EFFICIENT GENERATIVE AI BY TUCKER NETS

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose a trainable token mixing transformation based on Tucker maps — which are of (quasi)linear complexity in the input. On experiments on ViT and GPTs we show that replacing attention layers by our Tucker maps caused $\sim 50\%$ reduction in the number of FLOPs used in training — while almost never causing any deterioration of performance. We verify that the advantages of Tucker Nets on transformers, hold for both Adam(W) and Muon being used for training. Our Tucker layers are designed as drop-in replacements of the attention layers in any transformer architecture and thus pave a new way to efficient generative AI.

1 INTRODUCTION

Modern landscape of ML is shaped and driven in large parts by the power of the transformer architecture that enables parallel training and direct long-range interaction (Vaswani et al., 2017). Attention now supports bidirectional language representations (Devlin et al., 2019), in-context language modeling (Brown et al., 2020), image recognition (Dosovitskiy et al., 2021b), object detection (Carion et al., 2020), set processing (Lee et al., 2019), speech recognition (Gulati et al., 2020), graph learning (Ying et al., 2021), protein-structure prediction (Jumper et al., 2021), multimodal learning (Radford et al., 2021), and diffusion image generation (Peebles & Xie, 2023).

For N tokens of width C, dot-product attention forms $\mathbf{QK}^\top \in \mathbb{R}^{N\times N}$, costing $\Theta(N^2C)$ arithmetic and $\Theta(N^2)$ materialized memory. FlashAttention reduces memory traffic but not this quadratic dense arithmetic (Dao et al., 2022). Under SETH, Keles et al. (2023) establish quadratic lower bounds for several attention computations. Alman & Song (2023) show a bounded-entry phase transition between efficiently approximable and conditionally hard regimes. These are conditional, regime-dependent results.

Intense research has happened in the last couple of years to find attention-like mechanisms which circumvent the above quadratic complexity barrier of standard attention. Much of these attempts can be seen to be rooted in the pioneering kernelized view of attention in (Katharopoulos et al., 2020). Among these DeltaNet has emerged as a key template that treats the "attention matrix" $\mathbf{S}_t$, the $N \times N$ matrix from above, as fast weights and applies

$$\mathbf{S}_t = \mathbf{S}_{t-1} + \beta_t(\mathbf{v}_t - \mathbf{S}_{t-1}\mathbf{k}_t)\mathbf{k}_t^\top, \qquad \mathbf{o}_t = \mathbf{S}_t\mathbf{q}_t. \tag{1}$$

This corrects the value associated with $\mathbf{k}_t$ before writing (Yang et al., 2024b). Gated DeltaNet adds adaptive decay (Yang et al., 2024a); Kimi Delta Attention adds fine-grained gating and diagonal-plus-low-rank transitions in a hybrid model (Kimi Team, 2025); and Gated DeltaNet-2 decouples erase and write gates (Hatamizadeh et al., 2026). These works take delta-rule attention from an algorithmic proposal to released designs and trained billion-scale hybrid models. It is important to note that all such linear attention approaches are never deployed on their own but the successful architectures are a hybrid of them and the conventional scaled dot-product attention. *In contrast, our proposal is demonstrated to be a successful low complexity/(quasi-)linear drop-in replacement happening at every conventional attention layer of various successful transformers.*

Urgent Need to Reduce Resource Requirement of Generative-AI The rapidly increasing computational demand of large transformer models has become a energy and climate concern. It is estimated that global data-centre electricity consumption will rise from approximately 415 TWh in 2024 to 945 TWh in 2030, with AI-oriented accelerated servers accounting for nearly half of the projected increase (International Energy Agency, 2025).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

We posit that an important first step to meet this pressing challenge of our times is to be able to reduce the scaling of the computation with input size, particularly for transformers which form the backbone of most large models. Hence there is a pressing need for models whose forward passes require less computation than usual.

We recall that a common way to reduce the cost of transformer training and inference is to lower numerical precision, through low-precision training (e.g. FP16, BF16, or FP8 arithmetic) and post-training quantization of weights and activations i.e storing weights and activations in fewer bits and thereby lowering the cost of each arithmetic operation. Quantization, however, reduces cost only by a constant factor: a quantized self-attention layer still requires $\Theta(N^2C)$ operations for N tokens of width C. *Our proposal instead changes how the cost grows and hence a better scaling becomes increasingly more advantageous as data dimension grows.* The gains from our proposed transformer substitution are complementary to that of quantization.

We draw our primary inspiration from Tucker's multi-mode factor analysis (Tucker, 1966). We use that fundamental idea to define the new token mixing layer that we propose in Section 2. We define "Tucker Nets" as the general paradigm of building compositional architectures from interspersed layers of activations, affine/convolutional transforms, residual connections and "Tucker Maps" – as defined in Section 2. To emphasize, Tucker Nets do not use attention layers. In Sectiion 3 we give evidence on ViTs and GPTs of Tucker Nets working as an efficient replacement transformers.

2 Defining Tucker Maps

We will define three different options for the Tucker layer — the "free" version here, and the (causal) spectral versions in Sections 2.1.

Let $\mathcal{X} \in \mathbb{R}^{H\times W\times C}$ be the $C-$channel data arraanged as a matrix of dimensions $H \times W$, with entries x_{hwc}. For three matrices $\mathbf{G}_H \in \mathbb{R}^{H\times H}$, $\mathbf{G}_W \in \mathbb{R}^{W\times H}$ and $\mathbf{G}_C \in \mathbb{R}^{C\times C}$, we define three recall the definition of "mode products" as follows,

$$(\mathcal{X} \times_1 \mathbf{G}_H)_{h'wc} := \sum_{h=0}^{H-1} (\mathbf{G}_H)_{h'h} x_{hwc}. \tag{2}$$

$$(\mathcal{X} \times_2 \mathbf{G}_W)_{hw'c} := \sum_{w=0}^{W-1} (\mathbf{G}_W)_{w'w} x_{hwc}, \tag{3}$$

$$(\mathcal{X} \times_3 \mathbf{G}_C)_{hwc'} := \sum_{c=0}^{C-1} (\mathbf{G}_C)_{c'c} x_{hwc}. \tag{4}$$

Products on distinct modes commute. Drawing inspiration from the mechanism of LoRA Hu et al. (2022); Dettmers et al. (2023); Chen et al. (2024) for each "head" ℓ, we define a factorized representation of the matrices $\mathbf{G}_H$, $\mathbf{G}_W$ and $\mathbf{G}_C$ as follows,

$$\mathbf{G}_H^{(\ell)} = \mathbf{B}_H^{(\ell)} \mathbf{A}_H^{(\ell)}, \quad \mathbf{G}_W^{(\ell)} = \mathbf{B}_W^{(\ell)} \mathbf{A}_W^{(\ell)}, \quad \mathbf{G}_C^{(\ell)} = \mathbf{B}_C^{(\ell)} \mathbf{A}_C^{(\ell)}, \tag{5}$$

where $\mathbf{A}_H^{(\ell)} \in \mathbb{R}^{p\times H}$, $\mathbf{B}_H^{(\ell)} \in \mathbb{R}^{H\times p}$, and analogously for inner-dimension q for $\mathbf{A}_W^{(\ell)}$& $\mathbf{B}_W^{(\ell)}$ and for for inner-dimension r for $\mathbf{A}_C^{(\ell)}$& $\mathbf{B}_C^{(\ell)}$. Then the free Tucker map would be defined as,

$$\mathcal{T}_{\text{free}}(\mathcal{X}) = \sum_{\ell=1}^{L} \mathcal{X} \times_1 \mathbf{G}_H^{(\ell)} \times_2 \mathbf{G}_W^{(\ell)} \times_3 \mathbf{G}_C^{(\ell)}. \tag{6}$$

Note that (a) the above defines a $L-$headed trainable layer with 6 trainable matrices $\mathbf{A}_k^{(\ell)}$& $\mathbf{B}_k^{(\ell)}$ for $k = H, W$ & C. And (b) that this free Tucker layer preserves the rank and dimensions of the input (data) tensor. Evaluation of the above map costs $O(LHWC(p + q + r))$ i.e linear in the number of tokens $N = HW$ for fixed L, p, q, r, C.

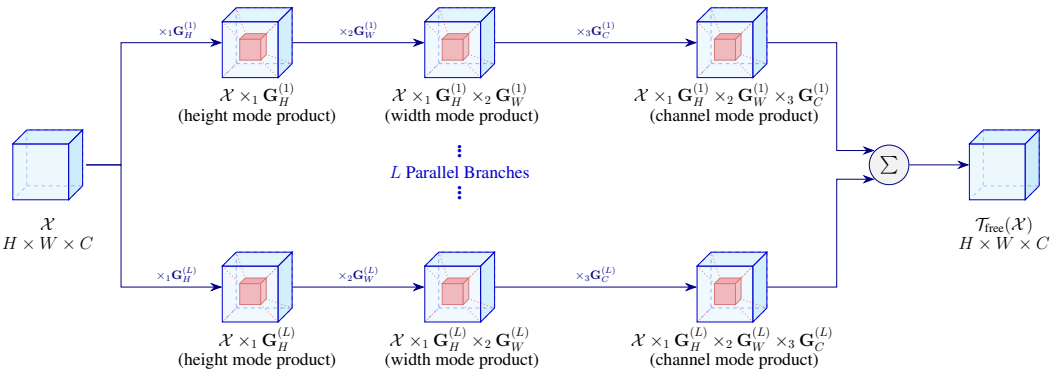


Figure 1: Visual representation of the free Tucker layer mapping $\mathcal{X} \mapsto \mathcal{T}_{\text{free}}(\mathcal{X})$. The red cores of the intermediate data states indicate that the mode products were with low rank constrained matrices.

2.1 Defining (Causal) Spectral Tucker Maps

For further acceleration of training and performance improvement we consider a particular class of low complexity parameterization of the $\mathbf{G}_H^{(\ell)}$ and $\mathbf{G}_W^{(\ell)}$ matrices from above. Our approach to these special choices of these two matrices is inspired by the structure of Fourier Neural Operator Li et al. (2021); Kovachki et al. (2021; 2023) in the context of data-driven PDE solving.

For defining "Spectral Tucker Maps" we construct the height and width factors by choosing either the Fourier bases that imposes periodic boundary conditions or the cosine bases inspired from the non-periodic DCT-II basis Strang (1999); Ahmed et al. (1974). Only one of these two following basis construction is used in a given spectral Tucker layer : which would be formally defined in Section 2.1.1 and its causal version in Section 2.2. In Section 2.3 we summarize how we envisage replacing the attention mechanism in any transformer architectures.

Spectral Bases Option 1 : Fourier for the Spatial Axes. For an $H \times W$ spatial grid, define the height-axis DFT matrix and the width-axis DFT matrix as, $\mathbf{F}_H, \in \mathbb{C}^{H \times H}$ and $\mathbf{F}_W \in \mathbb{C}^{W \times W}$ respectively by,

$$[\mathbf{F}_H]_{mh} := \frac{1}{\sqrt{H}} \exp\left(-\frac{2\pi \mathrm{i}\, mh}{H}\right), \; [\mathbf{F}_W]_{nw} := \frac{1}{\sqrt{W}} \exp\left(-\frac{2\pi \mathrm{i}\, nw}{W}\right). \tag{7}$$

Here h, w are spatial indices, while m, n are the corresponding Fourier mode indices. These matrices are unitary i.e $\mathbf{F}_W^* \mathbf{F}_W = \mathbf{I}_W$ & $\mathbf{F}_H^* \mathbf{F}_W = \mathbf{I}_H$.

For Tucker head ℓ, let $\boldsymbol{\alpha}^{(\ell)} = \left(\alpha_0^{(\ell)}, \ldots, \alpha_{k_h-1}^{(\ell)}\right) \in \mathbb{R}^{k_h}$ and $\boldsymbol{\beta}^{(\ell)} = \left(\beta_0^{(\ell)}, \ldots, \beta_{k_w-1}^{(\ell)}\right) \in \mathbb{R}^{k_w}$. The integers k_h and k_w are the numbers of retained *nonnegative* Fourier modes along height and width. We assume $1 \le k_h \le \left\lfloor \frac{H-1}{2} \right\rfloor + 1$, $1 \le k_w \le \left\lfloor \frac{W-1}{2} \right\rfloor + 1$. which avoids a separately treated Nyquist-frequency term – though experiments reveal no significant effect of having or not this "+1". To conveniently write the final matrices to be trained we embed the above trainable parameters into full even Fourier spectra $\widetilde{\boldsymbol{\alpha}}^{(\ell,H)} \in \mathbb{R}^H$ and $\widetilde{\boldsymbol{\beta}}^{(\ell,W)} \in \mathbb{R}^W$, defined by

$$\widetilde{\alpha}_m^{(\ell,H)} := \begin{cases} \alpha_0^{(\ell)}, & m = 0, \\ \alpha_m^{(\ell)}, & 1 \le m < k_h, \\ \alpha_{H-m}^{(\ell)}, & H - k_h < m < H, \\ 0, & \text{otherwise}, \end{cases} \quad \& \quad \widetilde{\beta}_n^{(\ell,W)} := \begin{cases} \beta_0^{(\ell)}, & n = 0, \\ \beta_n^{(\ell)}, & 1 \le n < k_w, \\ \beta_{W-n}^{(\ell)}, & W - k_w < n < W, \\ 0, & \text{otherwise}. \end{cases} \tag{8}$$

The DFT-based height and width factors are then, [1]

$$\mathbf{G}^{(\ell)}_{H,\mathrm{DFT}}(H) := \mathbf{F}^*_H \,\mathrm{diag}\left(\widetilde{\boldsymbol{\alpha}}^{(\ell,H)}\right)\mathbf{F}_H, \;\&\; \mathbf{G}^{(\ell)}_{W,\mathrm{DFT}}(W) := \mathbf{F}^*_W \,\mathrm{diag}\left(\widetilde{\boldsymbol{\beta}}^{(\ell,W)}\right)\mathbf{F}_W. \tag{10}$$

Spectral Bases Option 2 : DCT-II for the Spatial Axes. As an alternative to periodic Fourier modes, we may use the type-II discrete cosine transform. The DCT-II represents a finite signal using real cosine modes evaluated at half-shifted sample locations. It corresponds to applying a Fourier cosine expansion to an even extension of the signal and therefore does not identify the two ends of the spatial axis.

Define the height-axis DCT-II matrix $\mathbf{Q}_H \in \mathbb{R}^{H\times H}$ by

$$[\mathbf{Q}_H]_{mh} := \sqrt{\frac{2}{H}}\, s_m \cos\left(\frac{\pi m(h+\frac{1}{2})}{H}\right) \text{ for } 0 \le m,h < H \;\&\; s_m := \begin{cases} 2^{-1/2}, & m=0,\\ 1, & 1\le m<H. \end{cases} \tag{11}$$

Similarly, define the width-axis DCT-II matrix $\mathbf{Q}_W \in \mathbb{R}^{W\times W}$ by

$$[\mathbf{Q}_W]_{nw} := \sqrt{\frac{2}{W}}\, t_n \cos\left(\frac{\pi n(w+\frac{1}{2})}{W}\right) \text{ for } 0 \le n,w < W \;\&\; t_n := \begin{cases} 2^{-1/2}, & n=0,\\ 1, & 1\le n<W. \end{cases} \tag{12}$$

Both the above matrices are real and orthogonal. Further, for the DCT-II construction, let k_h and k_w denote the numbers of retained low-frequency cosine modes, $1 \le k_h \le H \;\&\; 1 \le k_w \le W$. Define the zero-padded spectral coefficient vectors $\widehat{\boldsymbol{\alpha}}^{(\ell,H)} \in \mathbb{R}^H$ and $\widehat{\boldsymbol{\beta}}^{(\ell,W)} \in \mathbb{R}^W$ by

$$\widehat{\alpha}^{(\ell,H)}_m := \begin{cases} \alpha^{(\ell)}_m, & 0\le m<k_h,\\ 0, & k_h\le m<H, \end{cases} \;\&\; \widehat{\beta}^{(\ell,W)}_n := \begin{cases} \beta^{(\ell)}_n, & 0\le n<k_w,\\ 0, & k_w\le n<W. \end{cases} \tag{11}$$

The DCT-II-based spatial factors are, [2]

$$\mathbf{G}^{(\ell)}_{H,\mathrm{DCT}} := \mathbf{Q}_H^\top \,\mathrm{diag}\left(\widehat{\boldsymbol{\alpha}}^{(\ell,H)}\right)\mathbf{Q}_H \;\&\; \mathbf{G}^{(\ell)}_{W,\mathrm{DCT}} := \mathbf{Q}_W^\top \,\mathrm{diag}\left(\widehat{\boldsymbol{\beta}}^{(\ell,W)}\right)\mathbf{Q}_W. \tag{13}$$

