DiJiang: Efficient Large Language Models through Compact Kernelization

Hanting Chen [*1] **Zhicheng Liu** [*1] **Xutao Wang** [1] **Yuchuan Tian** [2] **Yunhe Wang** [1]

{chenhanting,yunhe.wang}@huawei.com;

Abstract

In an effort to reduce the computational load of Transformers, research on linear attention has gained significant momentum. However, the improvement strategies for attention mechanisms typically necessitate extensive retraining, which is impractical for large language models with a vast array of parameters. In this paper, we present DiJiang, a novel Frequency Domain Kernelization approach that enables the transformation of a pre-trained vanilla Transformer into a linear complexity model with little training costs. By employing a weighted Quasi-Monte Carlo method for sampling, the proposed approach theoretically offers superior approximation efficiency. To further reduce the training computational complexity, our kernelization is based on Discrete Cosine Transform (DCT) operations. Extensive experiments demonstrate that the proposed method achieves comparable performance to the original Transformer, but with significantly reduced training costs and much faster inference speeds. Our DiJiang-7B achieves comparable performance with LLaMA2-7B on various benchmark while requires only about 1/50 training cost. Code is available at https://github.com/YuchuanTian/DiJiang.

1. Introduction

The Transformer architecture (Vaswani et al., 2017) has revolutionized the field of Natural Language Processing (NLP), achieving outstanding results in various tasks such as speech recognition (Dong et al., 2018), machine translation (Wang et al., 2019), and document generation/summarization (Kim et al., 2022). This success has led to an era dominated by large language models (LLMs), where the Transformer structure is scaled up to handle increasingly complex tasks. However, this scaling brings with it substantial computational demands, especially due to the attention mechanism which requires cross-correlation calculations between each token. These computational requirements, coupled with the significant inference costs and energy consumption, present considerable obstacles to deploying these models in resource-constrained environments like mobile devices and robotics.

In response to the pressing need for more efficient Transformer models, the research community has directed its efforts towards optimizing the Transformer architecture. A myriad of strategies has been put forward, encompassing methods such as model pruning, quantization, and the development of more efficient attention mechanisms. Among these initiatives, simplifying the attention mechanism has emerged as a particularly promising avenue. This approach focuses on transforming the traditionally quadratic complexity of attention mechanisms into a more manageable linear scale. (Katharopoulos et al., 2020) introduces Linear Transformers, which leverage kernel feature maps to transform self-attention, reducing complexity from quadratic to linear while maintaining comparable results to traditional Transformers. (Kitaev et al., 2020) proposes replacies dot-product attention with locality-sensitive hashing and using reversible residual layers to minimize memory usage in training. Performer (Choromanski et al., 2020) utilize positive orthogonal random features to approximate softmax-based self-attention in Transformers, achieving a transformative leap to linear complexity.

However, the majority of existing methods for optimizing Transformers, particularly in relation to their attention mechanisms, necessitate comprehensive retraining. This retraining process presents a formidable challenge, especially for models with an immense array of parameters. It requires a significant investment in terms of computational resources and time. For instance, the training of a large model like LLaMA-7B (Touvron et al., 2023) demands approximately 82,432 GPU-hours and incurs a total power consumption of around 36 MWh. Undertaking such extensive retraining for models of this magnitude is not only economically taxing

[*]Equal contribution [1]Huawei Noah's Ark Lab [2]Peking University. Correspondence to: Yunhe Wang <yunhe.wang@huawei.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

but also raises environmental concerns due to the substantial energy expenditure involved. This underscores the need for more efficient approaches to adapt and optimize these large-scale models. Despite few research (Zheng et al., 2023; Choromanski et al., 2020) efforts focusing on finding fast approximations for attention mechanisms, these methods have not been thoroughly validated in large-scale language models.

To address the issue of fast attention approximations in large language models, we conducted a thorough analysis of existing linear attention schemes. We discovered that the main source of approximation error in these methods is due to sampling based on the Monte Carlo method. Consequently, we propose the use of weighted Quasi-Monte Carlo sampling for mapping, specifically introducing Frequency Domain Kernelization. This approach efficiently and accurately maps the queries and keys of a Transformer to the frequency domain using Discrete Cosine Transform (DCT). This mapping allows us to effectively eliminate the softmax operation in the attention mechanism, rendering the attention computation linear in complexity, which is shown in Figure 1. We theoretically demonstrate that this frequency domain mapping is an approximate equivalent to the original attention mechanism. Our experiments show that our method achieves performance comparable to the original Transformer with a significantly smaller training cost ($< 1/10$), while also benefiting from faster inference speeds (up to about 10x).

2. Related Works

2.1. Linear Transformers

Reducing the computational load of attention in Transformers remains a hot topic in research (Xu et al., 2023; Tang et al., 2020). (Child et al., 2019) achieved this by sparsifying attention, thereby reducing its computational cost. Similarly, (Kitaev et al., 2020) used locality-sensitive hashing to expedite the computation of attention. However, these methods are hard to apply in auto-regressive Transformer models. As a result, there has been a series of works focusing on removing or substituting the softmax in attention. Notably, the Linear Transformer, first introduced by (Katharopoulos et al., 2020), represents a significant stride in this direction. (Qin et al., 2022) approximated attention calculations using a linear operator and a cosine-based distance reweighting. (Zhai et al., 2021) achieved linear complexity in Transformers by preprocessing keys and values. (Lu et al., 2021) used Gaussian kernel functions in place of dot-product similarity, allowing for the approximation of the full self-attention matrix through low-rank matrix decomposition. (Bello, 2021) bypassed the need for attention calculations by capturing interactions through transforming available contexts into linear functions and applying them to each input, showcasing the variety of methods explored to optimize attention mechanisms in Transformer models.

Additionally, recent proposals like RWKV (Peng et al., 2023), RetNet (Sun et al., 2023), and Mamba (Gu & Dao, 2023) have introduced potential alternatives to the Transformer with linear complexity. However, these existing improvements typically require significant modifications to the model's architecture and often necessitate training a new model from scratch to achieve optimal performance. Given the substantial training costs associated with large language models, such retraining is not always feasible. While methods like StreamingLLM (Xiao et al., 2023) or Longformer (Beltagy et al., 2020) can be implemented through fine-tuning, their reliance on window attention compromises their ability to truly model long sequences, leading to a decrease in accuracy. This highlights the challenge of balancing model training efficiency with the ability to maintain high performance in handling long sequences.

2.2. Frequency-based Transformers

A various of research has focused on applying the Transformer architecture in the frequency domain. For instance, FNet (Lee-Thorp et al., 2021) replaces the self-attention in BERT with Fourier Transform, significantly speeding up Transformer computations. A similar concept (Buchholz & Jug, 2022) has been adapted for image processing tasks. DCFormer (Li et al., 2023) proposes a Transformer-based network that learns semantic representations directly from frequency domain representations using Discrete Cosine Transform (DCT). In the realm of video prediction, ideas like the local frequency domain transformer (Farazi et al., 2021) have been introduced. However, applying these concepts to existing decoder-only large language models presents challenges. The auto-regressive inference style of these models makes token-level frequency domain transformations cumbersome. Each new token requires frequency domain transformation in conjunction with all previous tokens, which fails to reduce complexity and undermines the potential efficiency gains of frequency domain approaches in large-scale language models.

