PADRe: A Unifying Polynomial Attention Drop-in Replacement for Efficient Vision Transformer

Pierre-David Letourneau†, Manish Kumar Singh, Hsin-Pai Cheng, Shizhong Han, Yunxiao Shi, Dalton Jones, M. Harper Langston, Hong Cai, Fatih Porikli
Qualcomm AI Research*

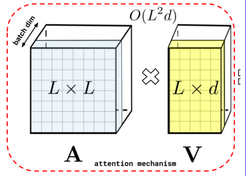
†pletourn@qti.qualcomm.com

Qualcomm AI research

ICLR

Motivation

- **Transformers** have become the backbone of modern models in **vision, language, and beyond**. But **self-attention mechanism** suffers from:
  - Quadratic compute and memory complexity
  - Poor scalability for on-device or embedded applications
- These limitations hinder deployment in real-time, mobile, and resource-constrained environments
- While many efficient alternatives exist, they often compromise generality or require redesigning the architecture

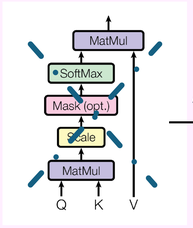


$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Can we build a lightweight, hardware-friendly, *drop-in replacement* for self-attention that improves performance while preserving accuracy?
Yes.

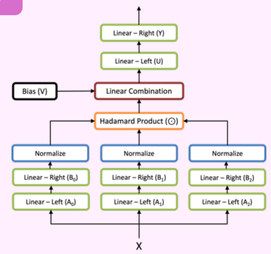
We replace attention with **PADRe**, keeping meta-architecture unchanged. PADRe uses
- Linear compute vs. quadratic compute of self-attention
- Linear memory vs. quadratic memory of self-attention
- Hardware-friendly operations vs. complex ops (e.g., softmax) in self-attention

Contributions



Attention is **NOT** all you need

PADRe is what you want



Each element of the output tensor is a polynomial function of the elements of the input tensor, the coefficients of which are polynomial functions of the parameters (weights). Further, this dependency is such that the coefficients and the output itself may be computed efficiently (e.g., in linear time).

- Propose PADRe, a principled, polynomial-based drop-in replacement for self-attention with linear compute and memory cost
- Unify multiple efficient attention variants (e.g., Hyena, Mamba, SimA, Conv2Former) under a single framework
- Achieve up to 43× speedup over standard self-attention on both GPUs and mobile NPUs, with comparable accuracy
- Enable efficient deployment of transformer models across vision and language tasks, especially in embedded and on-device settings

Method

A General Attention Drop-in Replacement Framework

- **Input** format identical to original network, e.g., tensor X of $B \times N \times D$
- Apply **linear transformations** (e.g., convs, matmuls) to both left and right of input tensor, given fast mat-vec products on hardware
- Use **hardware-friendly nonlinearities** (e.g., Hadamard products)
- Optionally, apply linear transformations if different out size is needed

Output is of same size as input. Replacement occurs in a black-box fashion; the rest of the network is unaware of the replacement

$$Y_i = A_i\, X\, B_i$$
$$Z_1 = Y_1$$
$$Z_{i+1} = (C_i Z_i D_i) \odot Y_{i+1}, \forall\, i \in \{1, \ldots, d-1\}$$
$$[P]_{m,n} = \sum_{i=1}^{d} [W]_{m,n,i}[Z_i]_{m,n} + [L]_{m,n,}$$
$$[P]_{m,n} = \sum_{i=1}^{d} [W]_{m,n,i}[Z_i]_{m,n} + [L]_{m,n} \sim \sum_{k \in \mathbb{N}^{N\times D}:\, |k| \le d} \pi_k x^k$$

Each element of the output tensor is a polynomial function of the elements of the input tensor

- Many recently proposed alternative attention mechanisms are in fact *special cases* of the PADRe framework:

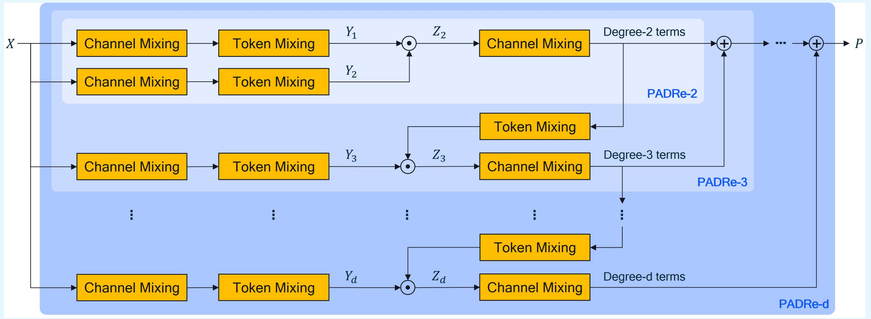
SimA = $\sum_{i=1}^{D}\sum_{j=1}^{N} \frac{1}{\|[Q^i]\|_1 \|[K^j]\|_1} (\sum_{k_Q=1}^{D}\sum_{k_K=1}^{D}\sum_{k_V=1}^{D} ([W_Q]_{k_Q,i}\,[W_K]_{k_K,i}\,[W_V]_{k_V,n})(X_{m,k_Q} X_{j,k_K} X_{j,k_V}))$

Hyena = $\sum_{i=1}^{N}\sum_{t_{i-1}=0}^{L} (P^N_{t_N,m_N} (\prod_{i=1}^{N} h^i_{t_i - t_{i-1}} P^{i-1}_{t_{i-1},m_{i-1}}))$

Mamba = $\sum_{n=0}^{t} (\sum_{i,j=1}^{L} [W_C \bar{A}^{t-n} W_B]_{i,j}\, (x_i \cdot x_j \cdot x_n))$

After performing algebraic manipulations, it can be shown that many proposed attention replacement mechanism can be expressed as polynomials in the input, and belong to the PADRe framework

Implementation: A PADRe Instance



We only use hardware-friendly ops: convolutions for token mixing, MLPs for channel mixing, and Hadamard products

Results

- We leverage our PADRe implementation to replace self-attention in several vision applications, including **image classification**, **single-image object detection**, and **point cloud object detection**

Baselines	Attention Mechanisms	Top-1 Accuracy ↑	FLOPs (G)	#Params (M)
DeiT-Tiny [27]	Standard self-attention [7]	72.2	1.3	5
	Hydra attention [1]	68.3	1.1	6
	Efficient attention [24]	70.2	1.1	6
	Linear-angular attention [36]	70.8	1.1	6
	Enhanced linear attention [4]	72.9	1.1	6
	k-NN attention [32]	73.0	1.3	6
	Focused linear attention [9]	74.1	1.1	6
	Vision Mamba [38]	76.1	2.6	7
	PADRe-2 (ours)	74.5	1.2	5
	PADRe-3 (ours)	76.5	1.7	7
DeiT-Small [27]	Standard self-attention [7]	79.8	4.6	22
	SimA [13]	79.8	4.6	22
	k-NN attention [32]	80.1	4.6	22
	Vision Mamba [38]	80.5	9.6	26
	PADRe-2 (ours)	80.4	4.8	20
	PADRe-3 (ours)	80.5	6.1	26
DeiT-Base [27]	Standard self-attention [7]	81.8	17.6	86
	Hydra attention [1]	76.4	16.9	-
	Hyena [20]	78.5	-	87
	PADRe-2 (ours)	81.4	18.8	80

Image classification on ImageNet

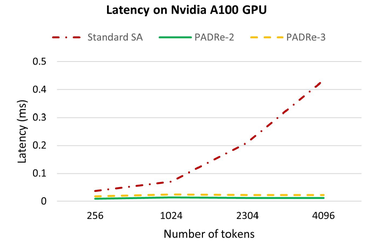
Model	Attention Mechanisms	AP ↑	FLOPs (G)	#Params (M)
DETR [5]	Standard self-attention [7]	40.6	10.1	6.1
	PADRe-2 (ours)	40.2	4.5	2.7

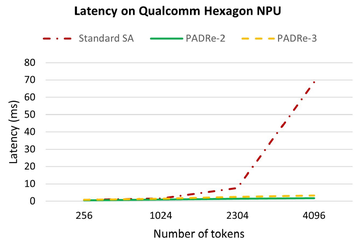
2D object detection on COCO

Baseline	Attention Mechanisms	NDS ↑	mAP ↑	FLOPs (G)	#Params (M)
DSVT [31]	Standard self-attention [7]	71.1	66.4	14.9	1.1
	PADRe-2 (ours)	71.1	66.0	10.7	0.9

3D object detection on nuScenes

- Latency evaluation on device





Up to 43× speedup and **linear memory scaling**

*Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc