REPORT ON THE PAPER "POSEIDON: EFFICIENT FOUNDATION MODELS FOR PDES"

DEVANSH TRIPATHI[1]
ETH Zürich

ABSTRACT. In the paper [3], the author introduces a foundation model POSEIDON, for learning the solution operators of PDEs. It is based on a multiscale operator transformer, with time-conditioned layer norms that enable continuous-in-time evaluations. They propose a novel training strategy leveraging the semi-group property of time-dependent PDEs to allow for significant scaling-up of the training data. POSEIDON is a pretrained model on a diverse, large scale dataset for the governing equations of fluid dynamics. The authors show that POSEIDON exhibits excellent performance by outperforming baselines significantly, both in terms of sample efficiency and accuracy. They also show the generalization ability of the model to unseen physics.

1. Introduction

Partial Differential Equations (PDEs) are referred to as the language of physics as they mathematically model a very wide variety of physical phenomena across a vast range of spatio-temporal scales. Numerical methods such as finite difference, finite element, spectral methods etc. are commonly used to approximate or simulate PDEs. However, their (prohibitive) computational cost, particularly for the so-called many-query problems, has prompted the design of various data-driven machine learning (ML) methods for simulating PDEs. Among them, operator learning algorithms have gained increasing traction in recent years.

These methods aim to learn the underlying PDE solution operator, which maps function spaces inputs (initial and boundary condition, coefficients, sources) to the PDE solution. They include algorithms which approximate a *discretization*, on a fixed grid, of the underlying solution operator. These can be based on convolutions [16], graph neural network [11, 14] or transformers [12, 2]. Other operator learning alogorithms are *neural operators* which can directly process function space input and outputs, possibly sampled on multiple grid resolutions. These include DeepONets [9], Fourier Neural Operator [5], CNO [13], among many others.

How can the number of training samples for PDE learning be significantly reduced? *Foundation models* are *generalist* models that are pretrained, at-scale, on laarge datasets drawn from a diverse set of data distributions. They leverage the intrinsic abilitiy of neural networks to learn *effective representations* from pretraining and are then successfully deployed on a variety of *downstream* tasks by *finetunning* them on a few task-specific samples. Example of such models include highly successful large language models [15].

The challenge of designing such foundation models for PDEs is formidable given the sheet variety of PDEs and data distributions. Authors concur that the feasibility of of designing PDE foundation models rests on the fundamental and unanswered science question of *why pretraining a model on a very small set of PDEs and underlying data-distributions can allow it to learn effective representations and generalize to unseen and unrelated PDEs and data-distributions via finetuning?*

The Poseidon family of PDE foundation models are based on i) scalable Operator Transformer or scOT, a *multiscalae vision transformer* with (shifted) windowed or Swin attention [6], adapted

[1]Seminar für Angewandte Mathematik, HG E 62.2, Rämistrasse 101, 8092 Zürich, Switzerland devansh.tripathi@sam.math.ethz.ch.

for operator learning, ii) a novel all2all training strategy for efficient leveraging *trajectories* of solutions of time-dependent PDEs to scale up the volume of training data and iii) an open source large-scale pretraining dataset, containing a set of novel solution operators of the compressible Euler and incompressible Navier-Stokes equations of fluid dynamics.

2. Approach

Problem Formulation. We denote a generic time-dependent PDE as,

$$\begin{aligned} &\partial_t u(x,t) + \mathcal{L}(u, \nabla_x u_x, \nabla_x^2 u, \dots) = 0, \quad \forall x \in D \subset \mathbb{R}^d, t \in (0,T), \\ &\mathcal{B}(u) = 0, \quad \forall (x,t) \in \partial D \times (0,T), \quad u(0,x) = a(x), \quad x \in D \end{aligned} \tag{2.1}$$

Here, with a function space $\mathcal{X} \subset L^p(D;\mathbb{R}^n)$ for some $1 \leq p < \infty$, $u \in C([0,T];\mathcal{X})$ is the solution of ((2.1)), $a \in \mathcal{X}$ the initial datum and $\mathcal{L}, \mathcal{B}$ are the underlying differential and boundary operators, respectively. Note that (2.1) accomodates both PDEs with high-order time derivatives as well as PDEs with (time-independent) coefficients and sources by including the underlying functions within the solution vector and augmenting $\mathcal{L}$ accordingly.

Even *time-independent* PDEs can be recovered from (2.1) by taking the *long-time limit*, i.e., $\lim_{t\to\infty} u = \overline{u}$, which will be the solution of the (generic) time-independent PDE,

$$\mathcal{L}(\overline{u}(x), \nabla_x \overline{u}, \nabla_x^2 \overline{u}, \dots) = 0, \quad \forall x \in D, \quad \mathcal{B}(\overline{u}) = 0, \quad \forall x \in \partial D. \tag{2.2}$$

Solutions of the PDE (2.1) are given in terms of the underlying *solution operator* $\mathcal{S} : [0,T] \times \mathcal{X} \mapsto \mathcal{X}$ such that $u(t) = \mathcal{S}(t,a)$ is the solution of 2.1 at any time $t \in (0,T)$. Given a data distribution $\mu \in Prob(\mathcal{X})$, the *underlying operator learning task (OLT)* is,

> **OLT**: *Given any initial datum $a \sim \mu$, find an approximation $\mathcal{S}^* \approx \mathcal{S}$ to the solution operator $\mathcal{S}$ 2.1, in order to generate the entire solution trajectory $\{\mathcal{S}^*(t,a)\}$ for all $t \in [0,T]$.*

It is essential to emphasize here that the learned opeator $\mathcal{S}^*$ has to generate the *entire solution trajectory for 2.1, given only the initial datum (and boundary conditions),* as this is what the underlying solution operator $\mathcal{S}$ (and numerical approximation to it) does.

Model Architecture. The backbone for the POSEIDON foundation model is provided by scOT or *scalable Operator Transformer,*. scOT is a *hierarchical multiscale vision transformer with lead-time conditioning* that processes lead time t and function space valued initial data input a to appropriate the solution operator $\mathcal{S}(t,a)$ of the PDE 2.1.

As in a vision transformer [1], any underlying input is first *partitioned into patches and (linearly) embedded into a latent space.* At the level of function inputs $a \in C(D;\mathbb{R}^n)$, this amounts to the action of the patch *patch partitioning and embedding* operator $v = \hat{\mathbf{E}}(a)$, with $\hat{\mathbf{E}}$ defined in A.1. This operator transforms the input function into a piecewise constant function, which is constant within patches (subdivisions of the domain D), by taking the weighted averages and then transforming these piecewise constant values into a C-dimensional latent space resulting in output $v \in C(D;\mathbb{R}^C)$.

As shown in Figure 1, this patch embedded output is then processed through a sequence of *SwinV2 transformer* blocks [7, 6], each of which has a structure of $SW_l : C(D;\mathbb{R}^C) \mapsto C(D;\mathbb{R}^C)$,

$$\begin{aligned} \mathbf{v_l} = SW_l(\mathbf{v}_{l-1}) &= \mathbf{v'}_l + LN_{\alpha_2^l,\beta_2^l}(MLP(\mathbf{v'}_l)), \\ \mathbf{v'}_l &= \mathbf{v}_{l-1} + LN_{\alpha_1^l,\beta_1^l}(W-MSA(\mathbf{v}_{l-1})). \end{aligned} \tag{2.3}$$

for layer index $l = 1,\dots,L$. The main building block of a SwinV2 transformer block 2.3 is the *windowed multi-head self attention* operator defined in A.3. The attention operator acts only inside window and the windows are shifted across layers so that all points in the domain can be attended to.

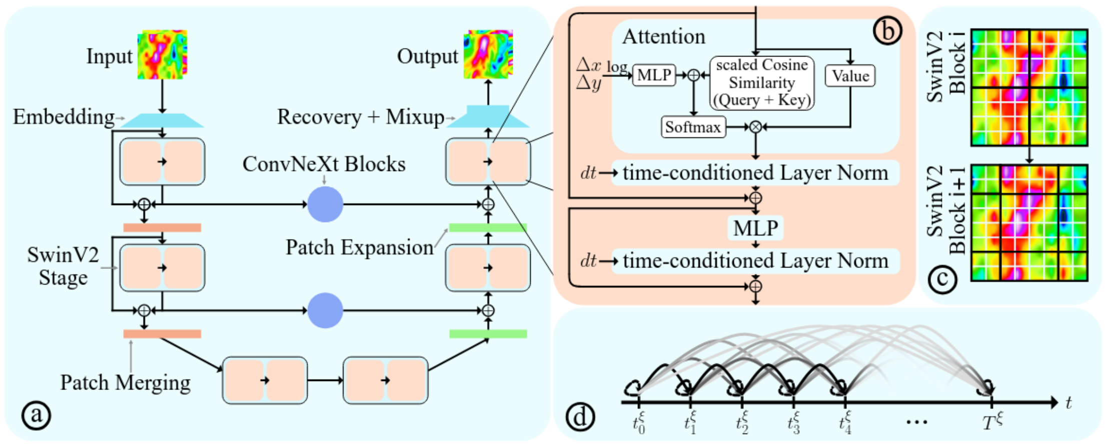
Input
Output
Embedding
Recovery + Mixup
ConvNeXt Blocks
SwinV2 Stage
Patch Expansion
Patch Merging
Attention
Δx Δy log
MLP
scaled Cosine Similarity (Query + Key)
Value
Softmax
dt time-conditioned Layer Norm
MLP
dt time-conditioned Layer Norm
SwinV2 Block i
SwinV2 Block i+1
t_0^ξ t_1^ξ t_2^ξ t_3^ξ t_4^ξ … T^ξ t
a
b
c
d

Figure 2: (a) scOT, the model underlying POSEIDON; (b) SwinV2 Transformer block; (c) Shifting Window over patch-based tokens with window (patch) boundaries with black (white); (d) all2all Training for time-dependent PDEs.

FIGURE 1. Taken from [3]

The MLP is 2.3, is defined in A.4. Author follows [10] to propose a *time-conditioning* strategy by introducing a *lead-time conditioned* layer norm in 2.3,

$$\begin{aligned} LN_{\alpha(t),\beta(t)}(\mathbf{v})(x) &= \alpha(t) \odot \frac{\mathbf{v}(x) - \mu_{\mathbf{v}}(x)}{\sigma_{\mathbf{v}}(x)} + \beta(t), \\ \mu_{\mathbf{v}}(x) = \frac{1}{C}\sum_{j=1}^{C} \mathbf{v}_j(x), &\quad \sigma^2_{\mathbf{v}}(x) = \frac{1}{C}\sum_{j=1}^{C}(\mathbf{v}_j(x) - \mu_{\mathbf{v}}(x))^2, \end{aligned} \tag{2.4}$$

Here $\alpha(t) = \alpha t + \overline{\alpha}$ and $\beta(t) = \beta t + \overline{\beta}$, with learnable $\alpha, \overline{\alpha}, \beta, \overline{\beta}$ although more general (small) MLPs can also be considered. This choice of time embedding enables *continuous-in-time evaluations.*

Finally, as in Figure 1, the SwinV2 transformer blocks 2.3 are arranged in a hierarchical, multiscale manner, within a U-Net style *encoder-decoder* architecture, by employing patch merging (downscaling) and patch expansion (upscaling) operator. Moreover, layers at the same scale, but within the encoder and decoder stages of scOT, respectively, are connected through *ConvNeXt* convolutional layers [8].

3. Universal Approximation of POSEIDON

This section deals with proving that the underlying architecture of POSEIDON is universal by showing that the building blocks of POSEIDON can proved to be equivalent to ANOs [4], and then we utilize the universality of ANOs to show that POSEIDON is indeed a universal approximator.

Lemma 1 (ConvNeXt is an ANO). *ConvNeXt blocks are given by*

$$\mathcal{Q}_i(\mathbf{v}, t) = (GeLU(LN(DwConv(\mathbf{v}), t)W_{\mathcal{Q},1} + b_{\mathcal{Q},1})W_{\mathcal{Q},2} + b_{\mathcal{Q},2}) \odot W_{\mathcal{Q},3} + \mathbf{v}. \tag{3.1}$$

where DwConv is a depthwise convolution with kernel size 7. With the following choice: taking all weight matrices to be identity matrix, $W_{\mathcal{Q},3}$ as vector of $1's$ and all bias vector to zero vectors then we have a reduced form as

$$\mathcal{Q}_i(\mathbf{v}, t) = GeLU(LN(DwConv(\mathbf{v}), t)) + \mathbf{v}. \tag{3.2}$$

This is an ANO and by Theorem 2.1 in [4], ConvNeXt is universal approximator.

Proof. The convolution given by DwConv can be represented as an integral and this will provide nonlocality to the network. Let $\mathcal{K}_l(v)$ be the convolution operator given by

$$(\mathcal{K}_l v)(x) = \int_D K_l(x,y)v(y)dy.$$

The particular choice $K_l(x,y) \equiv |D|^{-1} I_{7\times 7}$ with $I_{7\times 7}$ be the identity matrix, it recovers the form of average as in ANO [4].

Layer Norm. If we choose $\alpha(t)$ to be vector of 1's and $\beta(t)$ to be zero vector in LN (time conditioned layer norm as in 2.4) then LN only changes the first and second moments of the entries of the vector $\mathbf{v}(x)$ and the affine transformation becomes identity transform due the above choices. Overall output still captures the nonlocality needed for the ANO because of the integral and just changes the numerical values of the entries of the vector. Hence, it is still nonlocal and appyling LN does not affect the nonlocailty.

$$LN_{\alpha(t),\beta(t)}(\mathbf{v})(x) = \alpha(t) \odot \frac{\mathbf{v}(x) - \mu_{\mathbf{v}}(x)}{\sigma_{\mathbf{v}}(x)} + \beta(t) = \frac{\mathbf{v}(x) - \mu_{\mathbf{v}}(x)}{\sigma_{\mathbf{v}}(x)}$$

Residual connection. Applying residual connection mathematically means shifting the output vector of GeLU. It is required to take care of the issue of vanishing gradients while training. In particular the bias vector in the hidden layer of the ANO will absorbs the shift provided by residual connection and it will not affect the ANO. Hidden layer of ANO with residual connection is given by

$$v_l(x) = \underbrace{\sigma\left(Wv_{l-1}(x) + b + \fint v_{l-1}(y)dy\right)}_{\text{Original ANO}} + \underbrace{v_{l-1}(x)}_{\text{Residual connection}}$$

Hence $v_{l-1}(x) = v_{l-2}(x) + v'(x)$ where $v'(x) = \sigma\left(Wv_{l-2}(x) + b + \fint v_{l-2}(y)dy\right)$ be the output vector of original ANO without residual connection, then

$$\begin{aligned} v_l(x) &= \sigma\left(Wv_{l-1}(x) + b + \fint (v_{l-1}(y) + v'(y))dy\right) + v_{l-1}(x) \\ &= \sigma\left(Wv_{l-1}(x) + b + \fint v_{l-1}(y)dy + \fint v'(years)dy\right) + v_{l-1}(x) \\ &= \sigma\left(Wv_{l-1}(x) + b' + \fint v_{l-1}(y)dy\right) + v_{l-1}(x) \end{aligned}$$

where $b' = b + \fint v'(y)dy$ be a new bias vector. Hence the core structure of ANO is still preserved even with the shifted input and the core structure is proved to be universal in [4] which makes this *ResANO* **(ANO with a residual connection) a universal approximator**

In particular, the ConvNeXt block is a type of *ResANO* as shown above that it has ANO type nonlocailty and a residual connection. Hence, it is an universal approximator. ■

Setup. Now we show that the windowed multi-head self attention is equivalent to averaging. Notice that we can choose the query, key and value matrices, $Q_l = XW^Q$, $K = XW^K$ and $V = XW^V$ respectively, where $X \in \mathbb{R}^{m\times C}$, m is the total number of patches and X is the input sequence vector. Here W^* are the trainable matrices in $\mathbb{R}^{C\times C}$. For the proof of the lemma below, we set $m = 1$ and $W^V = X^\dagger$ (pseudo inverse of X) then the above quantities will be simplified: $X \in \mathbb{R}^{1\times C}$, Q_l, K_l and V_l will be in $\mathbb{R}^{1\times C}$. With these choices, we have the following lemma.

Lemma 2 (W-MSA is equivalent to average). *Windowed multi-head self attention as given by A.3 is equivalent to averaging with Q_l, K_l and V_l as given in the above paragraph and the following choices: take $\mathbf{W}_l^h = \alpha^{-1}\mathbb{I}$ where $\mathbb{I}$ is identity matrix and $\alpha = XX^\dagger$, $H = 1$ (number of attention head, without loss of generality), $D_{q_x}^l = D$ $\forall l$ and $\mathbf{B}_l(x,y)$ to be a zero vector. Then we have a simplified version of W-MSA as*

$$W - MSA(v)(x) = \alpha^{-1} \int_D \frac{e^{cos(\mathbf{Q}_l\mathbf{v}(x), \mathbf{K}_l\mathbf{v}(y))}}{\int_D e^{cos(\mathbf{Q}_l\mathbf{v}(z), \mathbf{K}_l\mathbf{v}(y))}dz} \mathbf{V}_l\mathbf{v}(y)dy.$$

This is equivalent to the nonlocailty of ANO (averaging over the domain).

Proof. Since Q_l and K_l aew in $\mathbb{R}^{1\times C}$, the product of these matrices with $v(y) \in \mathbb{R}^{C\times 1}\ \forall y \in D$ is a scalar. Also, $\cos(x, y) = \frac{x^\top y}{||x||||y||}$ for vectors x, y and for scalar it will becomes 1. With the choice of $W^V = X^\dagger$, we have $V = XX^\dagger = \alpha$ (let) (see above paragraph) then

$$\begin{aligned} W - MSA(v)(x) &= \alpha^{-1} \int_D \frac{e^{cos(\mathbf{Q}_l \mathbf{v}(x), \mathbf{K}_l \mathbf{v}(y))}}{\int_D e^{cos(\mathbf{Q}_l \mathbf{v}(z), \mathbf{K}_l \mathbf{v}(y))} dz} \mathbf{V}_l \mathbf{v}(y) dy \\ &= \alpha^{-1} \int_D \frac{e}{e \int_D 1 dz} XX^\dagger v(y) dy \\ &= \alpha^{-1} \int_D \frac{1}{|D|} \alpha v(y) dy = \frac{1}{|D|} \int_D v(y) dy \end{aligned}$$

■

Theorem 1 (Universal Approximation for SwinV2 transformer). *The hidden layer of SwinV2, $SW_l : C(D; \mathbb{R}^C) \to C(D; \mathbb{R}^C)$,*

$$v_l = SW_l(v_{l-1}) = v_{l-1} + LN_{\alpha_2^l, \beta_2^l} \left(\overline{W} \sigma \left(W v_{l-1} + W LN_{\alpha_1^l, \beta_1^l} (W - MSA(v_{l-1})) + \hat{B} \right) \right)$$

is equivalent to the hidden layer of ResANO for the choice: $\overline{W}, W = \mathbb{I}$ (identity matrix) and $\hat{B}$ to be a zero vector. Since, ResANO is universal as per the arguments given in paragraph 3, we have SwinV2 as universal appropriator.

Proof. With the above choices, the simplified form of hidden layer is given by

$$v_l = SW_l(v_{l-1}) = v_{l-1} + LN_{\alpha_2^l, \beta_2^l} \left(\sigma \left(v_{l-1} + LN_{\alpha_1^l, \beta_1^l} (W - MSA(v_{l-1})) \right) \right)$$

By Lemma 2, $W - MSA$ is equivalent to an average over the domain D and then applying $LN_{\alpha_1^l, \beta_1^l}$ will not affect the nonlocality as per the arguments given in paragraph 3. Now we have

$$\sigma \left(\mathbb{I} v_{l-1} + LN_{\alpha_1^l, \beta_1^l} \left(\frac{1}{|D|} \int_D v_{l-1}(y) dy \right) \right)$$

and this is exactly the hidden layer of ANO [4] with $W = \mathbb{I}$ since LN does not affect the nonlocality. Similarly, applying $LN_{\alpha_2^l, \beta_2^l}$ will also not affect the nonlocality (see paragraph 3). Then applying residual connection $(\cdot + v_{l-1})$ will result in a *ResANO* (ANO hidden layer with residual connection) which is an universal approximator as paragraph 3 shows.

This completes the proof that SwinV2 transformer block is an universal appropriator. ■

Theorem 2 (Universal Approximation for POSEIDON)**.** *With the given setup (see paragraph 3), the POSEIDON architecture is an universal approximator since it is a collection of SwinV2 transformer and ConvNeXt blocks which themselves are universal appropriators by Theorem 1 and Lemma 1 respectively.*

Proof. As shown in the figure 1, the POSEIDON model is a collection of SwinV2 transformer blocks, ConvNeXt blocks along with some operations such as patch merging, patch expansion, embedding and recovery. These operations are used for transforming the outputs of either the SwinV2 transformer block (patch merging and expansion) or of the whole model (recovery). They change the dimensions of the output using *linear* transformations and can be seen as lifting/projection opereators. These operations have practical implementation importance but they do not seem have a role in universal approximation property for the model.

Since each block of the model can be reduced to ANO (or *ResANO*), hence a universal approximator by Theorem 2.1 of [4], the POSEIDON model is also a universal approximator. ■

Appendix A. Architecture of the scalable Operator Transformer (scOT)

A.1. **Operator Learning with scOT.** We will discuss first how scOT transforms function space inputs into function outputs below.

For simplicity, we set $d = 2$ and specify $D = [0,1]^2$ as the underlying domain. A uniform computational grid is set with grid spacing Δ of J^2 equally spaced points $x_{j_x,j_y} = (j_x\Delta, j_y\Delta)$, with $J = 1/\Delta$. Let $1 < p < J$ such that $J\mathrm{mod}p = 0$ and set $P = J/p$. We divide the domain $D = \cup_{\rho=1}^{P^2} D_\rho$ into a set of P^2 non-overlapping and equal (in measure) patches. Any underlying input function $a \in C(D;\mathbb{R}^n)$ is then *partitioned* into a function that is piecewise constant on patches embedded into a $C-$dimensional latent representation by applying the operator,

$$v(x) = \hat{E}(a)(x) = \sum_{\rho=1}^{J^2} \mathbf{F}\left(\int_{D_\rho} W(x)a(x)dx\right)\mathbb{I}_{D_\rho}(x), \tag{A.1}$$

with $\mathbf{F} \in \mathbb{R}^{C\times n}$ is a learnable matrix and the weight function W is defined in terms of the underlying computational gris as $W(x) = \sum_{1\le j_x,j_y\le J}^{J} W_{ij}\delta_{x_{j_xj_y}}$, with δ denoting the Dirac measure, and the shared (across all patches) learnable weights are given by,

$$W_{j_xj_y} = \begin{cases}\omega_{j_xj_y} & \text{if } 1 \le j_x, j_y \le p \\ \omega_{j_x\mathrm{mod}p, j_y\mathrm{mod}p}, & \text{otherwise.}\end{cases} \tag{A.2}$$

The (patched and embedded) output function v of A.1 is then processes through a sequence of *SwinV2 transformer* blocks, each of which has the structure of $SW_l : C(D;\mathbb{R}^C) \mapsto C(D;\mathbb{R}^C)$, for layer index $l = 1,\dots,L$, formulated in the Main text (2.3).

The main building block of SwinV2 transformer block is the *windowed multi-head self attention* operator,

$$W - MSA^l(\mathbf{v})(x) = \sum_{h=1}^{H} \mathbf{W}_l^h \int_{D_{q_x}^l} \frac{e^{(cos(\mathbf{Q}_l^h\mathbf{v}(x),\mathbf{K}_l^h\mathbf{v}(y))+\mathbf{B}_l^h(x,y))}}{\int_{D_{q_x}^l} e^{(cos(\mathbf{Q}_l^h\mathbf{v}(z),\mathbf{K}_l^h\mathbf{v}(y))+\mathbf{B}_l^h(z,y))}dz}\mathbf{V}_l^h\mathbf{v}(y)dy, \tag{A.3}$$