2.3. Preliminary

In our study, we begin by revisiting the general form of self-attention (Vaswani et al., 2017). To simplify the notation and focus on the core aspects, we consider the single head form of self-attention and omit normalization factors. The self-attention mechanism is fundamentally composed of queries Q, keys K, and values V, expressed in the formula:

$$\begin{aligned}&\text{Attention}(Q,K,V)=\text{softmax}(QK^\intercal)V,\\&\text{where } Q,K,V\in\mathbb{R}^{n\times d},\end{aligned}\tag{1}$$

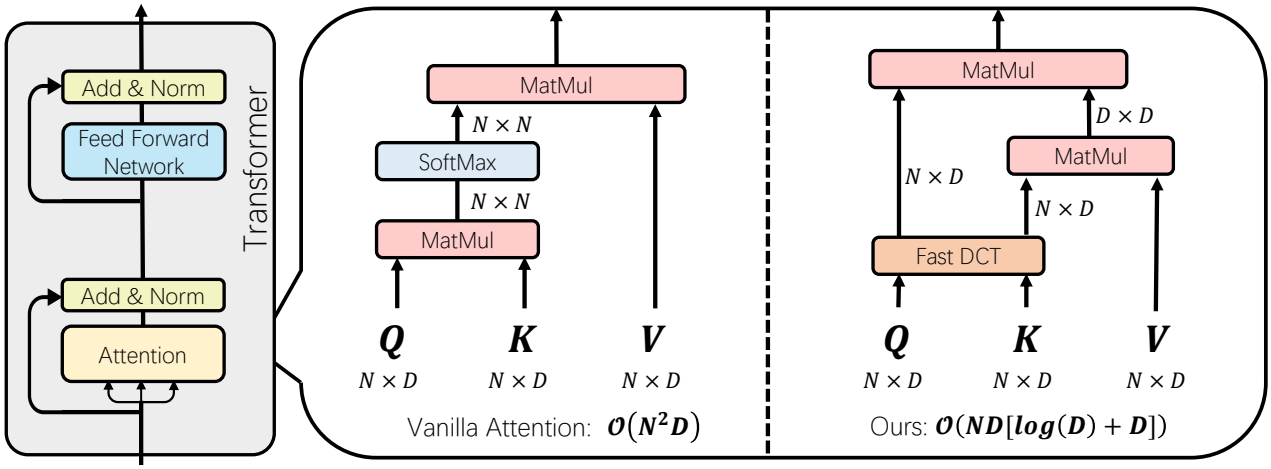


Figure 1. Illustration of the proposed method, where the computation of queries and keys in the attention mechanism of a Transformer is efficiently mapped to the frequency domain using a fast Discrete Cosine Transform (DCT). This mapping effectively eliminates the softmax operation, thereby substantially reducing the computational complexity of the Transformer.

where n denotes the number of tokens and d denotes the hidden dimension of the attention. Specifically, when we denote Q as $(q_1, q_2, ..., q_n)$, K as $(k_1, k_2, ..., k_n)$, V as $(v_1, v_2, ..., v_n)$, and output O as $(o_1, o_2, ..., o_n)$, Equation 1 can be reformulated as:

$$o_i = \sum_{j=1}^{n} \frac{e^{q_i k_j^\intercal}}{\sum_{j'=1}^{n} e^{q_i k_{j'}^\intercal}} v_j, \quad \text{where } q_i, k_i, v_i \in \mathbb{R}^{1\times d}, i = \{1, 2, ..., n\}. \tag{2}$$

It can be observed that the computational and memory complexity for calculating each output in a Transformer model is $\mathcal{O}(nd)$, where n is the sequence length and d is the dimensionality of the representation. Consequently, the time and memory complexity for processing a sentence of length n scales quadratically, becoming $\mathcal{O}(n^2d)$. This quadratic scaling poses a significant computational burden, particularly for longer sequences where n is large, making processing resource-intensive and challenging.

To mitigate this complexity, the concept of a kernel mechanism has been introduced as a means to reduce the computational demands of attention mechanisms, which has been introduced in (Tsai et al., 2019; Katharopoulos et al., 2020; Choromanski et al., 2020). Specifically, this involves the introduction of a kernel function $K(\cdot, \cdot)$, which acts as a positive-definite kernel capable of measuring similarity. By utilizing this kernel, the attention mechanism can be reformulated as:

$$o_i = \sum_{j=1}^{n} \frac{K(q_i, k_j)}{\sum_{j'=1}^{n} \mathcal{K}(q_i, k_{j'})} v_j, \tag{3}$$

By applying the kernel trick, it's possible to linearly decompose the attention mechanism:

$$o_i = \sum_{j=1}^{n} \frac{\phi(q_i)\phi(k_j)^\intercal}{\sum_{j'=1}^{n} \phi(q_i)\phi(k_{j'})^\intercal} v_j, \tag{4}$$

where $\phi(\cdot) : \mathbb{R}^d \to \mathbb{R}^m$ is a projection to map the inputs into m dimension features. This decomposition benefits from the fact that the computational dimensions of the keys and values can be merged, effectively reducing the computational complexity from $\mathcal{O}(n^2d)$ to $\mathcal{O}(nmd)$. Given that the dimensionality d and m is typically much smaller than the sequence length n, this linearization of the attention mechanism results in a substantial decrease in computational intensity.

In the context of large language models, the cost of retraining is prohibitively high. In such scenarios, it becomes imperative to find a kernel that can equivalently replace the vanilla attention mechanism without necessitating extensive retraining. Positive Random Features (PRF) (Choromanski et al., 2020) emerge as a viable candidate in this regard:

$$\phi_{\text{PRF}}(x) = e^{\omega x^\intercal - \frac{\|x\|^2}{2}}, \tag{5}$$

where $\omega \in \mathbb{R}^{m\times d}$. Theoretical demonstrations have established that $e^{qk^\intercal} = \mathbb{E}_{\omega\sim\mathcal{N}(0.I)}[e^{\omega q^\intercal - \frac{\|q\|^2}{2}} e^{\omega k^\intercal - \frac{\|k\|^2}{2}}]$. It means that when m, the dimension of the feature space, is sufficiently large, Positive Random Features (PRF) mapping becomes an equivalent of the original attention mechanism. This equivalence suggests that, in theory, it is feasible to directly transform existing vanilla attention into linear attention using PRF mapping, thereby achieving an acceleration without loss of functionality. However, a notable challenge arises due to the need for m to be set to a significantly large value to maintain the performance by reducing the approximation error. This requirement leads to a non-negligible

increase in computational demand. For instance, in the case of the Performer (Choromanski et al., 2020), to achieve a lossless linear attention, m often needs to be set to larger than d, diminishing the benefits of reduced computational load brought by linear attention.

2.4. Kernelized Attention in Frequency Domain

To address this issue, we first conduct a theoretical analysis of the kernel-based approach for approximating attention mechanisms. We begin with the application of Bochner's Theorem. This theorem allows us to equate the original attention computation involving queries (Q) and keys (K) – specifically the Gaussian kernel – to an integral computation akin to Equation 4.

Theorem 2.1. *(Bochner's Theorem) (Feller, 1966). A continuous shift invariant scaled kernel function $K(x, z) : \mathbb{R}^d \to R$ is positive definite if and only if it is the Fourier Transform of a unique finite probability measure p on $\mathbb{R}^d$.*

$$K(x,z) = \int_{\mathbb{R}^d} e^{i(x-z)^\intercal w} p(w)dw = E_{w\sim p(\cdot)}[e^{iw^\intercal x}(e^{iw^\intercal z})^*], \tag{6}$$

where the symbol z^* denotes the complex conjugate of z.

According to Bochner's theorem, there is a one-to-one correspondence between the kernel function $K(x, z)$ and the probability density $p(w)$ defined on $\mathbb{R}^d$. Monte Carlo is equal weight approximation to kernel integrals. Taking $\varphi_p(x) := \frac{1}{\sqrt{m}}[e^{-iw_1^\intercal x}, ..., e^{-iw_m^\intercal x}]^\intercal$, the feature maps can be constructed as:

$$K(x,z) = E_{w\sim p(\cdot)}[e^{iw^\intercal x}(e^{iw^\intercal z})^*] \approx \varphi_p(x)^\intercal \varphi_p^*(z), \tag{7}$$

where $w_i \sim p(\cdot)$ are samples constructed by Monte Carlo methods. $\varphi_p(\cdot)$ is the explicit finite dimensional feature map, which depends on the kernel K. Moving forward, instead of employing the Monte Carlo method as suggested in (Choromanski et al., 2020), we utilize the Quasi-Monte Carlo method (Le et al., 2013). This shift enables the estimation of the integral using a specific uniform distribution as opposed to a randomly sampled distribution.