2.1.1 Defining Spectral Tucker Maps.

In light of the choices outlined in Equations 10 and 13, we denote a basis choice $b \in \{\mathrm{DFT}, \mathrm{DCT}\}$. and define,

$$\mathbf{G}^{(\ell)}_{H,b} := \begin{cases} \mathbf{G}^{(\ell)}_{H,\mathrm{DFT}}, & b=\mathrm{DFT},\\ \mathbf{G}^{(\ell)}_{H,\mathrm{DCT}}, & b=\mathrm{DCT}, \end{cases} \;\&\; \mathbf{G}^{(\ell)}_{W,b} := \begin{cases} \mathbf{G}^{(\ell)}_{W,\mathrm{DFT}}, & b=\mathrm{DFT},\\ \mathbf{G}^{(\ell)}_{W,\mathrm{DCT}}, & b=\mathrm{DCT}. \end{cases}$$

$\mathbf{G}^{(\ell)}_{H,b}$ is a dense $H\times H$ matrix generated at the current height H from the *resolution-independent* coefficients $\boldsymbol{\alpha}^{(\ell)}$. Likewise, $\mathbf{G}^{(\ell)}_{W,b}$ is generated at width W from $\boldsymbol{\beta}^{(\ell)}$. Further, we instantiate the

[1] Explicitly stated in reals,

$$\left[\mathbf{G}^{(\ell)}_{H,\mathrm{DFT}}\right]_{hh'} = \frac{1}{H}\left[\alpha^{(\ell)}_0 + 2\sum_{m=1}^{k_h-1}\alpha^{(\ell)}_m \cos\left(\frac{2\pi m(h-h')}{H}\right)\right],$$
$$\left[\mathbf{G}^{(\ell)}_{W,\mathrm{DFT}}\right]_{ww'} = \frac{1}{W}\left[\beta^{(\ell)}_0 + 2\sum_{n=1}^{k_w-1}\beta^{(\ell)}_n \cos\left(\frac{2\pi n(w-w')}{W}\right)\right]. \tag{9}$$

[2] Explicitly stated in reals,

$$\left[\mathbf{G}^{(\ell)}_{H,\mathrm{DCT}}(H)\right]_{hh'} = \frac{2}{H}\sum_{m=0}^{k_h-1} s_m^2\alpha^{(\ell)}_m \cos\left(\frac{\pi m(h+\frac{1}{2})}{H}\right)\cos\left(\frac{\pi m(h'+\frac{1}{2})}{H}\right),$$
$$\left[\mathbf{G}^{(\ell)}_{W,\mathrm{DCT}}(W)\right]_{ww'} = \frac{2}{W}\sum_{n=0}^{k_w-1} t_n^2\beta^{(\ell)}_n \cos\left(\frac{\pi n(w+\frac{1}{2})}{W}\right)\cos\left(\frac{\pi n(w'+\frac{1}{2})}{W}\right). \tag{13}$$

low-rank structure of the channel factor as, $\mathbf{G}_C^{(\ell)} = \mathbf{B}_C^{(\ell)}\mathbf{A}_C^{(\ell)}$ for $\mathbf{A}_C^{(\ell)} \in \mathbb{R}^{r\times C}$ & $\mathbf{B}_C^{(\ell)} \in \mathbb{R}^{C\times r}$. and $r \ll C$.

Combining the above, the spectral L-headed Tucker map associated with basis b is,

$$\mathcal{T}^{H,W}_{\mathrm{spec},b}(\mathcal{X}) := \sum_{\ell=1}^{L} \mathcal{X} \times_1 \mathbf{G}^{(\ell)}_{H,b} \times_2 \mathbf{G}^{(\ell)}_{W,b} \times_3 \mathbf{G}^{(\ell)}_{C}. \tag{14}$$

Resolution Transfer. The trainable parameters of spectral Tucker head ℓ are $\boldsymbol{\alpha}^{(\ell)} \in \mathbb{R}^{k_h}, \boldsymbol{\beta}^{(\ell)} \in \mathbb{R}^{k_w}$ and $\mathbf{A}_C^{(\ell)} \in \mathbb{R}^{r\times C}, \mathbf{B}_C^{(\ell)} \in \mathbb{R}^{C\times r}$. In particular, the dimensions of these parameters do not depend on the spatial resolution $H \times W$. Denote the basis choice as $b \in \{\mathrm{DFT}, \mathrm{DCT}\}$.

At the training resolution $H \times W$, the spatial factors are generated from the learned mode coefficients as described above. Suppose that the trained layer is evaluated at a different resolution $H' \times W'$. We call the new resolution admissible when

$$k_h \le \begin{cases} \lfloor (H'-1)/2 \rfloor + 1, & b = \mathrm{DFT}, \\ H', & b = \mathrm{DCT}, \end{cases} \;\&\; k_w \le \begin{cases} \lfloor (W'-1)/2 \rfloor + 1, & b = \mathrm{DFT}, \\ W', & b = \mathrm{DCT}. \end{cases}$$

These conditions ensure that all learned modes remain representable at the new resolution. For an admissible $H' \times W'$, embed the same learned coefficients into spectra of lengths H' and W'. In the DFT case, define $\widetilde{\boldsymbol{\alpha}}^{(\ell,H')} \in \mathbb{R}^{H'}, \widetilde{\boldsymbol{\beta}}^{(\ell,W')} \in \mathbb{R}^{W'}$ using the same real-even construction as earlier. In the DCT case, define $\widehat{\boldsymbol{\alpha}}^{(\ell,H')} \in \mathbb{R}^{H'}, \widehat{\boldsymbol{\beta}}^{(\ell,W')} \in \mathbb{R}^{W'}$ by placing the learned coefficients in the first k_h and k_w entries and setting all remaining entries to zero.

The transferred spatial factors are therefore

$$\mathbf{G}^{(\ell)}_{H',b}(H') := \begin{cases} \mathbf{F}^*_{H'} \operatorname{diag}\left(\widetilde{\boldsymbol{\alpha}}^{(\ell,H')}\right) \mathbf{F}_{H'}, & b = \mathrm{DFT}, \\ \mathbf{Q}^\top_{H'} \operatorname{diag}\left(\widehat{\boldsymbol{\alpha}}^{(\ell,H')}\right) \mathbf{Q}_{H'}, & b = \mathrm{DCT}, \end{cases} \tag{15}$$

$$\mathbf{G}^{(\ell)}_{W',b}(W') := \begin{cases} \mathbf{F}^*_{W'} \operatorname{diag}\left(\widetilde{\boldsymbol{\beta}}^{(\ell,W')}\right) \mathbf{F}_{W'}, & b = \mathrm{DFT}, \\ \mathbf{Q}^\top_{W'} \operatorname{diag}\left(\widehat{\boldsymbol{\beta}}^{(\ell,W')}\right) \mathbf{Q}_{W'}, & b = \mathrm{DCT}. \end{cases} \tag{16}$$

The corresponding spectral Tucker map at the new resolution is

$$\mathcal{T}^{H',W'}_{\mathrm{spec},b}(\mathcal{X}) := \sum_{\ell=1}^{L} \mathcal{X} \times_1 \mathbf{G}^{(\ell)}_{H',b}(H') \times_2 \mathbf{G}^{(\ell)}_{W',b}(W') \times_3 \mathbf{G}^{(\ell)}_{C}. \tag{17}$$

To state the resolution-independence property formally, define the trainable parameter tuple $\theta := \left\{\boldsymbol{\alpha}^{(\ell)}, \boldsymbol{\beta}^{(\ell)}, \mathbf{A}_C^{(\ell)}, \mathbf{B}_C^{(\ell)}\right\}_{\ell=1}^{L}$. For every admissible resolution $H \times W$, the spectral construction defines an operator

$$\mathcal{T}^{H,W}_{\mathrm{spec},b}(\,\cdot\,;\theta) : \mathbb{R}^{H\times W\times C} \longrightarrow \mathbb{R}^{H\times W\times C}.$$

The parameter space is therefore, $\Theta_{\mathrm{spec}} = \prod_{\ell=1}^{L} \left(\mathbb{R}^{k_h} \times \mathbb{R}^{k_w} \times \mathbb{R}^{r\times C} \times \mathbb{R}^{C\times r}\right)$, which is independent of H and W. Consequently, changing the resolution changes the domain and codomain of the induced operator, as well as the deterministic basis matrices, but it does not change the trainable parameter tuple θ, the shapes of its components, or their numerical values.

This is the precise sense in which the spectral Tucker mixer is resolution-transferable: the same parameter tuple θ induces a compatible operator at every admissible resolution.[3]

[3]By contrast, the free-factor Tucker map learns $\mathbf{A}_H^{(\ell)} \in \mathbb{R}^{p\times H}$ and $\mathbf{B}_H^{(\ell)} \in \mathbb{R}^{H\times p}$ and $\mathbf{A}_W^{(\ell)} \in \mathbb{R}^{q\times W}$ & $\mathbf{B}_W^{(\ell)} \in \mathbb{R}^{W\times q}$. At resolution $H' \times W'$, these parameters would have to be replaced by matrices in

Parameter Count for a Spectral Tucker Layer For the DFT based implementation the explicitly stored retained modes consist of k_h height frequencies and the k_w width frequencies. Hence the number of retained stored DFT coefficients per channel is $k_h + k_w$. For the DCT-II implementation, the retained coefficient block contains the first k_h height modes and first k_w width modes. Further there are $Cr + Cr = 2Cr$ channel-factor parameters. The complete layer therefore contains $P_{\text{spec}} = L(k_h + k_w + 2Cr)$ parameters. *This count is independent of H and W, which is precisely the parameter-level property that enables the resolution transfer described above.*

Spectral Tucker Layer is of Quasi-Linear Complexity in Input Tokens The spectral Tucker layer has three steps. **Step 1 (forward transform):** each of the C channels of the $H \times W$ token grid is mapped to the frequency domain with a separable two-dimensional transform, either a fast Fourier transform (DFT basis) or a fast DCT-II (DCT basis). This costs $O(CN \log N)$. **Step 2 (spectral mixing):** only the retained low-frequency modes are kept. There are $k_h k_w$ of them in the DCT-II basis and $(2k_h - 1)k_w$ in the DFT basis, which is at most $2k_h k_w$. At each retained mode (m, n), every head ℓ multiplies the C-dimensional spectral channel vector by $\mathbf{A}_C^{(\ell)} \in \mathbb{R}^{r \times C}$ to reduce it to rank r, multiplies the result by $\mathbf{B}_C^{(\ell)} \in \mathbb{R}^{C \times r}$ to restore width C, and scales it by $\alpha_m^{(\ell)} \beta_n^{(\ell)}$. The L head outputs are then summed. This step costs $O(Lk_h k_w Cr)$, which does not depend on N, because the number of retained modes is fixed by k_h and k_w rather than by H or W. **Step 3 (inverse transform):** the mixed spectrum, with all discarded modes set to zero, is mapped back to the token grid with the inverse FFT or the fast inverse DCT-II, again costing $O(CN \log N)$. For fixed L, k_h, k_w, r, and C, only Steps 1 and 3 grow with N, so the total cost is $O(N \log N)$, which is quasi-linear in the number of spatial tokens.