for any $\mathbf{v} \in C(D;\mathbb{R}^C)$. Here, h denotes the h-th attention head, $\mathbf{W}_l^h \in \mathbb{R}^{C\times m}$ be the output matrix and $\mathbf{Q}_l^h, \mathbf{K}_l^h, \mathbf{V}_l^h \in \mathbb{R}^{m\times C}$ be the *query, key* and *value* matrices, respectively. For any two vectors α,β, the cosine similarity is defined as $\langle\alpha,\beta\rangle = |\alpha||\beta|\cos(\theta)$, θ is the angle between them, and $\mathbf{B}_l^h : D\times D \mapsto \mathbb{R}$ is a general form for *positional encodings.* We use *relative log position encodings* by setting the inputs to $\mathbf{B}_l^h$ to be the logarithm of the relative positions $(k,\overline{k})$ within the window and the function $\mathbf{B}_l^h$ itself to be a small MLP. Finally, the domain of integration $D_{q_x}^l$ is simply the window where the point of interest x lies, i.e. $1 \le q_x \le M^2$ such that $x \in D_{q_x}^l$ and $D = \cup_{q=1}^{M^2} D_q^l$, with $1 \le l \le L$ indexing the underlying layer within a SwinV2 transformer block. M^2 denoting the number of windows. Moreover, the windows are shifted across layers so that all the points can be attended to, by iteratively shifting windows across multiple layers/blocks.

The MLP is of the form, $MLP : C(D;\mathbb{R}^C) \mapsto C(D;\mathbb{R}^C)$ with

$$MLP(\mathbf{v})(x) = \overline{W}\sigma\left(W\mathbf{v}(x) + \hat{B}\right), \tag{A.4}$$

for the learnable weights matrices $W \in \mathbb{R}^{\overline{C}\times C}, \overline{W} \in \mathbb{R}^{C\times\overline{C}}$, bias vector $\hat{B} \in \mathbb{R}^{\overline{C}}$ and nonlinear activation function $\sigma : \mathbb{R} \mapsto \mathbb{R}$.

References

[1] 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, 2021. URL `https://arxiv.org/abs/2010.11929`.

[2] Zhongkai Hao, Zhengyi Wang, Hang Su, Chengyang Ying, Yinpeng Dong, Songming Liu, Ze Cheng, Jian Song, and Jun Zhu. GNOT: A general neural operator transformer for operator learning, 2023. URL `https://arxiv.org/abs/2302.14376`.

[3] Maximilian Herde, Bogdan Raonić, Tobias Rohner, Roger Käppeli, Roberto Molinaro, Emmanuel de Bézenac, and Siddhartha Mishra. Poseidon: Efficient foundation models for pdes, 2024. URL `https://arxiv.org/abs/2405.19101`.

[4] Samuel Lanthaler, Zongyi Li, and Andrew M. Stuart. Nonlocality and nonlinearity implies universality in operator learning, 2024. URL `https://arxiv.org/abs/2304.13221`.

[5] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations, 2021. URL `https://arxiv.org/abs/2010.08895`.

[6] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows, 2021. URL `https://arxiv.org/abs/2103.14030`.

[7] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, Furu Wei, and Baining Guo. Swin transformer v2: Scaling up capacity and resolution, 2022. URL `https://arxiv.org/abs/2111.09883`.

[8] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s, 2022. URL `https://arxiv.org/abs/2201.03545`.

[9] Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, March 2021. ISSN 2522-5839. doi: 10.1038/s42256-021-00302-5. URL `http://dx.doi.org/10.1038/s42256-021-00302-5`.

[10] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer, 2017. URL `https://arxiv.org/abs/1709.07871`.

[11] Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W. Battaglia. Learning mesh-based simulation with graph networks, 2021. URL `https://arxiv.org/abs/2010.03409`.

[12] Michael Prasthofer, Tim De Ryck, and Siddhartha Mishra. Variable-input deep operator networks, 2022. URL `https://arxiv.org/abs/2205.11404`.

[13] Bogdan Raonić, Roberto Molinaro, Tim De Ryck, Tobias Rohner, Francesca Bartolucci, Rima Alaifari, Siddhartha Mishra, and Emmanuel de Bézenac. Convolutional neural operators for robust and accurate learning of pdes, 2023. URL `https://arxiv.org/abs/2302.01178`.

[14] Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to simulate complex physics with graph networks, 2020. URL `https://arxiv.org/abs/2002.09405`.

[15] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023. URL `https://arxiv.org/abs/2302.13971`.

[16] Yinhao Zhu and Nicholas Zabaras. Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification. *Journal of Computational Physics*, 366:415–447, August 2018. ISSN 0021-9991. doi: 10.1016/j.jcp.2018.04.018. URL `http://dx.doi.org/10.1016/j.jcp.2018.04.018`.