Utilizing Bochner's theorem allows for a transformative interpretation of the attention mechanism in Transformer models. For the Gaussian Kernel:

$$K_G(x,y) := e^{-\frac{\|x-y\|^2}{2}} = e^{-\frac{\|x\|^2+\|y\|^2}{2}} e^{x^\intercal y}, \tag{8}$$

since the x and y in attention mechanism is usually normalized, the Gaussian Kernel can be regarded as $e^{x^\intercal y}$, which is the same as the calculation between the queries and keys.

Theorem 2.2. *The Positive Fixed Features (PFF) is formulated as:*

$$\varphi_{PFF}(x) := \frac{e^{-\|x\|^2}}{\sqrt{m}}[e^{\Phi^{-1}(t_1)x^\intercal v_1}, ..., e^{\Phi^{-1}(t_m)x^\intercal v_m}]^\intercal, \tag{9}$$

where $V = [v_1, ..., v_m] \in \mathbb{S}^{d\times m}$ is asymptotically uniformly distributed and $t_i \sim U(0, 1)$. Then, $\varphi_{PFF}(x)^\intercal \varphi_{PFF}(z)$ is an unbiased estimate of Gaussian kernel $K_G(x, y)$.

The proof of this theorem involves a transformation to spherical coordinates, which can be found in the supplementary material. Through this transformation, we demonstrate that an approximation based on any asymptotically uniformly distribution can closely approximate the original Gaussian kernel. Furthermore, according to (Asmussen & Glynn, 2007), when utilizing uniform sequences, the Quasi-Monte Carlo method can offer superior approximation efficiency compared to the traditional Monte Carlo method. The approximation efficiency of Quasi-Monte Carlo is $\mathcal{O}(1/m)$, which is more favorable than the $\mathcal{O}(1/m^{-0.5})$ efficiency of Monte Carlo. Consequently, this implies that using the PFF 9 kernel for approximating the Gaussian kernel is more advantageous than the PRF kernel in Equation 5.

Theorem 2.3. *The Weighted Positive Fixed Features (WPFF) is formulated as:*

$$\varphi_{WPFF}(x) := \frac{De^{-\|x\|^2}}{\sqrt{m}}[e^{\Phi^{-1}(t_1)x^\intercal v_1}, ..., e^{\Phi^{-1}(t_m)x^\intercal v_m}]^\intercal, \tag{10}$$

where D is a learnable parameter which can be optimized by the input x. Then the upper bound of the integral estimation error of the objective function by WPFF (Weighted Positive Fixed Features) method is not greater than the upper bound of the integral estimation error of the objective function by PFF (Positive Fixed Features) method.

Building upon the Quasi-Monte Carlo foundation, we further introduce the concept of weighted Quasi-Monte Carlo to enhance the efficiency of approximation. This advancement aims to leverage the strengths of the Quasi-Monte Carlo method, augmenting it with strategically weighted sampling to improve the precision and convergence rates of our approximations. The detailed proof is provided in the supplementary materials.

To further accelerate the training speed, we propose the use of frequency domain transformations to reduce the required computational resources. Fast Fourier Transform (FFT) and Discrete Cosine Transform (DCT) are commonly used methods for such transformations. Compared to ordinary orthogonal transformations, frequency domain transformations have algorithms for rapid computation, significantly reducing the computational cost of our proposed mapping. Specifically, the complexity of $\mathcal{O}(m)$ can be reduced to $\mathcal{O}(\log(m))$. Additionally, since DCT operates in the real number domain, it demands even less computational resources and is more hardware-friendly. Therefore, we opt for the DCT to carry out our kernel mapping.

Specifically, a DCT coefficient $\mathcal{C} \in \mathbb{R}^{d\times d}$ in the frequency

Algorithm 1 Frequency domain kernelization for efficient language models.

input A small amount of data x_i, a pre-trained Transformer model M.
1. Initialization: the DCT coefficient $\mathcal{C}$, the weight D, the diagonal matrix T in Equation 12 for each layer in M.
2. Transformation: transform the vanilla attention calculation $\text{Attention}(Q, K, V) = \text{softmax}(QK^\intercal)V$ to $\text{FKA}(Q, K, V) = \phi_{\text{WDCF}}(Q)\phi_{\text{WDCF}}(K)^\intercal V$ using the Weighted Discrete Cosine Features for each layer in M.
3. Get the transformed model M_{FKA}.
repeat
4. Randomly select a batch of data from x_i.
5. Employ the transformed model M_{FKA} on the mini-batch.
6. Update weights in M_{FKA} according to the loss and gradient;
until convergence.
output An efficient language model M_{FKA}.

domain is defined as:

$$\mathcal{C}_{j_1 j_2} = s_{j_1} s_{j_2} \sum_{i_1=0}^{n-1} \sum_{i_2=0}^{d-1} \cos\left(\frac{\pi(2i_1+1)j_1}{2d}\right) \cos\left(\frac{\pi(2i_2+1)j_2}{2d}\right), \tag{11}$$

where $s_j = \sqrt{1/d}$ if $j = 0$ and $s_j = \sqrt{2/d}$ otherwise. The weighted mapping using DCT (which is called Weighted Discrete Cosine Features) can be reformulated as:

$$\phi_{\text{WDCF}}(x) = De^{T\mathcal{C}x^\intercal}, \tag{12}$$

where $\mathcal{C} \in \mathbb{R}^{m\times d}$ is the DCT coefficient, $D \in \mathbb{R}^m$ is a learnable weight, and $T = \text{diag}(t_1, \ldots, t_m)$ is a random diagonal matrix following the inverse cumulative distribution. Note that since the x in attention mechanism is usually normalized, we ignore the term of $\|x\|^2$ in Equation 9 for efficiency. Therefore, using DCT as a kernel can closely approximate the original attention mechanism while have low computation complexity. For scenarios where $m > d$, more DCT transformations can be derived using different boundary conditions. Details can be referred to (Ahmed et al., 1974). It is noted that we set $m = d$ to avoid increasing computational complexity in the subsequent experiments.

Therefore, the kernelized attention in frequency domain (FKA) is then reformulated as:

$$\begin{aligned} &\text{FKA}(Q, K, V) = \phi_{\text{WDCF}}(Q)\phi_{\text{WDCF}}(K)^\intercal V, \\ &\text{where } Q, K, V \in \mathbb{R}^{n\times d}, \end{aligned} \tag{13}$$

This approach achieves a notable reduction in computational complexity by employing the Discrete Cosine Transform (DCT) to map the queries and keys within the Transformer's attention mechanism to a domain where operations are inherently more efficient.

In summary, our method leverages frequency domain kernelization for Transformer attention mechanisms, significantly cutting computational costs while either preserving or enhancing model performance. The details are shown in Algorithm 1. Through the strategic use of the weighted Quasi-Monte Carlo method, which outperforms traditional Monte Carlo sampling in efficiency and accuracy, combined with DCT for efficient frequency domain transformations, we attain linear complexity in attention computation. This reformulation not only improves the scalability of Transformers, enabling them to handle larger datasets and extended sequences with ease, but also markedly accelerates the training and inference phases.

Discussion Existing frequency domain Transformer approaches implement FFT transformations along the n dimension to reduce the computational complexity of attention from $o(n^2d)$ to $o(n\log(n)d)$. However, the autoregressive inference style of these models makes token-level frequency domain transformations cumbersome. Specifically, during training and inference, each token can only access information from preceding tokens, not allowing for frequency domain transformations with subsequent tokens. This results in a scenario where each new token requires a frequency domain transformation in conjunction with all previous tokens, failing to reduce complexity and undermining the potential efficiency gains of frequency domain approaches in large-scale language models. Therefore, existing frequency domain Transformer approaches are not feasible for application in LLMs. In contrast, our approach conducts frequency domain transformations along the d dimension and leverages the properties of linear Transformers, reducing the computational complexity of attention from $o(n^2d)$ to $o(nd\log(d))$, offering superior computational complexity and accuracy comparable to the original Transformer.

3. Experiments

In this section, we conduct extensive experimental validation of the proposed architecture, encompassing results across language models of varying scales. Additionally, we provide detailed analyses to substantiate the effectiveness of our approach.

3.1. Evaluation on Different Scales

Given the challenge of replicating the training processes of most language models, as only their checkpoints are openly available, we opted to validate our method using Pythia (Biderman et al., 2023), a model with a fully public dataset and training procedure, enabling fair comparisons.

We adhered to the exact training settings employed by Pythia, including learning rates, optimizers, and other hyperparameters, and utilized the Pile dataset. The Pile (Gao et al.,

Table 1. The experimental results of the proposed method. Training time is measured using A800. Inference throughput is evaluated with token length of 2048.

Model	PIQA	WinoGrande	WSC	ARC-E	ARC-C	LogiQA	Avg	Training (day)	Inference (tokens/s)
Pythia-70M	0.498	0.484	0.596	0.25	0.221	0.202	0.375	21.3	2037
DiJiang-70M	0.587	0.511	0.365	0.403	0.213	0.253	0.389	1.3	2605
Pythia-160M	0.532	0.484	0.634	0.265	0.227	0.202	0.391	42.9	622
DiJiang-160M	0.618	0.490	0.384	0.439	0.217	0.239	0.398	2.7	1315
Pythia-410M	0.668	0.537	0.567	0.521	0.213	0.22	0.454	105.8	203
DiJiang-410M	0.663	0.524	0.567	0.492	0.244	0.247	0.456	6.6	787
Pythia-1B	0.706	0.533	0.365	0.569	0.269	0.296	0.456	201.2	105
DiJiang-1B	0.677	0.521	0.365	0.537	0.253	0.284	0.440	12.6	611
Pythia-2.8B	0.737	0.596	0.384	0.640	0.295	0.215	0.478	593.3	34
DiJiang-2.8B	0.713	0.545	0.413	0.597	0.289	0.279	0.473	37.1	284
OPT-350M	0.645	0.524	0.365	0.441	0.208	0.210	0.399	-	201
DiJiang-350M	0.550	0.507	0.635	0.286	0.227	0.223	0.404	5.6	820
TinyLLaMA-1.1B	0.666	0.541	0.413	0.487	0.211	0.228	0.424	-	74
DiJiang-1.1B	0.535	0.508	0.635	0.286	0.243	0.212	0.403	13.9	613

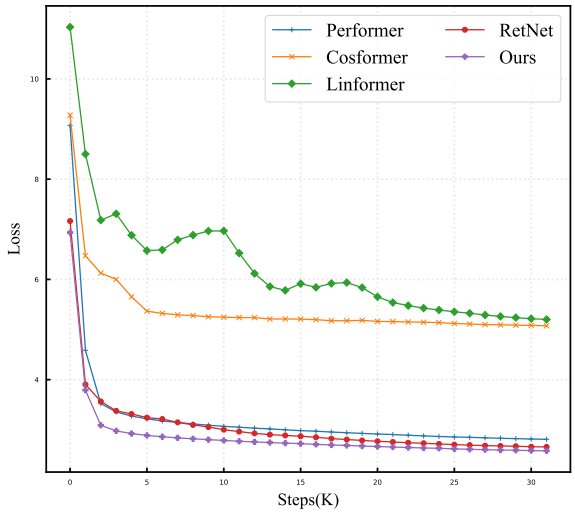


Figure 2. Training Curve of different methods. The proposed method achieves the lowest PPL and the fastest converge speed.

2020) is an 825 GiB corpus of English text, specifically designed for training large-scale language models. This project is composed of 22 distinct, high-quality subsets, both pre-existing and newly constructed, many of which originate from academic or professional sources. This comprehensive and diverse dataset serves as a robust foundation for developing and fine-tuning language models Our DiJiang model was fine-tuned from the pre-trained Pythia model. We evaluated our approach on six public datasets used by Pythia: PIQA (Bisk et al., 2020), WinoGrande, WSC (Sakaguchi et al., 2021), ARC-E, ARC-C (Clark et al., 2018), and LogiQA (Liu et al., 2020). The Pythia model's checkpoint was obtained from HuggingFace[1]. We adapt the learned gating mechanism (Peng et al., 2021) similar with the RetNet (Sun et al., 2023) to augment our DiJiang.

The experimental results, as shown in Table 1, indicate that our method achieved remarkable outcomes across different model sizes, ranging from 70M to 2.8B parameters. On average, the performance on the six datasets was nearly identical to that of the original Pythia, but with only $\sim 1/16$ of the training cost. Furthermore, the inference speed of our DiJiang model was significantly faster than that of the original Pythia. These results substantiate the effectiveness of our approach, demonstrating its potential to enhance the efficiency of large language models without compromising performance.

3.2. Evaluation on Different Models

To evaluate the effectiveness of our method across different models, as shown in Table 1, we further applied our approach to the OPT-350M (Zhang et al., 2022)[2] and TinyLLaMA-1.1B[3] models. It's important to note that since their training data are not fully accessible, we continued to use the Pile dataset for fine-tuning them.

[1] https://huggingface.co/EleutherAI
[2] https://huggingface.co/facebook/opt-350m
[3] https://huggingface.co/TinyLlama/TinyLlama-1.1B-python-v0.1

Table 2. Comparison of different linear attention models on fine-tuning Pythoia-410M (Biderman et al., 2023).

Model	PIQA	WinoGrande	WSC	ARC-E	ARC-C	LogiQA	Avg
Pythia-410M (Biderman et al., 2023)	0.668	0.537	0.567	0.521	0.213	0.22	0.454
Linformer (Wang et al., 2020)	0.5267	0.5114	0.6346	0.2656	0.244	0.2074	0.3982
Cosformer (Qin et al., 2022)	0.5218	0.5059	0.6058	0.2673	0.2637	0.2642	0.4047
Performer (Choromanski et al., 2020)	0.6431	0.4964	0.4327	0.4701	0.2312	0.2366	0.4183
RetNet (Sun et al., 2023)	0.4951	0.4957	0.6346	0.2508	0.227	0.2028	0.3843
PFF (Equation 9)	0.6453	0.4996	0.4712	0.4747	0.2295	0.2381	0.4264
DiJiang (Ours)	0.6638	0.5241	0.5673	0.4928	0.2449	0.2473	**0.4567**

Table 3. Comparison with LLaMA2-7B on various benchmarks.

Model	PIQA	SIQA	BoolQ	WSC	HellaSwag	ARC-E	ARC-C	MMLU	NQ	COPA	Race-Middle	Avg	Tokens
LLaMA2-7B	0.782	0.485	0.749	0.663	0.740	0.561	0.403	0.468	0.192	0.670	0.402	0.565	2000B
DiJiang-7B	0.775	0.346	0.626	0.683	0.694	0.626	0.427	0.407	0.194	0.730	0.618	0.557	40B

Finally, we conducted further experiments on the well-known publicly available large language model, LLaMA2-7B, fine-tuning it into the DiJiang-7B model. Table 3 reveal that the DiJiang-7B model achieves results that are virtually identical to the original LLaMA2-7B across various benchmarks. Remarkably, our model required only 40B training data, significantly less than the 2T tokens used by LLaMA2-7B. This demonstrates the successful application of our method to large-scale models at the 7B parameter level, highlighting the efficiency and effectiveness of our fine-tuning approach even when scaling to vast model sizes.

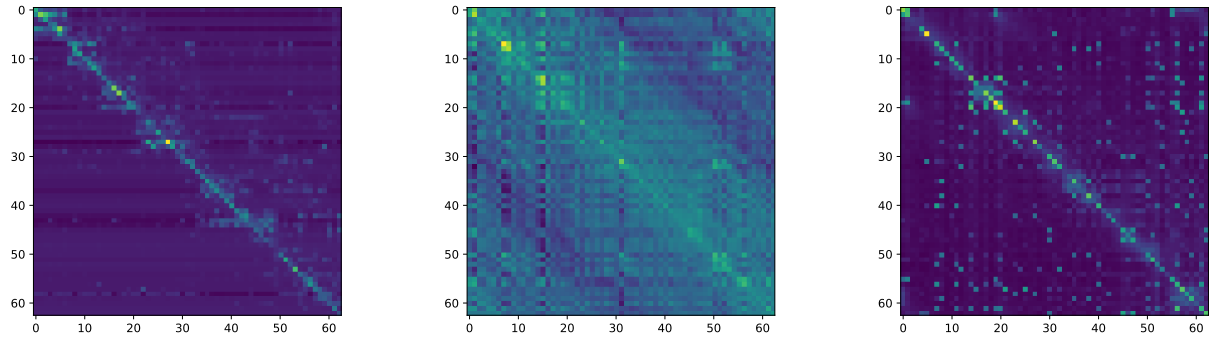
Interestingly, we found that despite using a limited dataset, our method achieved results similar to the original models with a significantly lower training cost and faster speed. This outcome further demonstrates the strong generalizability and flexibility of our approach, underscoring its potential applicability across a broad spectrum of language models, even in scenarios where the original training datasets are not available.

3.3. Comparison with Linear Transformers

To compare the superiority of our approach against other linear-complexity self-attention Transformer models, we validated the fine-tuning results on Pythia-400M for different models including Linformer, Performer, RetNet, and Cosformer. For a fair comparison, we employed the same training settings and data. Table 2 displays the comparative results, revealing that while existing methods can achieve good results through retraining, as evidenced by their original publications, most of them suffer from significant accuracy losses in scenarios where fine-tuning is done without retraining. This is largely because these methods struggle to accurately approximate the original attention mechanism, leading to an inability to restore the original accuracy with minimal training.

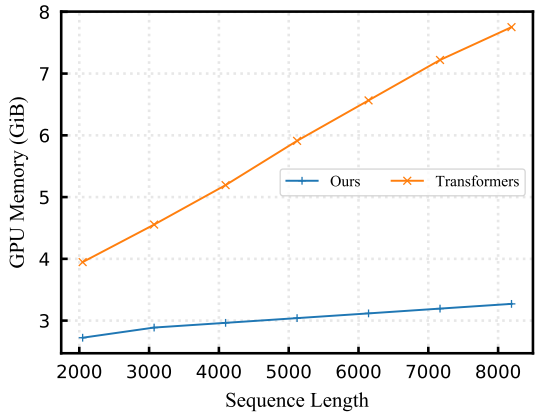
Among these comparison methods, Performer achieved the best results by approximating the original attention with Positive Random Features (PRF). However, as previously discussed, this Monte Carlo-based approximation method cannot achieve satisfactory outcomes, resulting in accuracy loss. By switching from Monte Carlo to the Quasi-Monte Carlo scheme using Positive Fixed Features (PFF) as described in Equation 9, we surpassed the accuracy of Performer but still fell short of the original vanilla Transformer's performance. Furthermore, by incorporating the Discrete Cosine Transform (DCT), our method achieves higher efficiency than approaches using PFF kernels. The DCT transformation enables a more compact and efficient representation of the frequency components of the attention mechanism. This efficiency stems from the DCT's ability to concentrate energy, allowing for a sparse representation that captures the most significant features of the data with fewer coefficients. Consequently, our approach not only closely approximates the original attention but also does so with improved computational performance compared to PFF-based methods. This advantage highlights the effectiveness of using DCT in optimizing the approximation of attention mechanisms, further underscoring the potential of our method in enhancing the efficiency of Transformer models. Further incorporating weighted Quasi-Monte Carlo, our DiJiang architecture ultimately achieved accuracy nearly identical to the original Pythia-400M, validating the efficacy of our approximation method. This demonstrates not only the potential of our approach for fine-tuning large-scale language models but also underscores the importance of choosing an efficient approximation strategy to maintain model performance.

We further visualized the training curves to showcase the approximation efficiency of different linear Transformer

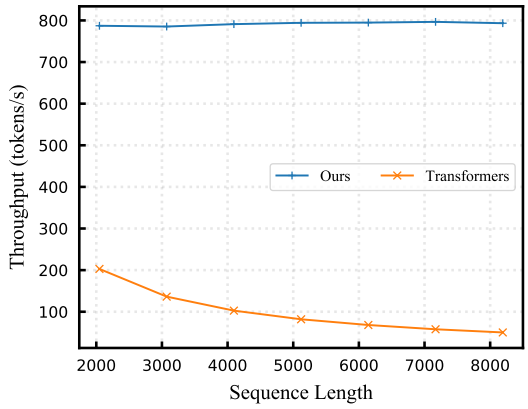


(a) Vanilla attention (b) PRF kernel (c) WDCF kernel

Figure 3. Visualization of attention map of different architectures. The results are averaged by multiple heads.



(a) Inference Memory



(b) Inference Throughput

Figure 4. Comparison of inference memory and throughput between the proposed DiJIang and vanilla Transformer architecture.

models, as depicted in Figure 2. RetNet, as an emerging language model architecture, has shown its potential by achieving significantly low loss values, underscoring its capability for language tasks. Despite its low loss, RetNet does not necessarily outperform on benchmark metrics and, in some cases, even falls short of the results achieved by the Performer. This discrepancy highlights the importance and advantages of employing kernel methods to approximate the original attention computation, particularly in fine-tuning scenarios.

Our method demonstrates the fastest rate of loss reduction and ultimately achieves the lowest loss value. This rapid convergence indicates that our approach can quickly reach a performance level similar to that of the original Transformer. The visualization clearly underscores the superiority of our method in terms of both convergence speed and final model accuracy, validating our approach's effectiveness in efficiently approximating the attention mechanism while maintaining high performance standards. This visual evidence further solidifies our claim that our method stands out among linear Transformer alternatives, offering a compelling solution for optimizing Transformer models without compromising on quality.

3.4. Comparison of Inference Cost

Furthermore, we also evaluated the memory usage and throughput of our method in comparison to the original Transformer model under various conditions. We selected the Pythia-410M model as our primary subject for analysis. We follow the implementation of RetNet (Sun et al., 2023) to efficient inference. The specific results, as depicted in Figure 4, demonstrate that as the token length increases, the memory footprint and inference speed of our model do not escalate. This observation is attributed to the linear complexity characteristic of our approach, indicating that our method is more conducive to long-sequence inference. In contrast, due to the quadratic complexity of attention computations, the original Transformer model experiences a continuous increase in both inference time and required memory as the token length grows. This comparison highlights the

efficiency and practicality of our solution, particularly in scenarios involving extensive sequences where computational resources are a critical concern.

3.5. Visualization

To further demonstrate the effectiveness of our model's approximation of the attention mechanism, we present attention maps generated by different methods in Figure 3. It is evident that the original Transformer's attention map (Figure 3 (a)) is rich in information, laying the foundation for its robust capabilities. In contrast, attention maps produced by other linear attention methods such as Performer (Figure 3 (b)) fail to adequately capture the relationships between tokens, resulting in maps that are dissimilar to those of the original Transformer and ultimately leading to decreased model accuracy, despite fine-tuning efforts. In contrast, our method (Figure 3 (c)), by employing the weighted Quasi-Monte Carlo scheme, closely approximates the original attention mechanism. This allows it to effectively model the relationships between different tokens, achieving results nearly identical to those of the original Transformer but with significantly faster inference efficiency. This comparison not only highlights the inadequacies of other linear attention methods in capturing token interdependencies but also showcases the superiority of our approach in accurately approximating attention while enhancing computational efficiency.

4. Conclusion

This paper introduces DiJiang, a groundbreaking Frequency Domain Kernelization method designed to address the computational inefficiencies inherent in traditional Transformer models. By leveraging linear attention mechanisms and a novel application of the weighted Quasi-Monte Carlo method for efficient sampling, our approach significantly reduces the necessity for extensive retraining. This is particularly beneficial for large language models, where the cost and time associated with training are substantial barriers to progress. The kernelization process, underpinned by Discrete Cosine Transform (DCT), not only diminishes the computational complexity but also ensures that the adaptation from a vanilla Transformer to a linear attention model incurs minimal training costs. Our extensive experiments validate that DiJiang achieves performance on par with conventional Transformers while reducing training costs by about 10x and enhancing inference speeds. This method represents a significant advancement in the development of efficient and scalable Transformer models, promising wider applicability and facilitating advancements in various tasks within the realm of natural language processing and beyond.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Ahmed, N., Natarajan, T., and Rao, K. R. Discrete cosine transform. *IEEE transactions on Computers*, 100(1):90–93, 1974.

Asmussen, S. and Glynn, P. W. *Stochastic simulation: algorithms and analysis*, volume 57. Springer, 2007.

Bello, I. Lambdanetworks: Modeling long-range interactions without attention. *arXiv preprint arXiv:2102.08602*, 2021.

Beltagy, I., Peters, M. E., and Cohan, A. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Biderman, S., Schoelkopf, H., Anthony, Q. G., Bradley, H., O'Brien, K., Hallahan, E., Khan, M. A., Purohit, S., Prashanth, U. S., Raff, E., et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pp. 2397–2430. PMLR, 2023.

Bisk, Y., Zellers, R., Gao, J., Choi, Y., et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Brauchart, J. S. and Grabner, P. J. Distributing many points on spheres: minimal energy and designs. *Journal of Complexity*, 31(3):293–326, 2015.

Buchholz, T.-O. and Jug, F. Fourier image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1846–1854, 2022.

Child, R., Gray, S., Radford, A., and Sutskever, I. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Dong, L., Xu, S., and Xu, B. Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 5884–5888. IEEE, 2018.

Farazi, H., Nogga, J., and Behnke, S. Local frequency domain transformer networks for video prediction. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–10. IEEE, 2021.

Feller, W. *Introduction to Probality Theory and Its Applications*. Wiley Eastern, 1966.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Gu, A. and Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Katharopoulos, A., Vyas, A., Pappas, N., and Fleuret, F. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pp. 5156–5165. PMLR, 2020.

Kim, G., Hong, T., Yim, M., Nam, J., Park, J., Yim, J., Hwang, W., Yun, S., Han, D., and Park, S. Ocr-free document understanding transformer. In *European Conference on Computer Vision*, pp. 498–517. Springer, 2022.

Kitaev, N., Kaiser, Ł., and Levskaya, A. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.

Le, Q., Sarlós, T., Smola, A., et al. Fastfood-approximating kernel expansions in loglinear time. In *Proceedings of the international conference on machine learning*, volume 85, pp. 8, 2013.

Lee-Thorp, J., Ainslie, J., Eckstein, I., and Ontanon, S. Fnet: Mixing tokens with fourier transforms. *arXiv preprint arXiv:2105.03824*, 2021.

Li, X., Zhang, Y., Yuan, J., Lu, H., and Zhu, Y. Discrete cosin transformer: Image modeling from frequency domain. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 5468–5478, 2023.

Liu, J., Cui, L., Liu, H., Huang, D., Wang, Y., and Zhang, Y. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. *arXiv preprint arXiv:2007.08124*, 2020.

Lu, J., Yao, J., Zhang, J., Zhu, X., Xu, H., Gao, W., Xu, C., Xiang, T., and Zhang, L. Soft: Softmax-free transformer with linear complexity. *Advances in Neural Information Processing Systems*, 34:21297–21309, 2021.

Lyu, Y. Spherical structured feature maps for kernel approximation. In *International Conference on Machine Learning*, pp. 2256–2264. PMLR, 2017.

Peloso, M. M. Classical spaces of holomorphic functions. *Lecture notes available on http://www. mat. unimi. it/users/peloso*, 2011.

Peng, B., Alcaide, E., Anthony, Q., Albalak, A., Arcadinho, S., Cao, H., Cheng, X., Chung, M., Grella, M., GV, K. K., et al. Rwkv: Reinventing rnns for the transformer era. *arXiv preprint arXiv:2305.13048*, 2023.

Peng, H., Pappas, N., Yogatama, D., Schwartz, R., Smith, N. A., and Kong, L. Random feature attention. *arXiv preprint arXiv:2103.02143*, 2021.

Qin, Z., Sun, W., Deng, H., Li, D., Wei, Y., Lv, B., Yan, J., Kong, L., and Zhong, Y. cosformer: Rethinking softmax in attention. *arXiv preprint arXiv:2202.08791*, 2022.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Sun, Y., Dong, L., Huang, S., Ma, S., Xia, Y., Xue, J., Wang, J., and Wei, F. Retentive network: A successor to transformer for large language models, 2023.

Tang, Y., Wang, Y., Xu, Y., Tao, D., Xu, C., Xu, C., and Xu, C. Scop: Scientific control for reliable neural network pruning. *Advances in Neural Information Processing Systems*, 33:10936–10947, 2020.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Tsai, Y.-H. H., Bai, S., Yamada, M., Morency, L.-P., and Salakhutdinov, R. Transformer dissection: a unified understanding of transformer's attention via the lens of kernel. *arXiv preprint arXiv:1908.11775*, 2019.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wang, Q., Li, B., Xiao, T., Zhu, J., Li, C., Wong, D. F., and Chao, L. S. Learning deep transformer models for

machine translation. *arXiv preprint arXiv:1906.01787*, 2019.

Wang, S., Li, B. Z., Khabsa, M., Fang, H., and Ma, H. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

Xiao, G., Tian, Y., Chen, B., Han, S., and Lewis, M. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.

Xu, Y., Li, C., Li, D., Sheng, X., Jiang, F., Tian, L., and Sirasao, A. Fdvit: Improve the hierarchical architecture of vision transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5950–5960, 2023.

Yang, J., Sindhwani, V., Avron, H., and Mahoney, M. Quasi-monte carlo feature maps for shift-invariant kernels. In *International Conference on Machine Learning*, pp. 485–493. PMLR, 2014.

Zhai, S., Talbott, W., Srivastava, N., Huang, C., Goh, H., Zhang, R., and Susskind, J. An attention free transformer. *arXiv preprint arXiv:2105.14103*, 2021.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Zheng, L., Yuan, J., Wang, C., and Kong, L. Efficient attention via control variates. *arXiv preprint arXiv:2302.04542*, 2023.

A. Theoretical Proof.

Theorem A.1. *The Positive Fixed Features (PFF) is formulated as:*

$$\varphi_{PFF}(x) := \frac{e^{-\|x\|^2}}{\sqrt{m}}[e^{\Phi^{-1}(t_1)x^\intercal v_1}, ..., e^{\Phi^{-1}(t_m)x^\intercal v_m}]^\intercal, \tag{14}$$

where $V = [v_1, ..., v_m] \in \mathbb{S}^{d\times m}$ is asymptotically uniformly distributed and $t_i \sim U(0,1)$. Then, $\varphi_{PFF}(x)^\intercal \varphi_{PFF}(z)$ is an unbiased estimate of Gaussian kernel $K_G(x,y)$.

Proof. The proof is motivated by (Lyu, 2017). We also use spherical coordinate changes to get the following proof. The Gaussian kernel is real-valued and therefore the imaginary part in Eq.(6) can be discarded.

$$K_G(x,y) = e^{-\frac{\|x-y\|^2}{2}} = \int_{\mathbb{R}^d} e^{i(x-y)^\intercal w}\mu(w)dw = \int_{\mathbb{R}^d} \cos((x-y)^\intercal w)\mu(w)dw = e^{-\|x\|^2-\|y\|^2}\int_{\mathbb{R}^d} e^{((x+y)^\intercal w)}\mu(w)dw, \tag{15}$$

where $\mu(\cdot)$ is the probability density function of d-dimensional standard normal distribution.

The Gaussian kernel is a shift and rotation invariant kernel. Given any rotation $R \in SO(d)$, where $SO(d)$ denotes rotation groups, the corresponding probability density is also Gaussian according to Bochner's theorem. For shift and rotation invariant kernels, we can convert the integral to spherical coordinates. $r = ||w||_2$ and $p(r)$ be the density function of r, and $w = rv$. Because of the rotation invariant property of $K_G(x,y)$, we achieve:

$$K_G(x,y) = \int_{\mathbb{R}^d} e^{i(x-y)^\intercal w}p(w)dw = \int_{R_+}\int_{\mathbb{S}^{d-1}} e^{i(x-y)^\intercal rv}p(r)drd\sigma(v) = \int_{[0,1]}\int_{\mathbb{S}^{d-1}} e^{i(x-y)^\intercal \Phi^{-1}(t)v}dtd\sigma(v), \tag{16}$$

where σ denotes the normalized surface area measure on $\mathbb{S}^d := \{x \in \mathbb{R}^d | \|x\|_2 = 1\}$ and $\Phi^{-1}(t)$ denotes the inverse cumulative distribution function w.r.t is a non-negative radial scale.

For real valued continuous shift and rotation invariant scaled kernel $K_G(x,y)$, the imaginary parts of the integral vanish. We can achieve:

$$K_G(x,y) = \int_{\mathbb{R}^d} e^{i(x-z)^\intercal w}p(w)dw = \int_{\mathbb{R}^d} \cos((x-z)^\intercal w)p(w)dw = \int_{[0,1]}\int_{\mathbb{S}^{d-1}} \cos((x-z)^\intercal \Phi^{-1}(t)v)dtd\sigma(v). \tag{17}$$

For Gaussian kernel, we can get another medium integral form:

$$K_G(x,y) = \int_{\mathbb{R}^d} e^{i(x-y)^\intercal w}\mu(w)dw = e^{-\|x\|^2-\|y\|^2}\int_{\mathbb{R}^d} e^{((x+y)^\intercal w)}\mu(w)dw = e^{-\|x\|^2-\|y\|^2}\int_{[0,1]}\int_{\mathbb{S}^{d-1}} e^{(x+y)^\intercal \Phi^{-1}(t)v}dtd\sigma(v). \tag{18}$$

According to (Brauchart & Grabner, 2015), if the point set $V = [v_1, ..., v_m] \in \mathbb{S}^{d\times m}$ is asymptotically uniformly distributed, the following equation holds true:

$$\lim_{m\to\infty}\frac{1}{m}\sum_{i=1}^{m} f(v_i) = \int_{\mathbb{S}^d} f(v)d\sigma(v). \tag{19}$$

Then, we have:

$$\begin{aligned}
\lim_{m\to\infty} E_{t_i\sim U(0,1)}[\varphi_{\text{PFF}}(x)^\intercal \varphi_{\text{PFF}}(y)] =& E_{t_i\sim U(0,1)}[\lim_{m\to\infty}\frac{e^{-\|x\|^2-\|y\|^2}}{m}\sum_{i=1}^{m} e^{\Phi^{-1}(t_j)(x+y)^\intercal v_i}] \\
=& e^{-\|x\|^2-\|y\|^2}\int_{[0,1]}\int_{\mathbb{S}^{d-1}} e^{(x+y)^\intercal \Phi^{-1}(t)v}dtd\sigma(v) \\
=& e^{-\|x\|^2-\|y\|^2}(2\pi)^{-\frac{d}{2}}\int_{\mathbb{R}^d} e^{(x+y)^\intercal w}e^{-\frac{\|w\|^2}{2}}dw \\
=& e^{-\|x\|^2-\|y\|^2}(2\pi)^{-\frac{d}{2}}\int_{\mathbb{R}^d} e^{\frac{\|x+y\|^2}{2}}e^{-\frac{\|w-(x+y)\|^2}{2}}dw \\
=& e^{-\|x\|^2-\|y\|^2}e^{\frac{\|x+y\|^2}{2}} \\
=& e^{-\frac{\|x-y\|^2}{2}} \\
=& K_G(x,y).
\end{aligned} \tag{20}$$

Therefore, $\varphi_{\text{PFF}}(x)^\intercal \varphi_{\text{PFF}}(z)$ is an unbiased estimate of Gaussian kernel $K_G(x, y)$. □

Theorem A.2. *The Weighted Positive Fixed Features (WPFF) is formulated as:*

$$\varphi_{WPFF}(x) := \frac{De^{-\|x\|^2}}{\sqrt{m}}[e^{\Phi^{-1}(t_1)x^\intercal v_1}, ..., e^{\Phi^{-1}(t_m)x^\intercal v_m}]^\intercal, \tag{21}$$

where D is a learnable parameter which can be optimized by the input x. Then the upper bound of the integral estimation error of the objective function by WPFF (Weighted Positive Fixed Features) method is not greater than the upper bound of the integral estimation error of the objective function by PFF (Positive Fixed Features) method.

Proof. The proof is motivated by (Yang et al., 2014). We use some of the same mathematical definitions and similar proofs from this paper to show that the WPFF method has a smaller upper bound on the overall estimation error of the objective function. Theorem A.2 Lemma A.3 and Lemma A.4 are all relevant to this paper.

Consider the task of computing an approximation of the following integral $I_{d,p}[f] = \int_{\mathbb{R}^d} f(x)p(x)dx$, because of $I_{d,p}[f] = E_{x\sim p(\mathbb{R}^d)}[f(x)]$, an empirical approximation called Monte Carlo (MC) to the integral can be computed by drawing a random point set $S = \{w_1, ..., w_s\}$ independently from $p(\mathbb{R}^d)$. When S is a set of fixed points, the empirical approximation is a quasi-Monte Carlo (QMC) method. The purpose of the QMC method is to improve convergence speed by constructing S using deterministic low-differential sequences instead of random sampling points. We have $I_S[f] = \frac{1}{s}\sum_{w\in S} f(w)$.

We define the integration error with respect to the point set S as $\epsilon_S[f] = |I_{d,p}(f) - I_S(f)|$. The integration error for PFF is as follows:

$$\epsilon_{S,p}[f] = |\int_{\mathbb{R}^d} f(x)p(x)dx - \frac{1}{s}\sum_{j=1}^{s} f(w_j)|, \tag{22}$$

where S is a set of fixed points.

The classical Monte Carlo and quasi-Monte Carlo approximations of integrals have consistent weights. However, it makes sense to weight the approximations, approximate $I_{d,p}[f] = \int_{\mathbb{R}^d} f(x)p(x)dx$ using $I_{S,\Xi}[f] = \sum_{j=1}^{s} \xi_j f(w_j)$, where $\Xi = \{\xi_1, \xi_2, ..., \xi_s\}$, $\xi_i \geq 0$ for $i \in \{1, 2, ..., s\}$, we do not need to normalize the weights, that is it is possible that $\sum_{i=1}^{s} \xi_i \neq 1$.

$$\begin{aligned}\int_{\mathbb{R}^d} e^{i(x-z)^T w} p(w)dw &\approx \sum_{j=1}^{s} \zeta_j(x)\zeta_j(z)f(w_j) \\ &= \sum_{j=1}^{s} \zeta_j(x)\zeta_j(z)e^{ix^T w_j}e^{-iz^T w_j} \\ &= \varphi_{\text{WPFF},S}(x)^T \varphi_{\text{WPFF},S}(z),\end{aligned} \tag{23}$$

where $\zeta_j(x)\zeta_j(z) = \xi_j$, $\zeta_j(x), \zeta_j(z) \geq 0$, $e^{ix^T w_j}e^{-iz^T w_j} = f(w_j)$ for $j \in \{1, 2, ..., s\}$, and $\Psi_S(x) = [\zeta_1(x)e^{ix^T w_1}, ..., \zeta_s(x)e^{ix^T w_s}]^T$.

The integration error for WPFF is as follows:

$$\epsilon_{S,p,\Xi}[f] = |\int_{\mathbb{R}^d} f(x)p(x)dx - \sum_{j=1}^{s} \xi_j f(w_j)|. \tag{24}$$

For a vector $b \in R_d$, let us define $\square b = \{u \in \mathbb{R}^d | |u_j| \leq |b_j|\}$. Let

$$F_{\square b} = \{f_u(x) = e^{ix^T u} | u \in \square b\}, \tag{25}$$

and consider the space of functions that admit an integral representation over $F_{\square b}$ of the form

$$f(x) = \int_{u\in\square b} \hat{f}(u)e^{ix^T u}du, \tag{26}$$

where $\hat{f}(u) \in \ell_2(\square b)$. This space is associated with the functions with compactly-supported inverse Fourier transforms called bandlimited functions, which play an important role in Shannon-Nyquist sampling theory. Under a natural choice of inner product, these spaces are called Paley-Wiener spaces and they constitute an RKHS.

Lemma A.3. *(The Kernel of Paley-Wiener RKHS) According to (Peloso, 2011), PW_b denotes the space of functions which are represented in the form of Eq.26, with the inner product $< f, g >_{PW_b} = (2\pi)^{2d} < \hat{f}, \hat{g} >_{L_2(\square b)}$. PW_b is an RKHS with kernel function,*

$$\sin c_b(u, v) = \pi^{-d} \prod_{i=1}^{d} \frac{sin(b_j(u_j - v_j))}{u_j - v_j} \tag{27}$$

Lemma A.4. *According to (Yang et al., 2014), for $f \in PW_b$ (Paley-Wiener spaces), we have*

$$\epsilon_{S,p}[f] \leq \|f\|_{PW_b} D_p^{\square b}(S) \tag{28}$$

where

$$D_p^{\square b}(S)^2 = \pi^{-d} \int_{\beta \in \square b} |\Psi(\beta)|^2 d\beta - \frac{2(2\pi)^{-d}}{s} \sum_{j=1}^{s} \xi_j \int_{\beta \in \square b} \Psi(\beta) e^{i\beta^T w_j} d\beta + \frac{1}{s^2} \sum_{l=1}^{s} \sum_{j=1}^{s} sinc_b(w_l, w_j). \tag{29}$$

Suppose that $p(\cdot)$ is a probability density function. Let $\Psi(\cdot)$ be the characteristic function associated with $p(\cdot)$.

Following (Yang et al., 2014), we can derive the following discrepancy measure that takes into account the weights:

$$\begin{aligned} D_p^{\square b}(S, \Xi)^2 =& \pi^{-d} \int_{\beta \in \square b} |\Psi(\beta)|^2 d\beta - 2(2\pi)^{-d} \sum_{j=1}^{s} \xi_j \int_{\beta \in \square b} \Psi(\beta) e^{i\beta^T w_j} d\beta + \sum_{l=1}^{s} \sum_{j=1}^{s} \xi_l \xi_j \sin c_b(w_l, w_j) \\ =& \pi^{-d} \int_{\beta \in \square b} |\Psi(\beta)|^2 d\beta - 2(2\pi)^{-d} \sum_{j=1}^{s} \zeta_j(x)\zeta_j(z) \int_{\beta \in \square b} \Psi(\beta) e^{i\beta^T w_j} d\beta + \sum_{l=1}^{s} \sum_{j=1}^{s} \zeta_l(x)\zeta_l(z)\zeta_j(z)\zeta_j(z) \sin c_b(w_l, w_j). \end{aligned} \tag{30}$$

We note that if we fix the points, then optimizing just the weights is a optimization problem. The discrepancy measure can be written as

$$\begin{aligned} D_p^{\square b}(S, \xi)^2 =& \pi^{-d} \int_{\beta \in \square b} |\Psi(\beta)|^2 d\beta - 2v^T \xi + \xi^T H \xi \\ =& \pi^{-d} \int_{\beta \in \square b} |\Psi(\beta)|^2 d\beta - 2v^T (\zeta(x) \otimes \zeta(z)) + (\zeta(x) \otimes \zeta(z))^T H (\zeta(x) \otimes \zeta(z)), \end{aligned} \tag{31}$$

where $\xi = (\xi_1, ..., \xi_s)^T$, $\zeta(x) = (\zeta_1(x), ..., \zeta_s(x))^T$, and $\xi = \zeta(x) \otimes \zeta(z) = (\zeta_1(x)\zeta_1(z), ..., \zeta_s(x)\zeta_s(z))^T$. $\otimes$ is the Hadamard product. $v \in R^s$, $v_j = (2\pi)^{-d} \int_{\beta \in \square b} \Psi(\beta) e^{i\beta^T w_j} d\beta$ and $H \in R^{s \times s}$, $H_{lj} = sinc_b(w_l, w_j)$.

Thus, the optimal weights can be found by solving the following convex optimization problem

$$\begin{aligned} &\arg \min_{\zeta(x), \zeta(z) \in R^s} -2v^T (\zeta(x) \otimes \zeta(z)) + (\zeta(x) \otimes \zeta(z))^T H (\zeta(x) \otimes \zeta(z)) \\ &\text{s.t.} \zeta(x) \leq 0, \zeta(z) \leq 0. \end{aligned} \tag{32}$$

So obviously for a fixed sampling point, there is $D_p^{\square b}(S, \xi) \leq D_p^{\square b}(S)$. Then the upper bound of the integral estimation error of the objective function by WPFF (Weighted Positive Fixed Features) method is not greater than the upper bound of the integral estimation error of the objective function by PFF (Positive Fixed Features) method.

□

A.1. Examples generated by DiJiang-7B

Prompt: There is a time when I had to make a difficult decision.

DiJiang-7B: I was in a situation where I could either stay or leave. It was a very difficult time for me. I was very happy in my job. But I felt that I needed to move on. So I decided to leave my current job and start my own business. This was not an easy decision to take. There were many things that needed my attention. And I didn't have enough time to do everything.

LLaMA2-7B: I had to choose between my family and the people who were close with me. It was a hard decision, but it turned out that they weren't really friends after all! I am not sure if you have ever been in this situation before or what your thoughts are on making such an important choice like choosing one over another person when there is no clear answer as far back then goes. . . but here we go anyway: "The best way forward would be for both parties involved (you included)to come together again soon so everyone can move past whatever happened.

Prompt: The sun goes down, and finally Gauss and Curie find time to relax and discuss after an exhausting day of work.

DiJiang-7B: "So, what do you think about the newest research on the effects of the magnetic field on human brain?" Gau asks, taking a sip of his coffee. He's been working on a new project for the last few weeks, trying to find a way to use the Earth's magnetic fields to help people with disabilities. It'd be a huge breakthrough for science, but it'll take a lot of time and money to get it to work properly. Gaug is a bit of a dreamer, so he'd rather not think too much about it. Curry, on other hand, is more pragmatic. She'll be the first to tell you that the world doesn't need more scientists, it needs more people who can actually do something. But she'l always been a realist, too. So, when Gaus asks her opinion, she doesn t hesitate to give it, even though she knows it might not be what he wants to hear. "Well, I think it s a great idea, Gause. I mean, if we can find some way of using the earth's magnetic energy to improve people's lives, that would be amazing. And I know you're working hard on it." Currey says, sounding a little bit tired. They both know that it ll take some time, though. The project is still in its early stages, after all. Still, Curreys smile is enough to make Gauc smile back. After all, they're both scientist. That means they can talk about science all day long, right?

LLaMA2-7B: Gauss: So, how did you like the new job? I heard that it was a bit of an adjustment. You're not used to working with people who are so. . . well-rounded and diverse in their interests! It must be quite different from your old life as one of Newton's apprentices at Cambridge University where all they cared about were numbers... or maybe just pi... but anyway - what do YOU think is more important for success these days – being able to calculate complex mathematical formulas quickly on paper using only pencil & eraser OR having good social skills when dealing w/ other humans?? Hmmm.... let me know if u find out!! Bye now!!!