WAVELET NEURAL OPERATOR: A NEURAL OPERATOR FOR PARAMETRIC PARTIAL DIFFERENTIAL EQUATIONS

Tapas Tripura
Department of Applied Mechanics
Indian Institute of Technology Delhi
tapas.t@am.iitd.ac.in

Souvik Chakraborty
Department of Applied Mechanics
Yardi School of Artificial Intelligence (ScAI)
Indian Institute of Technology Delhi
souvik@am.iitd.ac.in

ABSTRACT

With massive advancements in sensor technologies and Internet-of-things (IoT), we now have access to terabytes of historical data; however, there is a lack of clarity in how to best exploit the data to predict future events. One possible alternative in this context is to utilize operator learning algorithm that directly learn nonlinear mapping between two functional spaces; this facilitates real-time prediction of naturally arising complex evolutionary dynamics. In this work, we introduce a novel operator learning algorithm referred to as the Wavelet Neural Operator (WNO) that blends integral kernel with wavelet transformation. WNO harnesses the superiority of the wavelets in time-frequency localization of the functions and enables accurate tracking of patterns in spatial domain and effective learning of the functional mappings. Since the wavelets are localized in both time/space and frequency, WNO can provide high spatial and frequency resolution. This offers learning of the finer details of the parametric dependencies in the solution for complex problems. The efficacy and robustness of the proposed WNO is illustrated on a wide array of problems involving Burgers equation, Darcy flow, Navier-Stokes equation, Allen-Cahn equation, and Wave advection equation. Comparative study with respect to existing operator learning frameworks are presented. Finally, the proposed approach is used to build a digital twin capable of predicting Earths air temperature based on available historical data.

***K*eywords** Nonlinear mappings · Operator learning · Wavelet · Wavelet Neural Operator · Scientific machine learning.

Introduction

The application of partial differential equations (PDEs) for modelling of various physical and complex phenomena in scientific and engineering problems such as fluid flow, traffic flow, chemical reactions, biological growth etc [1, 2, 3]. are inevitable. In the absence of analytical solutions the large dimensional problems involving PDEs are traditionally solved using finite element [4], finite difference [5] and finite volume [6] based approaches. These traditional solvers are usually discretization dependent, due to which they are computationally expensive and requires independent forward runs for different parameter values in order to obtain the intended solutions. No need to mention the complexity of these traditional approaches as the dimension of the problem increases.In the recent years, neural networks (NN) have emerged as a possible alternative to the popular classical solvers. NNs approximate highly non-linear functions by passing the weighted sum of inputs through a non-linear activation function. NNs are universal approximators that can learn any functional mapping between two measurable space [7]. The learning of NNs are either data-driven [8, 9, 10, 11] or physics-informed [12, 13, 14] in nature. Physics-informed NNs are very accurate and ensures that the underlying governing law of the physical problem is satisfied. Therefore, if the physics of the underlying processes are known in advance then physics-informed learning of NN becomes practical. However, for most of the natural processes the governing physics are not know a-priory and training of NNs are done by resorting to data-driven approaches. One problem with the data-driven NNs is that they do not ensure the governing physics of the underlying problem, as a consequence they don not generalise beyond the training data. As an efficient and effective approach

the trained networks should generalize for any unseen input function and predict the output with as much as low error. Towards addressing this challenges, recently, NNs are generalised to neural operators.

Neural operators learn the mapping between two infinite-dimensional function spaces and therefore, they are required to be trained only once. Once trained they can be used for prediction of the solution for any given input function. Similar to NNs, neural operators also learn the complex nonlinear operators by passing the global integral operators through the nonlinear local activation functions. Neural operators also share the same network parameters between different discretizations that makes them discretization invariant. Since, they are discretization invariant they provide huge computational advantages over classical PDE solvers. Similar to universal function approximation theory, the neural operators work on the theorem of universal operator approximation by a single-layer of neural networks [15]. Among the recently developed neural operators the DeepONet architecture was the first to learn such infinite-dimensional function spaces [16, 17]. The DeepONet consists of two neural networks that are referred as branch and trunk NN. The branch NN is responsible for the input function and the trunk NN corresponds to the output function at some sensor point. The output in the DeeONet is then obtained from the inner product of these NNs. Its application can be found in reliability analysis [18], bubble dynamics [19], fracture mechanics [20] etc. In a different approach the graph neural operator (GNO) was proposed [21]. The GNO focuses on learning of the infinite-dimensional mapping by composition of nonlinear activation functions and certain class of integral operators. The kernel integration in the GNO is learned through message passing between the graph networks. Though development of GNO was completely different from classical NNs and gave new ideas of using graphs, the GNO has the tendency to become unstable with the increase of the number of hidden layers. In the same time, Fourier neural operator (FNO) was proposed with the aim to learn the parameters of the network in Fourier space [22]. The heart of the FNO is the spectral decomposition of the input using fast Fourier transform (FFT) and computation of the convolution integral kernel in the Fourier space. Simple yet accurate and efficient performance of the FNO was demonstrated on various state-of-the art problems including Burgers, Darcy and Navier-Stokes equations.

One major shortcoming of the FNO is that the basis functions in FFTs are generally frequency localised with no spatial resolution [23]. Therefore the FFTs are not suitable for studying the spatial behaviour of any signal or image, as a consequence, the performance of the FNO gets hindered in complex boundary conditions. In this context, one can think of wavelets which are both space and frequency localized [23, 24]. Since wavelets have spatial information they can handle signal with discontinuity and/or spikes better and therefore can learn the patterns in images better compared to FFTs. To explain further, wavelet provides us characteristic frequency and characteristic space information, with the help of which we can not only tell the frequencies present in the signal as a function of characteristic frequency but also the location of the frequency as a function of characteristic space information. The application of wavelets can be found in various aspects of data compression such as fingerprint compression [25], compressed sensing [26], iris recognition [27], signal denoising [28], motion analysis [29], fault detection [30], to name few. Use of wavelets in neural networks can also be found in the literature [31, 32, 33, 34].

In this work, we propose a new spectral decomposition based neural operator for learning mappings between infinite-dimensional spaces of functions and call this Wavelet Neural Operator (WNO). In the proposed WNO, the snapshots are first transformed to its high and low frequency components using discrete wavelet transform [35, 36]. In general, the high frequency components represent the edges in an image and the low frequency components account for the smooth regions. Since in a convolution setup the prime aim is to detect the edges present in a image, the high frequency components are of more importance than the low frequency components in our case. Thus, the single tree discrete wavelet transform with multi-level discrete wavelet transform of the high frequency components are utilized in this work. The proposed WNO harnesses the benefits of the wavelets and can learn solution operators of highly nonlinear parametric PDEs with both smooth and discontinuity either on the boundary conditions or within the domain. It can further be shown that, if the mother wavelet basis in WNO is replaced by trigonometric functions then FNO becomes the special case of WNO. Overall, the salient points of the proposed WNO architecture are as follows:

- The network parameters in the proposed WNO are learned in the wavelet space that are both frequency and spatial localized, thereby can learn the patterns in the images and/or signals more effectively.
- The proposed WNO can handle highly nonlinear family of PDEs with discontinuities and abrupt changes in the solution domain and the boundary.
- The accuracy of the proposed WNO is consistent for both smooth and complex geometry.
- The proposed WNO can learn the frequency of changes in the patterns over the solution domain, which makes it amenable for online implantation's without requiring for the entire dataset. Thus proposed WNO can be generalised to images and videos, as a consequence proposed WNO can also be implemented for short-to-medium range climate modelling and weather forecast purposes.