2.2 Defining Causal Spectral Tucker Maps.

For row-major $i = hW + w$, define $\text{vec}_{H,W}(\mathcal{X}) = \mathbf{X} \in \mathbb{R}^{HW \times C}$ by $X_{i,c} = x_{hwc}$. Its inverse satisfies $[\text{unvec}_{H,W}(\mathbf{X})]_{h,w,c} = X_{hW+w,c}$. With $M_{ij} = \mathbf{1}\{j \le i\}$, we define the causal form of our proposal as,

$$\mathcal{T}_{\text{causal}}(\mathcal{X}) = \text{unvec}_{H,W}\left(\sum_{\ell=1}^{L} [\mathbf{M} \odot (\mathbf{G}_H^{(\ell)} \otimes \mathbf{G}_W^{(\ell)})] \, \text{vec}_{H,W}(\mathcal{X})(\mathbf{G}_C^{(\ell)})^\top\right). \tag{11}$$

In above the output token i depends only on $j \le i$. [4]

2.3 The Attention Replacement Proposal.

In summary we propose the replacement of transformer blocks by Tucker blocks as follows,

Transformer Block	**Tucker Block**
$\mathcal{U}_A = \mathcal{X} + \text{Dropout}(\text{Attn}(\text{LN}_1(\mathcal{X})))$,	$\mathcal{U}_T = \mathcal{X} + \text{Dropout}(\mathcal{T}_{\{\text{free,spec}\}}(\text{LN}_1(\mathcal{X})))$,
$\mathcal{Y}_A = \mathcal{U}_A + \text{Dropout}(\mathcal{N}(\text{LN}_2(\mathcal{U}_A)))$.	$\mathcal{Y}_T = \mathcal{U}_T + \text{Dropout}(\mathcal{N}(\text{LN}_2(\mathcal{U}_T)))$.

In above both blocks get the tensor $\mathcal{X}$ as input and they produce tensor outputs $\mathcal{Y}_T$ and $\mathcal{Y}_A$ respectively, of the same rank and dimensions, LN are layer normalization and $\mathcal{N}$ is a generic neural net with activations, all of which are preserved when going from Transformers to Tucker blocks. Note that no activation is inserted in $\mathcal{T}$.

$\mathbb{R}^{p \times H'}, \mathbb{R}^{H' \times p}, \mathbb{R}^{q \times W'}, \mathbb{R}^{W' \times q}$. The parameter spaces at the two resolutions are therefore different. So the free-factor map does not have the direct resolution transfer property.

Also note that the resolution-transfer statement applies to the spectral Tucker mixer. Other resolution-dependent components of the host architecture, such as learned absolute positional embeddings, may still require interpolation.

[4] Masking breaks pure Fourier diagonalization, so the previous complexity estimates do not automatically apply but there is no experimentally observed slowing between the two spectral Tucker alternative.

3 EMPIRICAL EVIDENCE FOR TUCKER NETS

In our experiments only the mixer changes as outlined above — positional encoding, normalization, residuals, MLP activations, dropout, embeddings, objectives, optimization, augmentation, stopping, and splits are shared between the original transformer and its Tucker version. For architectures which originally had D layers of attention, validation jointly selects $\{(L_d, p_d, q_d, r_d)\}_{d=1}^{D}$ for the Tucker version of the experiment. Independent seeds provide means and standard deviations. A Tucker configuration is considered favorable only if it trains faster without material predictive-quality loss.[5]

3.1 VIT IMAGE CLASSIFICATION

We evaluate Spectral Tucker Attention as a replacement for standard self-attention in a Vision Transformer (ViT) (Dosovitskiy et al., 2021a) trained on CIFAR-10. The dataset contains $50,000$ training images and $10,000$ test images, each with spatial dimensions of 32×32 and three color channels. Training images are augmented using random cropping with four-pixel padding and random horizontal flipping. All images are converted to tensors and normalized using the CIFAR-10 channel-wise mean and standard deviation.

The baseline model is a six-layer ViT with an embedding dimension of 192, three attention heads, and a patch size of 4×4. Each image is therefore represented by a 8×8 grid of 64 patch tokens. No class token is used; instead, the token representations are aggregated using global average pooling before classification. Each transformer block uses the standard ViT architecture with pre-normalization, multi-head self-attention, residual connections, and a two-layer feed-forward network with a hidden dimension four times larger than the embedding dimension. The classification head produces logits for the ten CIFAR-10 classes.

The Tucker variant has exactly the same architecture and hyperparameters as the baseline, except that each self-attention module is replaced with Spectral Tucker Attention. The patch tokens are interpreted as an 8×8 spatial grid, and the spectral operator is configured with three components, four retained modes along each spatial dimension, and a channel Tucker rank of 32. The operator uses a discrete Fourier basis and performs a low-rank spectral transformation over the patch grid and channel dimension. Since the class token is omitted, all 64 image patches participate directly in the spatial transformation. The Tucker model is initialized with the same non-attention parameters as the baseline, while its spectral attention outputs are initialized to zero. Consequently, each Tucker block initially behaves as an identity mapping through its residual connection, providing a controlled starting point for optimization.

Both models are trained from scratch for 20 epochs using AdamW with a batch size of 128, an initial learning rate of 3×10^{-4}, weight decay of 0.05, and cosine learning-rate decay. Gradients are clipped to a maximum norm of 1.0. The models are initialized with the same random seed, and all components other than the attention mechanism are kept identical. This paired design isolates the effect of replacing conventional self-attention with the spectral Tucker operator.

After each epoch, we evaluate both models on the CIFAR-10 test set and record the average cross-entropy loss and classification accuracy. We also report the number of trainable parameters, the best and final test accuracy, and the average time per training epoch. To compare computational efficiency, the script estimates the training cost of each model by counting the FLOPs of a forward pass and multiplying the result by three to approximate the combined forward and backward cost. Training and test loss are plotted as functions of optimization steps, wall-clock time, and cumulative training FLOPs, while accuracy is plotted against the training iteration and epoch-level evaluation checkpoints.

Figure 2 compares the Standard ViT and Spectral Tucker ViT as a function of the number of CIFAR-10 training samples. For each training-set size, both models are trained from scratch using identical architectural and optimization settings. In Figure 3, the horizontal axis uses a logarithmic scale to show performance across training-set sizes ranging from 1,000 to 50,000 images. This comparison evaluates both data efficiency and computational cost: the left panel measures achievable predictive

[5] Let $\mathcal{S}_d$ be the pre-declared tuple set at layer d and $\tau < 1$ a required runtime ratio. Search $\mathcal{S}_1 \times \cdots \times \mathcal{S}_D$ on validation data, discard a combination c when $\bar{t}_c \geq \tau \bar{t}_{\text{attn}}$, maximize validation quality with runtime and parameters as fixed tie-breakers, then we lock that before test evaluation.

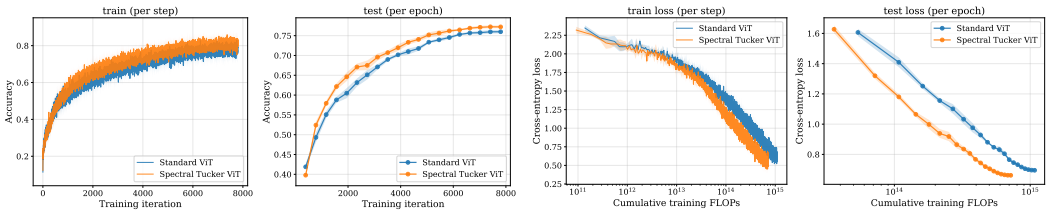


Figure 2: ViT Image Classification

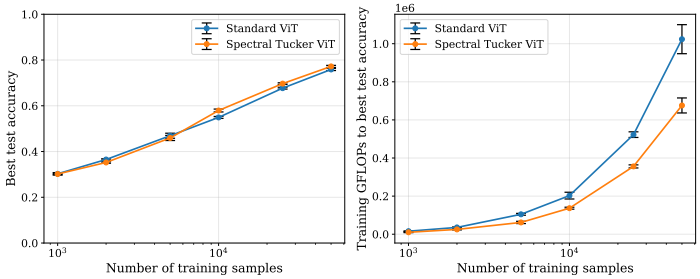


Figure 3: Scaling ViT across different training-set sizes

performance as more training data become available, while the right panel indicates the estimated computation required to reach each model's best observed test accuracy.

3.2 GPT ON TEXT GENERATION

We evaluate the Causal Spectral Tucker Map as a replacement for causal self-attention in a decoder-only GPT model using the Tiny Shakespeare character-level language-modeling benchmark. The corpus is tokenized at the character level and divided sequentially into training and validation partitions containing 90% and 10% of the data, respectively. During training, each example consists of a randomly sampled contiguous sequence of 128 characters, and the learning target is the same sequence shifted by one character.

The baseline is a four-layer GPT model with 192-dimensional token representations and four attention heads. Each layer contains pre-normalized causal multi-head self-attention, a residual connection, and a two-layer GELU feed-forward network with an expansion ratio of four. Learned positional embeddings are added to the token embeddings, and a final layer-normalization and linear projection produce next-character logits. The Tucker variant has the same embedding, positional encoding, normalization, feed-forward, residual, and output-projection components. Its attention modules are replaced with Causal Spectral Tucker Map operators configured with four spectral components, 16 retained sequence-direction frequency modes, one width-direction mode, and a channel Tucker rank of 32.

Both models are trained using AdamW for $3,000$ optimization steps with a batch size of 64, an initial learning rate of 3×10^{-4}, weight decay of 0.05, cosine learning-rate decay, and gradient-norm

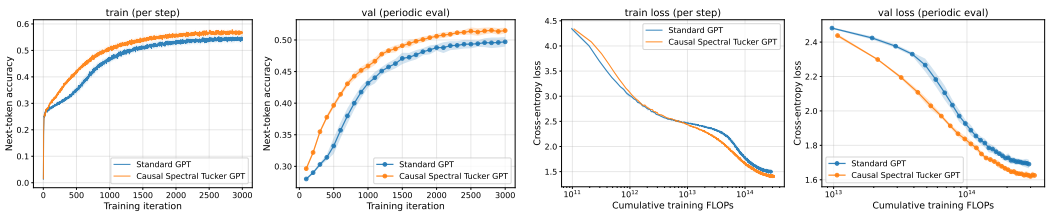


Figure 4: GPT on Text Generation

#chars	Standard GPT peak-accuracy target			Tucker GPT peak-accuracy target		
	Peak accuracy	Standard TFLOPs	Tucker TFLOPs	Peak accuracy	Standard TFLOPs	Tucker TFLOPs
10K	$27.10^{\pm 0.13}$	$1.961^{\pm 0.103}$	$0.681^{\pm 0.000}$	$31.99^{\pm 0.30}$	NR	$2.612^{\pm 0.300}$
50K	$34.29^{\pm 0.36}$	$8.464^{\pm 0.273}$	$2.839^{\pm 0.114}$	$38.11^{\pm 0.16}$	NR	$7.836^{\pm 0.197}$
200K	$35.11^{\pm 0.11}$	$8.154^{\pm 0.372}$	$3.293^{\pm 0.300}$	$38.94^{\pm 0.17}$	NR	$8.745^{\pm 0.632}$
803K	$35.82^{\pm 0.39}$	$8.258^{\pm 0.273}$	$3.293^{\pm 0.454}$	$39.71^{\pm 0.22}$	NR	$8.404^{\pm 0.495}$

Table 1: GPT scaling: Values are means with standard errors shown as superscripts. FLOPs count training compute and exclude evaluation. Accuracy is measured on the held-out validation split.

