AI in the Sciences and Engineering 2024: Lecture 17

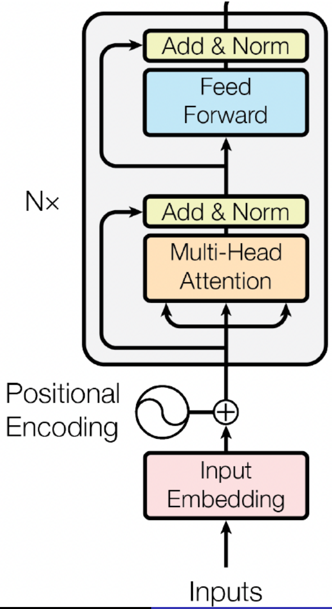
Siddhartha Mishra

Seminar for Applied Mathematics (SAM), D-MATH (and),
ETH AI Center,
ETH Zürich, Switzerland.

What you learnt so far

- Operator learning: Given Abstract PDE: $\mathcal{D}_a(u) = f$
- Learn Solution Operator: $\mathcal{G} : \mathcal{X} \mapsto \mathcal{Y}$ with $\mathcal{G}(a, f) = u$
- Approximate with Operator Learning Algorithms:
- We focus on using Transformers

Final version of a Transformer Block



Attention as a Neural Operator: I

- Let $\mathrm{v} \in C(D, \mathbb{R}^n)$ be the input function.
- $x_k \in D$ be sampling points on a Regular Grid with size Δ
- Apply Self-Attention to Tokens $\mathrm{v}_k = \mathrm{v}(x_k)$:

$$\mathrm{u}_k = \mathrm{W} \sum_{j=1}^{K} \text{softmax}_k \left(\frac{\langle \mathrm{Q}\mathrm{v}_k, \mathrm{K}\mathrm{v}_j \rangle}{\sqrt{m}} \right) \mathrm{V}\mathrm{v}_j, (\text{softmax}\ (w))_i = \frac{e^{w_i}}{\sum\limits_{\ell=1}^{L} e^{w_\ell}}$$

- Passing to the limit as $\Delta \to 0$ yields

$$\mathrm{u}(x) = \mathbb{A}(\mathrm{v})(x) = \mathrm{W} \int\limits_D \frac{e^{\frac{\langle \mathrm{Q}\mathrm{v}(x), \mathrm{K}\mathrm{v}(y) \rangle}{\sqrt{m}}}}{\int\limits_D e^{\frac{\langle \mathrm{Q}\mathrm{v}(z), \mathrm{K}\mathrm{v}(y) \rangle}{\sqrt{m}}} dz} \mathrm{V}\mathrm{v}(y) dy.$$

Attention as a Neural Operator: II

- Operator Attention $\mathbb{A} : C(D, \mathbb{R}^n) \mapsto C(D, \mathbb{R}^n)$ with

$$\mathrm{u}(x) = \mathbb{A}(\mathrm{v})(x) = \mathrm{W} \int_D \frac{e^{\frac{\langle \mathrm{Qv}(x), \mathrm{Kv}(y) \rangle}{\sqrt{m}}}}{\int_D e^{\frac{\langle \mathrm{Qv}(z), \mathrm{Kv}(y) \rangle}{\sqrt{m}}} dz} \mathrm{Vv}(y) dy.$$

- Can be interpreted as as a Nonlinear Kernel Neural Operator:

$$\mathbb{A}(\mathrm{v})(x) = \int_D K(v(x), v(y)) v(y) dy.$$

- In contrast, FNO/CNO etc are Linear Kernel Neural Operators:

$$\mathbb{C}(\mathrm{v})(x) = \int_D K(x, y) v(y) dy = \int_D K(x - y) v(y) dy.$$

Caveat: Computational Complexity

- Computational Cost is Quadratic in # (Tokens) !!

$$\text{Compute} \sim \mathcal{O}(mnK^2)$$

- With K- Input Length, n Input features and m hidden dimension.
- But lots of possible Parallelism in Computation
- $\mathcal{O}(1)$ sequential operations.
- $\mathcal{O}(1)$ Path Length.
- Nevertheless, Infeasible for 2 or 3-d inputs.

Possible Solution

- Vision Transformers (ViT) of Dosovitskiy et. al.
- Key Ingredient: Patching (Patchification):
- Given Image on resolution $H \times W$, i.e. $\mathrm{v} \in \mathbb{R}^{H \times H \times C}$
- Divide into $N = \frac{H^2}{p^2}$ patches, each of size $p \times p$
- $\mathrm{v} \sim [\mathrm{v}_1, \mathrm{v}_2, \dots, \mathrm{v}_N]$ with $\mathrm{v}_k \in \mathbb{R}^{p^2 \times C}$, $1 \leq k \leq N$
- Introduce Patch Embeddings $\mathrm{E} \in \mathbb{R}^{n \times (p^2 \cdot C)}$
- Input is a Sequence of Patch Tokens:

$$\hat{\mathrm{v}} = [\mathrm{E}\mathrm{v}_1, \mathrm{E}\mathrm{v}_2, \dots, \mathrm{E}\mathrm{v}_N]$$

- N Tokens, each with n features are fed into a transformer !!