Due to these reasons, WNO can be straightforwardly applied to a wide array of problems including fluid mechanics, traffic flow modelling, and climate modelling. The efficacy and robustness of the proposed WNO is illustrated using several examples involving Burger's equation, Darcy flow, Navier-Stokes equation, and Wave advection equation. Comparative study with respect to other existing neural operators has been presented. Finally, we utilize the proposed WNO to develop a digital twin capable of predicting Earth's air temperature at 2m above the ground at a resolution of $2^\circ \times 2^\circ$ based on historical data.

Results

Operator learning through neural operator

An operator is responsible for mapping between infinite-dimensional input and output function spaces. To understand it better, let $D \in \mathbb{R}^d$ be the n-dimensional domain with boundary ∂D. For a fixed domain $D = (a, b)$ and $x \in D$, consider a PDE which maps the input function spaces i.e., the space containing the source term $f(x,t) : D \mapsto \mathbb{R}$, the initial condition $u(x,0) : D \mapsto \mathbb{R}$, and the boundary conditions $u(\partial D, t) : D \mapsto \mathbb{R}$ to the solution space $u(x,t) : D \mapsto \mathbb{R}$, with t being the time coordinate. In the present work, we focus on learning the operator that maps the input functions to the solution space $u(x,t)$. Now, let us define two complete normed vector spaces $(\mathcal{A}, \|.\|_{\mathcal{A}})$ and $(\mathcal{U}, \|.\|_{\mathcal{U}})$ of functions taking values in $\mathbb{R}^{d_a}$ and $\mathbb{R}^{d_u}$ with given number of partial derivatives. These function spaces are often called as Banach spaces and denoted as $\mathcal{A} := C(D; \mathbb{R}^{d_w})$ and $\mathcal{U} := C(D; \mathcal{R}^{d_u})$. For the given domain $D \subset \mathbb{R}^d$ and normed spaces $(\mathcal{A}, \|.\|_{\mathcal{A}})$ and $(\mathcal{U}, \|.\|_{\mathcal{U}})$, we aim at learning the nonlinear differential operator $\mathcal{D} : \mathcal{A} \mapsto \mathcal{U}$. The modern data-driven and physics-informed NNs center around the learning of the function $u(x,t)$ that satisfies the differential operator $\mathcal{D}$ in the domain D and boundary conditions ∂D. Although these approaches are cost effective than classical PDE solvers, they can still be computationally expensive due to the fact that they need to be trained multiple times for multiple set of PDE parameters $\boldsymbol{a} \in \mathcal{A}$. One elegant approach is to learn the nonlinear differential operator $\mathcal{D}$ so that the solutions to a family of PDEs can be obtained for different sets of input parameters $\boldsymbol{a} \in \mathcal{A}$, which results in computationally efficiency and has more practical utility.

Let, the input and output functions in $\mathcal{A}$ and $\mathcal{U}$ are denoted by $\boldsymbol{a} : D \mapsto a(x \in D) \in \mathbb{R}^{d_a}$ and $\boldsymbol{u} : D \mapsto u(x \in D) \in \mathbb{R}^{d_u}$. Further, consider that $u_j = \mathcal{D}(a_j)$ and we have access to N number of pairs $\{(a_j, u_j)\}_{j=1}^N$. Then, we aim to approximate $\mathcal{D}$ by neural networks from the pairs $\{(a_j, u_j)\}_{j=1}^N$ as,

$$\mathcal{D} : \mathcal{A} \times \boldsymbol{\theta}_{NN} \mapsto \mathcal{U}; \quad \boldsymbol{\theta}_{NN} = \{\mathcal{W}_{NN}, \boldsymbol{b}_{NN}\}, \tag{1}$$

where $\boldsymbol{\theta}_{NN}$ denotes the finite-dimensional parameter space of the neural networks. Although the input and output functions a_j and u_j are continuous, however, in the data-driven framework we assume the n point discretization $\{x_p\}_{p=1}^n$ of the domain D. Therefore, for the N collection of the input-output pair we assume that we have access to the datasets $\{a_j \in \mathbb{R}^{n \times d_a}, u_j \in \mathbb{R}^{n \times d_u}\}_{j=1}^N$. With this setup, the objective here is to develop a network that can exploit the data to learn the operator $\mathcal{D}$. To that end, we first lift the inputs $a(x,t) \in \mathcal{A}$ to some high dimensional space through a local transformation $P(a(x)) : \mathbb{R}^{d_a} \mapsto \mathbb{R}^{d_v}$ and denote the high dimensional space as $v_0(x) = P(a(x))$, where $v_0(x)$ takes values in $\mathbb{R}^{d_v}$. In a neural network framework, the local transformation $P(a(x))$ can be achieved by constructing a shallow fully connected neural network (FNN). Further, let us denote that a total number of l steps are required to achieve satisfactory convergence. Therefore, we apply l numbers of updates $v_{j+1} = G(v_j) \forall\, j = 1, \ldots, l$ on $v_0(x)$, where the function $G : \mathbb{R}^{d_v} \mapsto \mathcal{R}^{d_v}$ takes value in $\mathbb{R}^{d_v}$. Once all the updates are performed, we define another local transformation $Q(v_l(x)) : \mathbb{R}^{d_v} \mapsto \mathbb{R}^{d_u}$ and transform back the output $v_l(x)$ to the solution space $u(x) = Q(v_l(x))$. The step-wise updates $v_{j+1} = G(v_j)$ are defined as,

$$v_{j+1}(x) := g\left(\left(K(a;\phi) * v_j\right)(x) + W v_j(x)\right); \quad x \in D, \quad j \in [1, l] \tag{2}$$

where $g(\cdot) : \mathbb{R} \to \mathbb{R}$ is a non-linear activation function, $W : \mathbb{R}^{d_v} \to \mathbb{R}^{d_v}$ is a linear transformation, $K : \mathcal{A} \times \boldsymbol{\theta} \to \mathcal{L}(\mathcal{U}, \mathcal{U})$ is the integral operator on $C\left(D; \mathbb{R}^{d_v}\right)$. In the neural network setup, we represent $K(a;\phi)$ to be a kernel integral operator parameterized by $\phi \in \boldsymbol{\theta}$. The kernel integral operator $K(a;\phi)$ is defined as,

$$\left(K(a;\phi) * v_j\right)(x) := \int_{D \in \mathbb{R}^d} k\left(a(x,y), x, y; \phi\right) v_j(y) \mathrm{d}y; \quad x \in D, \quad j \in [1, l] \tag{3}$$

where $k_\phi : \mathbb{R}^{2d + d_a} \mapsto \mathbb{R}^{d_v \times d_v}$ is a neural network parameterized by $\phi \in \boldsymbol{\theta}$. Suppose, there is some kernel $k \in C\left(D; \mathbb{R}^{d_v}\right)$, then we want to employ the concept of degenerate kernels to learn the κ_ϕ from the data such that,

$$\| \left(k_\phi(a) - k(a)\right) v\| \leq \| \left(k_\phi(a) - k(a)\right) \| \|v\|, \tag{4}$$

and $k_\phi \rightarrow k$. Since the kernel $k \in C(D)$ and $K \in \mathcal{L}(C(D;\mathbb{R}^{d_v}))$ together defines the infinite-dimensional spaces, therefore, Eq. (2) can learn the mapping of any infinite-dimensional function space. As far as the approximation of nonlinear operators are concerned the structure of Eq. (2) follows the conventional neural networks. We propose a new neural operator referred to as the WNO that learns the mapping with higher order of accuracy. One key component of the proposed WNO is the wavelet kernel integral layer. The overarching framework of the proposed WNO is presented in Fig. 1. While Fig. 1(a) provides the schematic description of proposed WNO, in Fig. 1(b), a simplistic WNO with wavelet kernel integral layer is shown for easy understanding.