Table 2: Causal Spectral Tucker vs. causal self-attention in nanoGPT, character-level TinyShakespeare, Muon + Adam. Three seeds per model, all runs stopped by a uniform convergence criterion. Validation loss and all cost figures are taken at each run's minimum validation loss; accuracy is the best observed over the run. Entries are means over seeds, with $\pm$ s.d. where shown

		quality			cost to reach minimum val. loss			
Model	Params	Val loss	PPL	best val. acc	Iters	Time (s)	FLOPs	s/step
nanoGPT	10.77M	$\mathbf{1.4476}$ ± 0.0044	$\mathbf{4.254}$	$\mathbf{0.5863}$ ± 0.0004	2100	1238	2.32×10^{15}	0.590
causal spectral Tucker	**9.59M**	1.5142 ± 0.0052	4.546	0.5694 ± 0.0004	**700**	**318**	$\mathbf{7.07\times 10^{14}}$	**0.454**

clipping at 1.0. The two models share the same initialization for all non-attention parameters; the Tucker attention parameters are initialized when the attention modules are replaced. We report cross-entropy loss, next-character prediction accuracy, and estimated training FLOPs in Figure 4. These measurements are used to compare optimization behavior and computational cost as functions of training steps, and cumulative FLOPs.

Table 1 shows the GPT scaling across 10,000 to 803,000 training characters. Tucker GPT achieved higher peak validation accuracy than Standard GPT. The standard errors were small, ranging from 0.12 to 0.39 percentage points. Tucker GPT also reached Standard GPT's peak accuracy with 50% fewer FLOPs. Standard GPT did not reach Tucker GPT's peak accuracy in any of the three runs within 3,000 steps.

3.3 Modded nanoGPT Training using Muon

Replacing every causal self-attention layer of nanoGPT (Karpathy, 2022) with the causal spectral Tucker map trades a small amount of final quality for a large amount of compute. Under an identical Muon + Adam recipe (Jordan et al., 2024) and a uniform convergence-based stopping rule, the Tucker model uses 11% fewer parameters and reaches its early-stopping optimum with $3.3\times$ fewer training FLOPs and $3.9\times$ less wall-clock, but converges to a slightly higher validation loss and slightly lower next-token accuracy (Table 2). The advantageous trade-off of our proposal lies in the above slight decrease in performance being compensated on the compute axis: as each step is 23% cheaper in measured wall-clock and a 70% reduction in total training FLOPs to the optimum.

4 Conclusion

In this work we have proposed a new layer for token mixing that is provably of complexity (quasi)linear in input dimensions/number of tokens. Our proposal is designed to work as an immediate and easy to implement replacement of the attention layer and is grounded in classical ideas of the mathematics of Fourier analysis and factorization of tensors. In experiments with GPTs and ViTs we have shown evidence that our "Tucker Layer" when replaced in standard transformers can cause a substantial drop in the compute requirement, measured in FLOPs while causing a small change (often positive) in accuracy attained by the baseline. We posit that the evidence presented strongly suggests exploring this transformer alternative for more usecases — particularly those similar and/or scaled up versions of the three benchmark cases demonstrated here — and developing kernels dedicated to Tucker Nets. Such further studies can open up a new path to efficient generative AI.

AI USE STATEMENT

We used generative AI tools to assist with formulating mathematical claims, code generation for implementing parts of our method, identifying related literature. We did not use generative AI tools for generating synthetic datasets, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, cleaning or reformatting datasets, supporting qualitative or thematic data analysis, or interpreting results. The authors take full responsibility for the final content of this work, including all text and code produced with AI assistance.

REFERENCES

Nasir Ahmed, T. Natarajan, and K. R. Rao. Discrete cosine transform. *IEEE Transactions on Computers*, C-23(1):90–93, 1974.

Josh Alman and Zhao Song. Fast attention requires bounded entries. In *Advances in Neural Information Processing Systems*, 2023.

Tom B. Brown et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, 2020.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, 2020.

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. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2309.12307.

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

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2305.14314.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021a. URL https://openreview.net/forum?id=YicbFdNTTy.

Alexey Dosovitskiy et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021b.

Anmol Gulati et al. Conformer: Convolution-augmented transformer for speech recognition. In *Interspeech*, 2020.

Ali Hatamizadeh, Yejin Choi, and Jan Kautz. Gated DeltaNet-2: Decoupling erase and write in linear attention. *arXiv preprint arXiv:2605.22791*, 2026.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022. arXiv:2106.09685.

International Energy Agency. Energy and AI. Technical report, International Energy Agency, Paris, France, 2025. URL https://www.iea.org/reports/energy-and-ai.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks. https://kellerjordan.github.io/posts/muon/, 2024.

John Jumper et al. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596:583–589, 2021.

Andrej Karpathy. nanoGPT. https://github.com/karpathy/nanoGPT, 2022. GitHub repository.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, 2020.

Feyza Duman Keles, Pruthuvi Maheshakya Wijewardena, and Chinmay Hegde. On the computational complexity of self-attention. In *Proceedings of the 34th International Conference on Algorithmic Learning Theory*, volume 201 of *Proceedings of Machine Learning Research*, pp. 597–619, 2023.

Kimi Team. Kimi linear: An expressive, efficient attention architecture. *arXiv preprint arXiv:2510.26692*, 2025.

Nikola Kovachki, Samuel Lanthaler, and Siddhartha Mishra. On universal approximation and error bounds for Fourier neural operators. *Journal of Machine Learning Research*, 22(290):1–76, 2021.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to PDEs. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International Conference on Machine Learning*, 2019.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations (ICLR)*, 2021. arXiv:2010.08895.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *IEEE/CVF International Conference on Computer Vision*, 2023.

Alec Radford et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021.

Gilbert Strang. The discrete cosine transform. *SIAM Review*, 41(1):135–147, 1999.

Ledyard R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3): 279–311, 1966. doi: 10.1007/BF02289464.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving Mamba2 with delta rule. *arXiv preprint arXiv:2412.06464*, 2024a.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. In *Advances in Neural Information Processing Systems*, 2024b.

Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform bad for graph representation? In *Advances in Neural Information Processing Systems*, 2021.