Operator Version

- Patch Formation + Embeddings can be written as operator:

$$\hat{\mathrm{E}}(\mathrm{v})(x) = \sum_{k=1}^{N} \mathrm{F}\left(\int_{D_k} W(x)\mathrm{v}(x)dx\right)\mathbb{I}_{D_k}(x).$$

- With $D = \cup_{k=1}^{N} D_k$ non-overlapping and equal partition.
- With learnable $F \in \mathbb{R}^{n\times C}$
- Weight Function $W(x) = \sum\limits_{1\leq i,j\leq H} W_{ij}\delta_{z_{ij}}$
- With uniform grid points z_{ij} and discrete weights defined by,

$$\begin{aligned} W_{ij} &= \omega_{ij} \quad \text{if} \quad 1 \leq i,j \leq p \\ &= \omega_{i \bmod p, j \bmod p}, \quad \text{otherwise.} \end{aligned}$$

Issue: Where is Position ?

- Need to add Positional Encodings
- Input:
- Sequence of Patch Tokens + Learnable Positional Encodings

$$\hat{\mathrm{v}} = [\mathrm{Ev}_1, \mathrm{Ev}_2, \dots, \mathrm{Ev}_N] + \mathrm{E}_{pos}$$

- With $E_{pos} \in \mathbb{R}^{n \times N}$
- Can be viewed as an operator:

$$\mathrm{E}_{pos}\mathrm{v}(x) = \sum_{k=1}^{N} w_k \mathbb{I}_{D_k}(x), \quad w_k \in \mathbb{R}^n, \ \forall 1 \leq k \leq N.$$

ViT operator Block

- For $D \subset \mathbb{R}^2$ + input $v \in C(D, \mathbb{R}^C)$
- A sequence of Operators of the form:
- Patch Embeddings+ Positional Encoding: $\hat{v} = \hat{E}(v) + E_{pos}(v)$
- LayerNorm + MSA+ Residual: $\bar{u} = \hat{v} + MSA(LN(v))$
- LayerNorm + MLP+ Residual: $u = \bar{u} + MLP(LN(\bar{u}))$

Computational Complexity

- Given an Image at resolution $H \times W$
- Standard Transformer needs

$$\text{Compute} \sim \mathcal{O}((HW)^2)$$

- ViT needs

$$\text{Compute} \sim \mathcal{O}\left(\frac{(HW)^2}{p^4}\right)$$

- Still not scalable for small patch size p

Another Idea: Windowed Attention

- Introduced in Liu et. al.
- Use Windowed Attention:

- With M-Windows,

$$\text{Compute} \sim \mathcal{O}\left(\frac{HWM^2}{p^2}\right)$$

Operator version of Windowed Attention

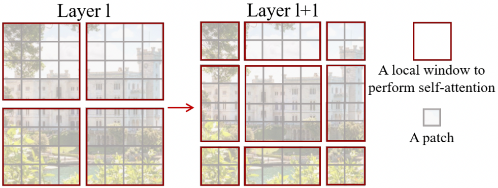
- For layer ℓ, Assume $D = \cup_{q=1}^{M} D_q^{w,\ell}$
- With non-overlapping Windows.
- Windowed Attention is instantiated as Operator:

$$\mathrm{u}(x) = \mathbb{A}_W(\mathrm{v})(x) = \mathrm{W} \int\limits_{D_{q_x}^{w,\ell}} \frac{e^{\frac{\langle \mathrm{Qv}(x), \mathrm{Kv}(y) \rangle}{\sqrt{m}}}}{\int\limits_{D_{q_x}^{w,\ell}} e^{\frac{\langle \mathrm{Qv}(z), \mathrm{Kv}(y) \rangle}{\sqrt{m}}} dz} \mathrm{Vv}(y) dy.$$

- Where $1 \leq q_x \leq M$ such that $x \in D_{q_x}^{w,\ell}$

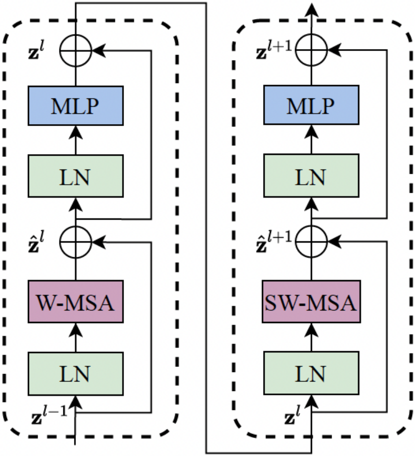
Shifting the Windows

- How to Tokens outside the Window ?
- Solution: Window shifts across Layers !!

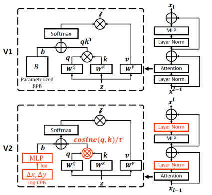


- Reduces Path Length across Tokens.

Swin Transformer Block



Modifications for Scalability



- Replace scaled dot product with Scaled Cosine
- Use MLPs on Relative Position Coordinates to generate positional encodings.

scOT: scalable Operator Transformer