For training the network, we define an appropriate loss function $\mathcal{L} = \mathcal{U} \times \mathcal{U}$ as follows:

$$\boldsymbol{\theta}_{NN} = \arg\min_{\boldsymbol{\theta}_{NN}} \mathcal{L}\left(\mathcal{D}(\boldsymbol{a}), \mathcal{D}(\boldsymbol{a}, \boldsymbol{\theta})\right). \tag{5}$$

We leverage the universal approximation theorems of the neural networks [7, 15] and try to learn the parameter space (weights-$\mathcal{W}_{NN}$ and biases-$\boldsymbol{b}_{NN}$ of neural networks) by approaching to the loss function from a data-driven perspective.

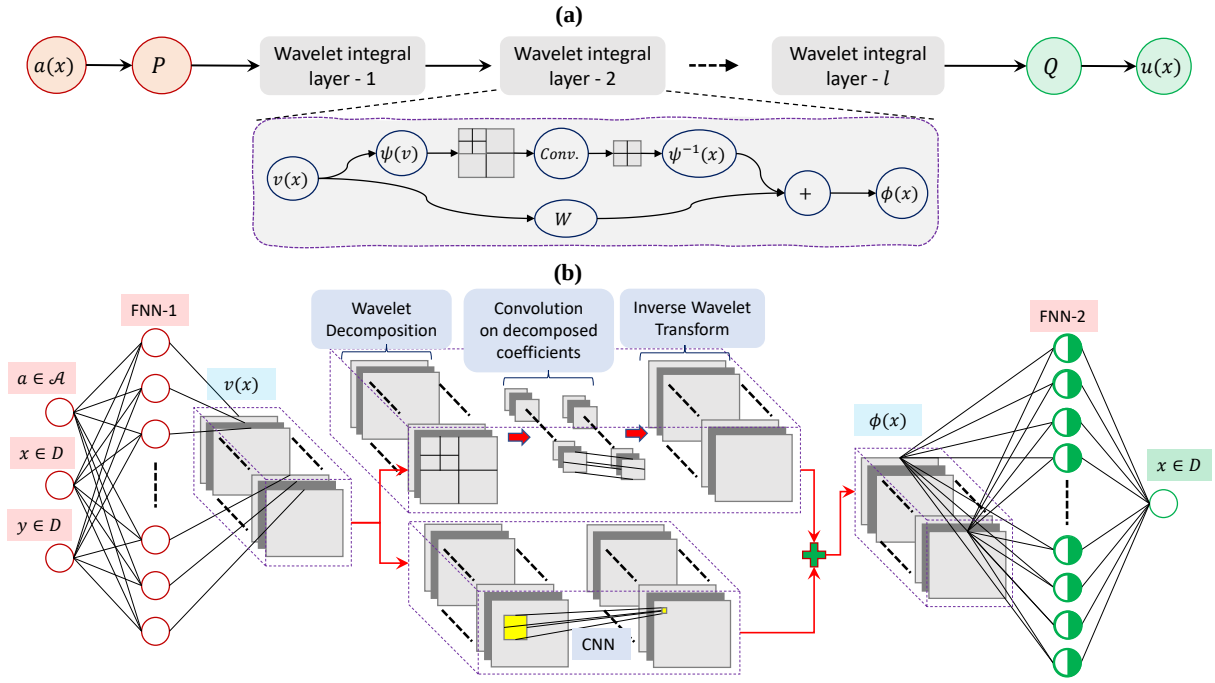

Figure 1: **Illustration of the architecture of the wavelet neural operator (WNO)**. (a) **Schematic of the proposed neural operator**. First lift the inputs to a higher dimension by a local transformation $P(\cdot)$. Then pass the lifted inputs to a series of wavelet kernel integral layer. Transform back the final output of the wavelet integral layer using a local transformation $Q(\cdot)$. Finally, activate the output of $Q(\cdot)$, which will provide the solution $u(x)$. In the wavelet kernel integral layer, the multilevel wavelet decomposition of the inputs are performed first, as a result of which the horizontal, vertical and diagonal coefficients at different levels are obtained. The convolution between neural network weights and the subband coefficients of last level are then performed. The inverse wavelet transform is performed on the convoluted coefficients to transform the reduced inputs to original dimension. Suitable activation is done on the output of the wavelet kernel integration layer. (b) **A simple WNO with one wavelet kernel integral layer**. The inputs contain the initial parameters and space information. The local transformations $P(\cdot)$ and $Q(\cdot)$ are modelled as shallow fully connected neural network. The output of $P(\cdot)$ is fed into the wavelet integral layer. The integral layer consists of two separate branches. In the first branch the wavelet decomposition of inputs followed by parameterization of integral kernel is done. In the second branch a convolution neural network (CNN) with kernel size 1 is constructed. The outputs of the two branches are then summed and activation's are performed. Then the outputs are passed through the transformation $Q(\cdot)$, which provides the target solution $u(x)$. In similar manner, a WNO with arbitrary number of wavelet integral layers can be be constructed.

Operator learning of example problems through WNO

Numerical case studies on a wide variety of systems representing various class of problems in fluid dynamics, gas dynamics, traffic flow, atmospheric science and phase field modelling are considered in this section. In all the case, the performance of the network architectures were compared using the L^2 relative error of the predictions. We compare the results obtained with four popular neural operators and their variants: (a) DeepONet [17, 16], (b) Graph Neural Operator (GNO), (c) Fourier Neural Operator (FNO) [22], and (d) Multiwavelet Transformation operator (MWT) [34]. The overall WNO architecture consists of four layers of WNO, each of which are activated by the GeLU activation function [37]. The mother wavelet is chosen from the Daubechies family [38, 39]. For all the cases, the sizes of the individual datasets, the wavelet, the level of wavelet tree and the number of layers in the network against each problem are listed in the Table. 1. For optimization of the parameters of the WNO, the ADAM optimizer with initial learning rate of 0.001 and a weight decay of 10^{-6} is utilized. During the optimization, decay in the learning rate of each parameter group are introduced at every 50 epochs at a rate of 0.75. The total number of epochs used for training the WNO architecture against the undertaken examples are given in Table 2. The batch size in the data loader varies according to the underlying system between 10-25. The numerical experiments are performed on a single RTX A2000 6GB GPU. A summary of results for different example problems solved are shown in Table 3. It is observed that the error for the proposed WNO varied in the range of 0.21% – 1.75% with WNO yielding the best results (lowest error) in four out of the six cases. In the remaining two problems also, the results obtained using the proposed WNO are highly accurate (and close to the best results).

Table 1: Dataset size for each problem, unless otherwise stated. WNO architecture for each problem, unless otherwise stated.

Examples	Number of data		Wavelet	m	Network dimensions				$g(.)$
	Training	Testing			FNN1	FNN2	CNN	WNO	
Burgers	1000	100	db6	8	64	128	4	4	GeLU
Darcy (rectangular)	1000	100	db4	4	64	128	4	4	GeLU
Darcy (triangular)	1900	100	db6	3	64	128	4	4	GeLU
Navier-Stokes	1000	100	db4	3	26	128	4	4	GeLU
Allen-Cahn	1400	100	db4	1	64	128	4	4	GeLU
Advection	900	100	db6	3	96	128	4	4	GeLU

Table 2: Number of epochs required to obtain the best prediction results. Note that the number of epochs here for DeepONet and FNO, except the Allen-Cahn example are reported from their respective original papers.

