbol{\theta}$:

$$\mathbf{f}_{\mathbf{W}}^{\mathrm{PI}}(x_i) = \mathbf{f}(x_i, \alpha^{-1} \odot \boldsymbol{\theta}^d). \tag{9}$$

Neural Tangent Kernel Interpolation RoPE (NTK-RoPE) addresses the limitations of Linear Position Interpolation (PI) by adjusting the frequency base more dynamically. While PI scales the position linearly, it may not adequately capture the non-linear relationships in extended contexts. NTK-RoPE modifies the base frequency of the rotation in RoPE by changing $\boldsymbol{\theta}_j^d = b^{-\frac{2j}{d_k}}$ to $\boldsymbol{\theta}_j^d = (b\kappa)^{-\frac{2j}{d_k}}$. This adjustment allows NTK-RoPE to better preserve the relative positional encoding over longer sequences.

When $j = \frac{d_k}{2} - 1$, NTK-RoPE sets $(b\kappa)^{-\frac{2j}{d_k}} = \frac{1}{\alpha}(b^{-\frac{2j}{d_k}})$. Hence, κ can be derived as $\kappa = (\frac{C'}{C})^{\frac{d_k}{d_k-2}} = \alpha^{\frac{d_k}{d_k-2}}$. This derivation shows how NTK-RoPE compensates for the linear scaling limitation by introducing a non-linear scaling factor, κ, which adapts based on the context length extension. NTK-RoPE can be written as

$$\mathbf{f}_{\mathbf{W}}^{\text{NTK-RoPE}}(\mathbf{x}_i) = \mathbf{f}(\mathbf{x}_i, \alpha^{-\frac{2j}{d_k-2}} \odot \boldsymbol{\theta}^d), \tag{10}$$

An extension to this approach, Dynamic NTK-RoPE suggests that instead of using a fixed scale factor α during inference, the formula should adapt to the current context length for a specific example. One implementation updates the scale factor to $\alpha_{\text{test}} = \frac{sC_{\text{test}}}{C} - (s-1)$, where $s \in [1, \frac{C'}{C}]$ is a hyperparameter used to adjust α_{test} and C_{test} is maximum observed length during training or inference (Fu et al., 2024).

Another approach from (Fu et al., 2024), sets $\alpha_{\text{test}} = s \cdot \frac{\max(C', C_{\text{test}})}{C} - (s-1)$, where s is set to $\frac{C'}{2C}$ during training and inference.

References

Guanzheng Chen, Xin Li, Zaiqiao Meng, Shangsong Liang, and Lidong Bing. Clex: Continuous length extrapolation for large language models, 2024. URL https://arxiv.org/abs/2310.16450.

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

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

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/.

emozilla. Dynamically Scaled RoPE further increases performance of long context LLaMA with zero fine-tuning, 2023. URL https://www.reddit.com/r/LocalLLaMA/comments/14mrgpr/dynamically_scaled_rope_further_increases/.

Yao Fu, Rameswar Panda, Xinyao Niu, Xiang Yue, Hannaneh Hajishirzi, Yoon Kim, and Hao Peng. Data engineering for scaling language models to 128k context, 2024.

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

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

Together.AI. Llama-2-7b-32k-instruct — and fine-tuning for llama-2 models with together api, 2023. URL https://www.together.ai/blog/llama-2-7b-32k-instruct.