Architectures	Number of epochs					
	Burgers'	Darcy flow	Navier-Stokes	Allen-Cahn	Darcy (Notch)	Advection
DeepONet	500000	100000	100000	100000	20000	250000
POD-DeepONet	500000	100000	100000	100000	20000	250000
FNO	500	500	500	500	500	500
MWT	500	500	500	500	500	500
WNO	500	500	500	500	500	500

1D Burgers equation: In the first example, we consider the 1D Burgers equation. The 1D Burgers equation is popularly used in modelling of 1D flows in fluid mechanics, gas dynamics, and traffic flow. The 1D Burgers equation is given by the form, $\partial_t u(x,t) + \frac{1}{2}\partial_x u^2(x,t) = \nu \partial_{xx} u(x,t); x \in (0,1), t \in (0,1]$, subjected to the initial condition $u(x,0) = u_0(x); x \in (0,1)$, where, $\nu \in \mathbb{R}^{+*}$ is the viscosity of the flow and $u_0(x) \in L^2_{per}((0,1);\mathbb{R})$. A periodic boundary condition of the form, $u(x-\pi,t) = u(x+\pi,t); x \in (0,1), t \in (0,1]$ is considered. The aim here is to learn the operator, $\mathcal{D} : u_0(x) \mapsto u(x,1)$. The datasets are taken from the Ref. [22], where ν=0.1 and a spatial resolution of 1024 is considered. **Results**: the prediction results along with their accuracy are illustrated in the Fig. 2 and Table 3. From the error values in the Table. 3, it is evident that the MWT obtains the lowest error followed by FNO. The proposed WNO has slightly higher error than FNO but lesser than DeepONet and POD-DeepONet. Although from the quantitative standpoint the differences in the error in the predictions using WNO is slightly higher than MWT and FNO, from Fig. 2, the differences in the predictions and true results are not discerned. This indicates that, although not the best, the proposed WNO performs reasonably well for this problem.

2D Darcy flow equation in a rectangular domain: The 2D Darcy flow equation is widely used in modelling of flow, whether liquid or gas, through a porous medium. In the absence of the time component it represents a second order nonlinear elliptic PDE and the form in a 2D domain is given as, $-\nabla \cdot (a(x,y)\nabla u(x,y)) = f(x,y); x,y \in (0,\mathbb{R})$, subjected to the initial condition, $u(x,y) = u_0(x,y); x,y \in \partial(0,\mathbb{R})$, where $a(x,y)$ is the permeability field, $u(x,y)$ is the pressure,

Table 3: Mean L^2 relative error between the true and predicted results.

PDE	Network Architectures						
	GNO	DeepONet	FNO	MWT	POD-DeepONet‡	dgFNO+†	WNO
Burgers' equation	≈ 6.15 %	≈ 2.15 %	≈ 1.60 %	≈ 0.19 %	≈ 1.94 %	-	≈ 1.75 %
Darcy (rectangular)	≈ 3.46 %	≈ 2.98 %	≈ 1.08 %	≈ 0.89 %	≈ 2.38%	-	≈ 0.84 %
Darcy (triangular)	-	≈ 2.64 %	-	≈ 0.87 %	≈ 1.00 %	≈ 7.82%	≈ 0.77 %
Navier-Stokes equation	-	≈ 1.78 %	≈ 1.28 %	≈ 0.63 %	≈ 1.71 %	-	≈ 0.31 %
Allen-Cahn	-	≈ 17.7 %	≈ 0.93 %	≈ 4.84 %	-	-	≈ 0.21 %
Wave-advection	-	≈ 0.32 %	≈ 47.7 %	≈ 10.22 %	≈ 0.40 %	≈ 0.62 %	≈ 0.62 %

‡ POD-DeepONet is the modified version of DeepONet, proposed in Ref. [17]. † Note that in complex geometric condition FNO does not work, thus we use dgFNO+, which is a modified version of FNO (Ref. [17]).

$\nabla u(x, y)$ is the pressure gradient, and $f(x, y)$ is a source function. In this problem definition, the Darcy flow is defined on a unit box domain with $x \times y \in (0, 1)^2$ with $u_0(x, y) = 0$ (zero Dirichlet boundary condition). The aim is to learn the operator, $\mathcal{D} : a(x, y) \mapsto u(x, y)$. The datasets for training the network architectures are taken from the Ref. [22]. During training a spatial resolution of 85×85 is used. **Results**: the predictions for the 2D Darcy flow in rectangular domain is plotted in Fig. 2, while the associated errors are given in Table. 3. In this case, it is observed that the mean prediction error for WNO is lowest among all the considered methods. It is further evident in the Fig. 2, where the differences between true and predicted solutions can not be discerned in naked eye.

2D Darcy flow equation with a notch in triangular domain : This example emulates the previous 2D Darcy problem but with more complex boundary condition. The setup and datasets for this case are taken from the Ref. [17], where the boundary conditions for the triangular domain are generated using the following Gaussian process (GP), $u(x) \sim \text{GP}(0, \mathcal{K}(x, x'))$. The kernel is taken as $\mathcal{K}(x, x') = \exp\left(-(x - x')^2/2l^2\right); l = 0.2; x, x' \in [0, 1]$. In addition to the triangular geometry, a notch is introduced in the flow medium. The permeability field $a(x, y)$ and the forcing function $f(x, y)$ are set as 0.1 and -1, respectively. The aim here is to learn the operator, responsible for the mapping of the boundary conditions to the pressure field in the entire domain, represented as, $\mathcal{D} : u(x, y)|_{\partial\omega} \mapsto u(x, y)$. **Results**: the testing results with corresponding testing errors are given in Fig. 2 and Table 2, respectively. From the error values in table it can be observed that our proposed WNO has the lowest prediction error, which is more than three fold lower than DeepONet and ten fold less than FNO. The errors between the truth and prediction can also be observed in Fig. 2, that shows that the WNO is able to map the given boundary conditions to the pressure filed correctly. Upon careful observation, it can further be noticed that the predictions are not only correct in the smooth regions but also provides near perfect match near the slit. This indicates that the proposed WNO can be applied for effective and accurate learning of nonlinear operators in complex geometric conditions.

2D time dependent Navier-Stokes equation: The Navier-Stokes equation is a second order nonlinear parabolic PDE and its application can be found in various facets of fluid flow problems such as air flow around the aeroplane wing, ocean currents, thermodynamics, etc. The 2D incompressible Navier-Stokes equation in the vorticity-velocity form is given by the equations, $\partial_t \omega(x, y, t) + u(x, y, t) \cdot \nabla\omega(x, y, t) = \nu\Delta\omega(x, y, t) + f(x, y); x, y \in (0, 1), t \in (0, T]$ and $\nabla \cdot u(x, y, t) = 0; x, y \in (0, 1), t \in [0, T]$, where $\nu \in \mathbb{R}$ and $f(x, y)$ are the viscosity of the fluid and source function. The initial vorticity field is taken as $\omega(x, y, 0) = \omega_0(x, y); x, y \in (0, 1)$. The terms $u(x, y, t)$ and $\omega(x, y, t)$ are the velocity and vorticity field of the fluid, respectively. The viscosity in our experiment is considered as $\nu = 10^{-3}$. The aim here is to learn the operator responsible for the mapping between the vorticity field at the time steps $t \in [0, 10]$ to the vorticity at a target time step $t \in (10, T]$ i.e., $\omega|_{(0,1)^2\times[0,10]} \mapsto \omega|_{(0,1)^2\times(10,T]}$. The initial vorticity $\omega_0(x, y)$ is simulated from a Gaussian random field. For this example, the datasets for initial vorticity $\omega_0(x, y)$ and the solution $\omega_{10}(x, y)$ at time $t = 10$ are taken from Ref. [22]. For both training and testing, the spatial resolution of the vorticity fields are fixed at resolutions 64×64. **Results**: the prediction results for vorticity field at $t \in \{10, 20\}$ are given in Fig. 2, along with the mean prediction errors in Table 3. The results show that the prediction results of the proposed WNO has the lowest error and emulates the true solutions almost accurately.

2D Allen-Cahn equation: The AllenCahn equation is a reactiondiffusion equation and often used in the context of chemical reactions and modelling of phase separation in multi-component alloy. The Allen-Cahn equation in 2-dimensional space is given as, $\partial_t u(x, y, t) = \epsilon\Delta u(x, y, t) + u(x, y, t) - u(x, y, t)^3; x, y \in (0, 3)$, subjected to the initial condition $u(x, y, 0) = u_0(x, y); x, y \in (0, 3)$, where $\epsilon \in \mathbb{R}^{+*}$ is a real positive constant and responsible for the amount of diffusion. The problem is defined on periodic boundary with $\epsilon = 10^{-3}$ and the initial condition is simulated from the Gaussian Random Field using the following kernel, $\mathcal{K}(x, y) = \tau^{(\alpha-1)}\left(\pi^2(x^2 + y^2) + \tau^2\right)^{\alpha/2}$ with $\tau = 15$ and $\alpha = 1$. The aim here is to learn the operator $\mathcal{D} : u_0(x, y) \mapsto u(x, y, t)$. In this case, t=20s is taken and the solution is obtained on a grid of 43×43. **Results**: The prediction results with mean prediction error of the proposed WNO for different

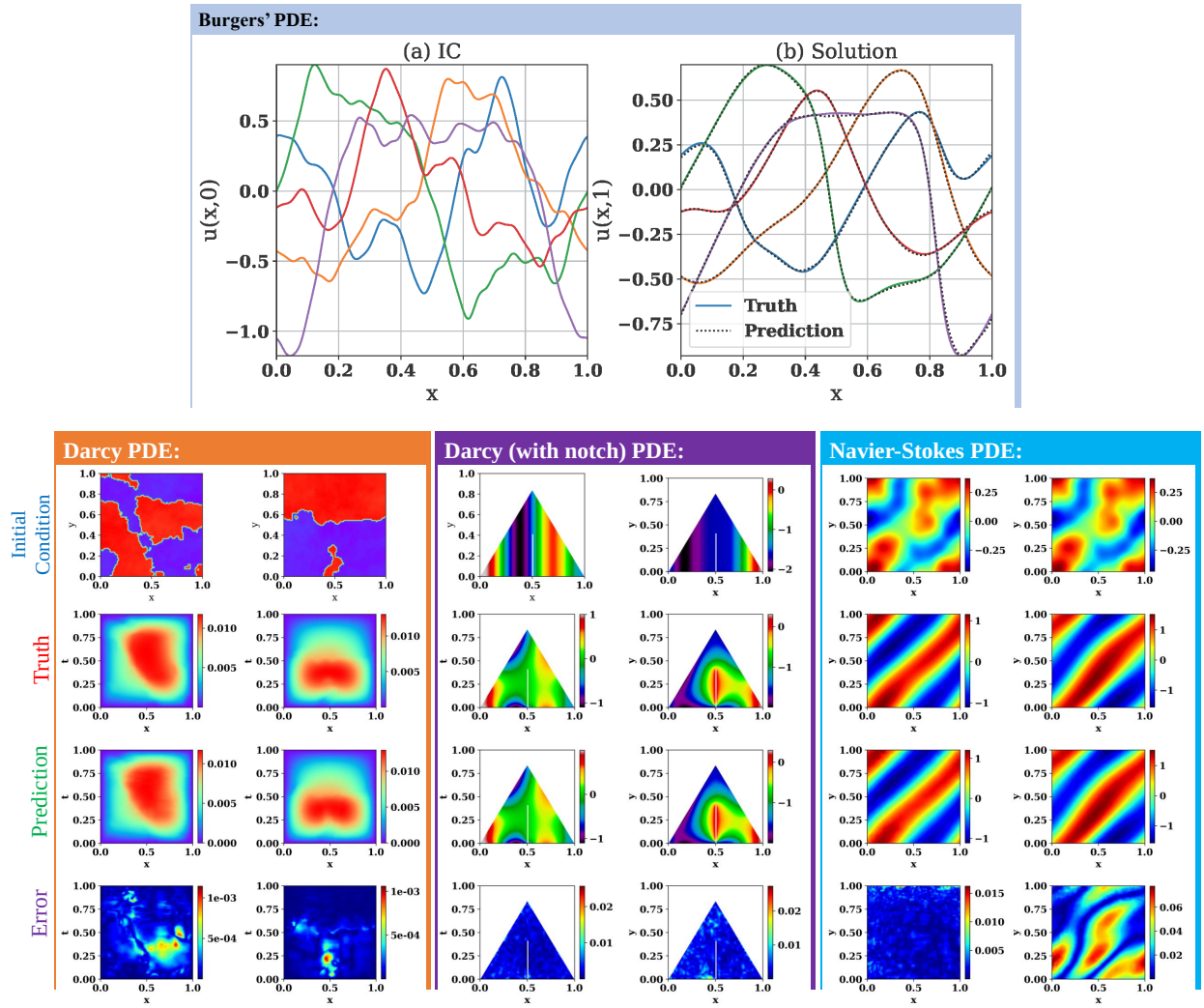

Figure 2: **Operator learning of flow problems**. First row: **Burgers' PDE with 1024 spatial resolution**. (a) the initial conditions. (b) The solution to the 1D Burgers flow equation at $t = 1$. The aim was to learn the differential operator of Burgers' PDE. For given input functions, the predictions are shown and it can be seen that the predictions of proposed WNO matches with the true solution almost exactly. **Darcy flow in the rectangular domain with spatial resolution of 85×85**. The aim is to learn the differential operator of Darcy equation. The prediction of output pressure fields from two different input permeability conditions are shown in the figure. The errors indicate that the predictions match the true solutions almost exactly. **Darcy flow in a triangular domain with a notch**. The goal here is to test the performance of proposed WNO for learning nonlinear operators in complex geometric conditions. For the given boundary conditions, the pressure fields obtained using the proposed WNO are shown in the figure. The errors against each of the predictions depict that the proposed WNO can learn operators in complex geometric conditions also. **Navier-Stokes equation with spatial resolution of 64×64**. In this example, the aim is to use WNO for learning of differential operators of 2D time dependent problems. The initial vorticity (top row) and the predictions at $t = 11$s (second row first column) and $t = 20$s (second row second column) are shown. The errors indicates that the proposed WNO can learn highly nonlinear operators.

input functions are given in Fig. 3 and Table 3. The mean errors in Table 3 indicates that the DeepONet and MWT has significant error. The FNO has shown good performance with $\approx 1\%$ error, however, our proposed WNO obtains the predictions with lowest error which is closes to 0.2%. In Fig. 3, it is straightforward to see that the true and prediction results are non-differentiable for all the given input functions. This further illustrates that the proposed WNO can be successfully implemented for any 2D time dependent highly nonlinear problems.

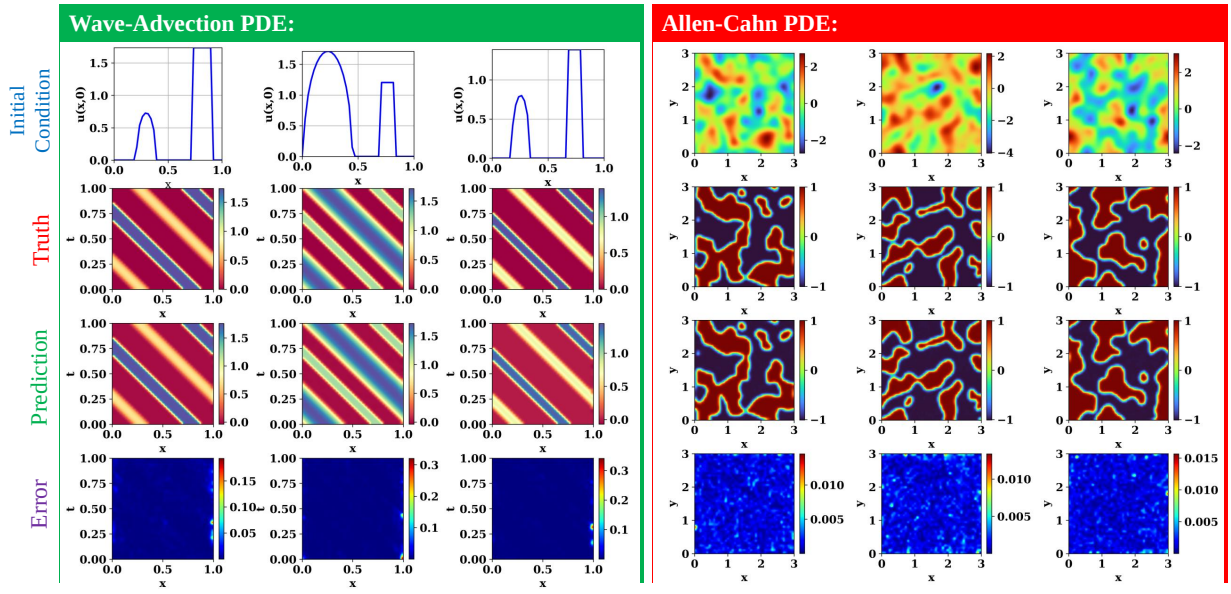

Figure 3: **Operator learning of Allen-Cahn reactor-diffusion equation and wave advection equation**. **Wave advection PDE**: For solution, a spatio-temporal resolution of 40×40 is used. The proposed WNO is used here to learn the operator that can predict the wave propagation through continuous medium. Three independent initial conditions (IC) were taken, for which the true, prediction and corresponding errors are plotted. The error plots demonstrate that the prediction results of proposed WNO is very accurate. **Allen-Cahn PDE**: For three different initial conditions, the final solutions at $t = 20$s are plotted. This PDE has chaotic nature, i.e., the solutions diverges significantly for small perturbations in initial conditions. The aim here is to test the ability of the proposed WNO for prediction of chaotic type solutions. The solutions are obtained on a spatial resolution of 43×43. Three different IC are taken and for each IC the true and WNO predicted solutions are shown. The errors against each case demonstrate that the proposed WNO could predict the true solutions almost exactly.

Wave advection equation: The wave advection equation is a hyperbolic PDE and primarily describes the solution of a scalar under some known velocity field. The advection equation with periodic boundary condition is given by the expression, $\partial_t u(x,t)+v\partial_x u(x,t) = 0; x \in (0,1), t \in (0,1)$ and $u(x-\pi,t) = u(x+\pi,t); x \in (0,1), t \in (0,1)$. Here $v \in \mathbb{R} > 0$ represents the speed of the flow. The initial condition is chosen as, $u(x,0) = h1_{\{c-\frac{\omega}{2},c+\frac{\omega}{2}\}} + \sqrt{\max(h^2-(a(x-c))^2,0)}$. where the variables ω and h represents the width and height of the square wave, respectively and the wave is centered at $x = c$. The values of $\{c,\omega,h\}$ are randomly chosen from $[0.3,0.7]\times[0.3,0.6]\times[1,2]$. For v=1, the solution to the advection equation is given as, $u(x,t) = u_0(x-t)$. The aim here is to learn the operator, that defines the mapping $\mathbb{G} : u_0(x) \mapsto u(x,t)$, for some later time t. The solutions are obtained on a spatial resolution of 40×40. The datasets for this example are taken from the Ref. [17]. **Results**: The prediction results along with their L^2 relative error are given in Fig. 3 and Table 3. Similar to the previous examples, it can be seen in Fig. 3 that the WNO is able to predict the scalar field for arbitrary input function. The corresponding errors show that the FNO and MWT suffers the most with $\approx$ 47% and $\approx$ 10% error, respectively. The DeepONet yields the lowest error, however, both POD-DeepONet, modified FNO and proposed WNO also obtains relatively small error (very close to the DeepONet results).

Weather forecast: prediction of the monthly mean 2m air-temperature

The previous examples have showcased the effectiveness of the proposed WNO framework in accurately solving parametric PDEs. In general sense, however, the solutions of the PDEs can be imagined as videos of snapshots of the time-evolving process. Since, the images and videos presents the state of an object at a particular time and their motions, respectively, they can be idealised as 2D and 2D time dependent problems. Thus, the applications of WNO can also be extended for learning of dynamical systems from images and videos recorded using phone, high-speed camera or satellites. Further, once the WNO is trained, the predictions stage is extremely efficient as compared to traditional methods; this makes WNO an ideal candidate when it comes to real-time prediction. In this section, we implement the proposed WNO to develop a digital twin that can predict Earth's temperature. In particular, we use WNO for accurate short to medium-range prediction of the hourly and mean monthly 2m temperature of air based

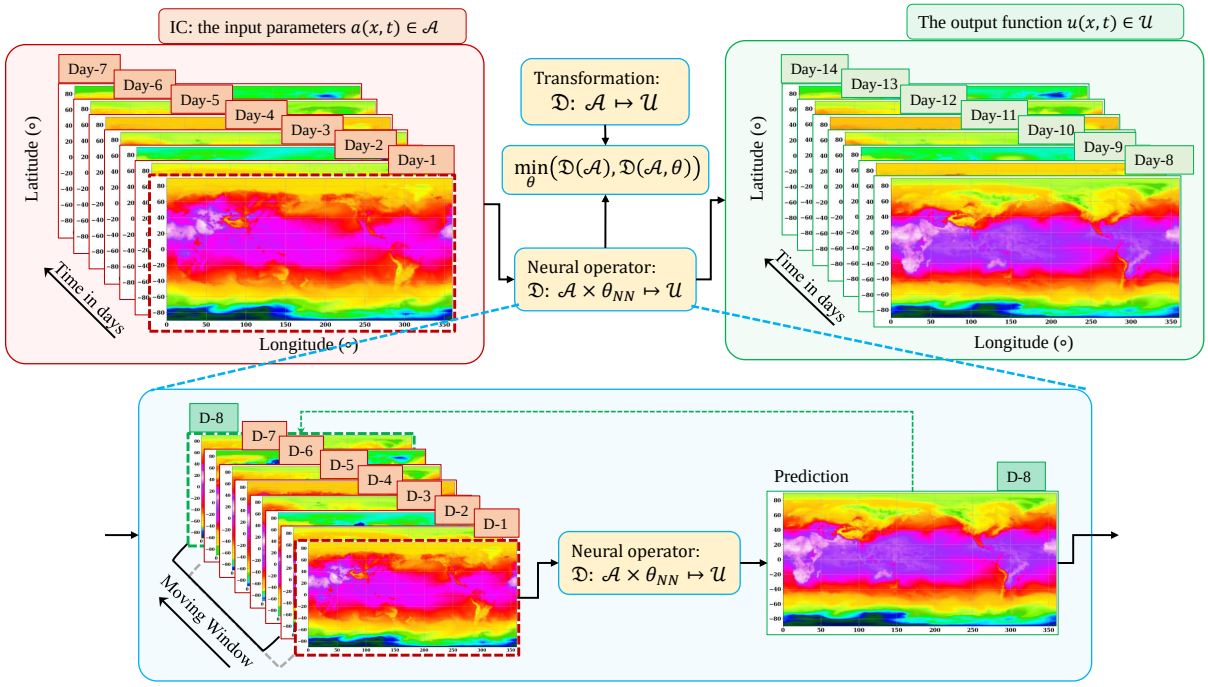

Figure 4: **Methodology for the short-term weekly forecast of 2m air temperature on a resolution of $2^\circ \times 2^\circ$.** The aim of the short-term forecast is to train the WNO using previous 7 days data and then predict the 2m temperature for next 7 days. To do this, a recurrent neural network (RNN) structure is employed within the WNO. The RNN framework takes the previous 7 days data as input and then performs prediction for Day 8. For prediction the parameters are learned through convolution using the proposed WNO architecture. The Day 8 prediction data is then appended in the previous 7 day database. To maintain the recurrent structure i.e., to transfer the information from Day 8 to future predictions the information of Day 1 from previous data is discarded and the process is continued until the next 7 day predictions are made. This results in a highly efficient and accurate.

on historical data. The datasets are taken from European Centre for Medium-Range Weather Forecasts (ECMWF) which provides publicly available hourly and monthly averaged datasets (so called ERA5) of various parameters of the climate. The ERA5 datasets are the fifth generation ECMWF reanalysis of various parameters of the atmosphere that uses data assimilation techniques to combine the numerical weather prediction data with freshly obtained observations to create new improved predictions of the various states of the atmosphere. The ERA5 database is a huge collection of hourly and monthly measurements for a large number of atmospheric, ocean-wave and land-surface parameters. The hourly and monthly datasets are stored both on single levels (atmospheric, ocean-wave and land surface quantities) and different pressure levels (upper air fields). The resolution of the datasets are maintained as $0.25^\circ \times 0.25^\circ$ for atmosphere, $0.5^\circ \times 0.5^\circ$ for ocean waves and $0.1^\circ \times 0.1^\circ$ for land. While there are many variables present and many combinations are possible for various pressure levels, we only focus on one parameter with a spatial resolution of $2^\circ \times 2^\circ$. Combining all the parameters to develop a complete digital twin for Earth's climate is left as a future research direction.

The framework for short-term prediction of the 2m temperature is provided in Fig. 4. The predictions are done at 12.00 Universal Time Coordinated (UTC) standard. For this purpose, the daily measurement of 2m temperature at 12.00 UTC time standards from Jan 1st 2017 to Dec 31st 2021 are taken from ERA5 database. We first scaled down the data from $0.25^\circ \times 0.25^\circ$ to $2^\circ \times 2^\circ$ and arranged it into batches of 7, where the 7 denotes the number of days in a week. We then feed the dataset to the proposed WNO. Given the observations of the previous week the aim here is to predict the 2m temperatures at 12.00 UTC time standards for next 7 days. For this problem, a recurrent neural network (RNN) structure is formulated within the WNO. The WNO architecture consists of 4 WNO layer where db4 Daubechies wavelet with 2 level of decomposition is performed. A total number of 276 training samples, 6 testing samples and 500 epochs with batch size of 3 are used. The initial learning rate is kept at 0.001 with a step decay constant of 0.75 with step size 50. From previous 7 days database, the WNO performs the prediction for Day 8 and discards the Day 1 information from previous record by appending newly obtained Day 8 information to the previous database. This process is continued until the complete prediction for next 7 days are obtained. The prediction results

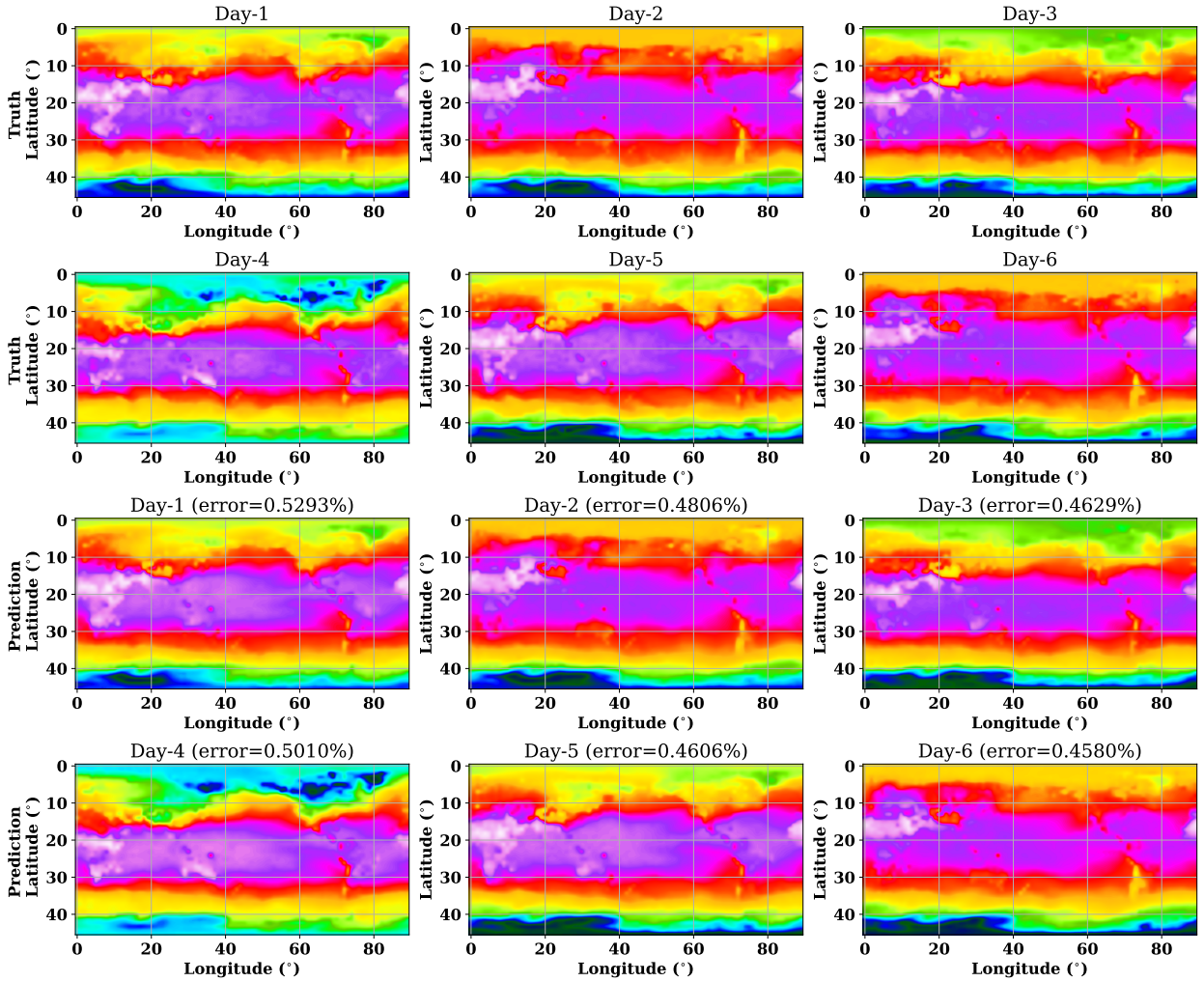

Figure 5: **WNO for forecast of the weekly 2m air temperature on a resolution of $2^\circ \times 2^\circ$.** The predictions along with their corresponding errors for weekly forecast of 2m temperature at 12.00 UTC between 10^{th} to 15^{th} April are shown in the above figure. The errors are obtained with respect to the forecast made by Integrated Forecast System (IFS) of ECMWF. The predictions using proposed WNO are obtained with an error of the magnitude of $< 1\%$ that is a proof of the nearly accurate match with the actual forecast data. Once the training is done the predictions for the entire week are made within 3 seconds on the given GPU, which is evident of the computational advantage of the proposed WNO framework.

are compared with the resolution-reduced results of Numerical Weather Prediction (NWP) system from ECMWF. The prediction results with their corresponding error are presented in Fig. 5. At the accuracy of $2^\circ \times 2^\circ$, the short-term weekly prediction results obtained using the proposed WNO is highly accurate. Close observations on the results reveal that the WNO has almost accurately captured the finest details present in the true results at $2^\circ \times 2^\circ$ resolution. Additionally, once trained, the predictions for entire week took only 3 second; this indicates computationally efficiency of the proposed WNO.

Next, we focus on medium-range prediction of monthly mean 2m temperature at 12.00 UTC time standard. For this purpose, the monthly averaged datasets from 1979 to 2022 at 12.00 UTC standard are taken from ECMWF-ERA5 database. The problem is formulated as a simple 2D problem and the proposed WNO is trained from the previous datasets to perform the predictions for next monthly mean 2m temperature. For training and testing the total dataset is sliced into 480 and 40 samples and a total of 500 epochs with batch size 5 is used. The bd4 Daubechies wavelet with 5 level of decomposition are utilized. The initial learning rate is set as 0.001 which is later decayed with decay constant 0.75 at a step size of 50. The prediction results with corresponding errors are given in Fig. 6.

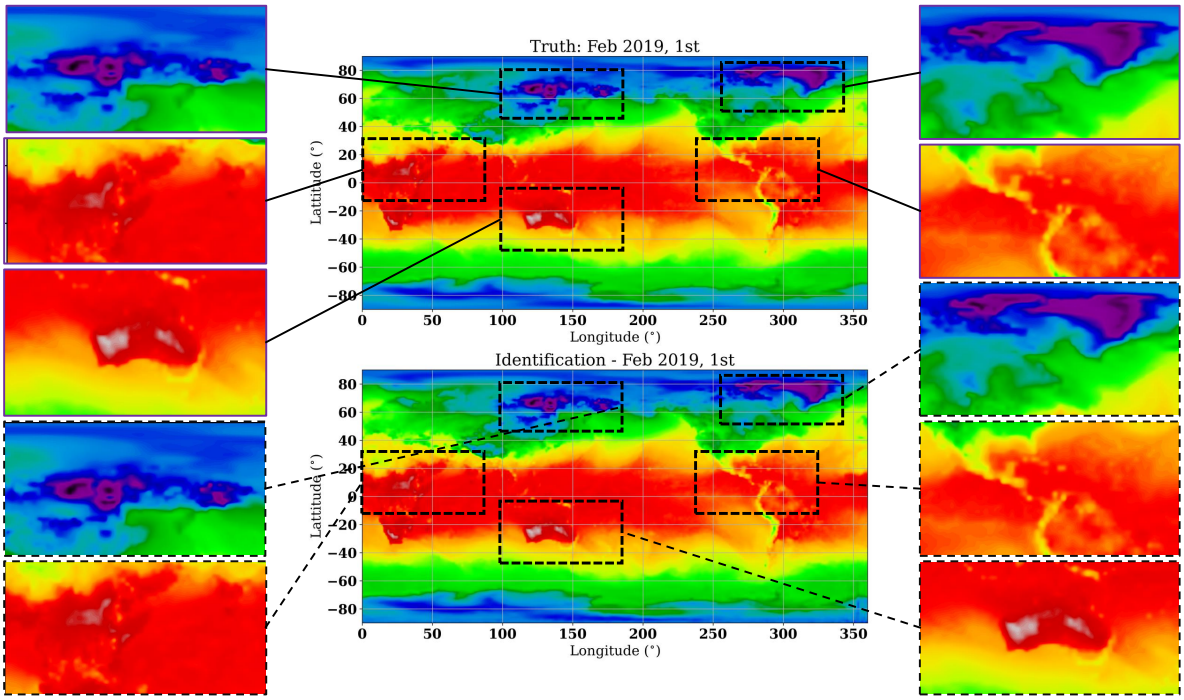

Figure 6: **WNO for forecast of the monthly averaged 2m air temperature on a resolution of $2^\circ \times 2^\circ$.** The predictions for monthly averaged 2m temperature for 1st February of 2019 is performed. The datasets from Numerical Weather Prediction and proposed WNO are shown in the above figure. The predictions shows accurate match with the actual data with error of the magnitude of $\approx 0.1\%$. On close observation it can be observed that, in addition to the actual temperature gradients the proposed WNO has also captured the finer details like white spots. Even in small resolutions the WNO has learned the small-scale features.

From the presented results, it can be conjectured that the proposed WNO can successfully learn the operators of the various parameters of climate and atmosphere, thereby can be used for computationally faster and accurate predictions of arbitrary high-time scale parameters. We agree that the low-resolution based prediction are not enough to accurately capture the finest details of climate dynamics, however, in this example, we only take a small subset of all the possible applications and showcase the possibility for the proposed WNO for weather prediction. Given computational facilities, the WNO can also learn the climate dynamics at desired resolutions that will facilitate the predictions of small-scale features in the climate variables.

Discussion

In this article, we proposed a new neural operator for learning of highly nonlinear operators that arise naturally in day-to-day scientific and engineering problems. Theoretically, the proposed WNO performs a local transformation on the input and then carries out a series of updates on the locally transformed inputs. The series of updates are formulated using a combination of nonlinear activation functions and kernel integral operators. The integral kernels are parameterized by neural network, where the parameterization of the integral kernel is done using a convolution type architecture; however, instead of performing the convolution directly in the physical space, we propose a wavelet kernel integral operator. Within wavelet kernel integral operator, the inputs are first transformed into the spectral domain using the wavelet and thereafter the convolutions are performed. Together with nonlinear activation functions and wavelet kernel integral updates, the proposed neural operator is able to represent any infinite-dimensional function space. While parameterization of integral kernel, it exploits the spatial and frequency localisation property of the wavelets to learn the frequency of change in pixels over the spatial domain. The use of wavelet space for learning of integral kernel thus allows to track the finest patterns in the inputs; this allows the proposed WNO to learn the highly nonlinear operators even in complex geometric and boundary conditions. In terms of applications, we applied the proposed WNO to a variety of time-independent and time-dependent 1D and 2D PDE examples, exhibiting nonlinear spatial and spatio-temporal behaviours in flow, wave and phase-field problems. The results obtained using the proposed WNO is compared with recently developed state-of-the art neural operators like DeepONet, FNO and MWT. The

results suggests that the proposed WNO is able to learn the highly nonlinear differential operators very effectively even when complex geometry like triangular domain and notch was involved. Quantitatively, for the six examples presented, the error in results obtained using WNO varies in the range 0.21% – 1.75% with WNO outperforming the other operator learning approaches in four problems.

The proposed neural operator can learn the mapping between infinite-dimensional function spaces by combining the nonlinear activation functions with wavelet integral layers. We particularly exploits this fact and extends the application of proposed WNO beyond PDEs to satellite images. In a broader sense, the images and videos can also be contemplated as 2D and time dependent 2D problems, describing the motion or dynamics of an object/process. Thus, the proposed WNO can also be implemented on images and videos captured from cameras and satellite. As an possible application, we demonstrated the proposed WNO for prediction of weekly and monthly mean 2m air temperature of the Earth at a resolution of $2^\circ \times 2^\circ$. In the given resolution, the results are highly encouraging, the proposed framework produces highly accurate predictions for weekly and monthly forecast.

It is worthwhile to note that the proposed WNO is highly efficient and hence, is ideally suited for scenarios where real-time predictions are necessary. In this context, two interesting directions of future work includes application of WNO for development of digital twins and WNO based active vibration control. The weather prediction used-case presented in this paper can be regarded as a weak digital twin for Earth's climate. However, the framework in its current form can only predict the Earth's temperature (weekly and mean monthly). An extremely useful extension will be to develop a fully functional digital twin for Earth's climate that can also predict other climate-related parameters such as rainfall precipitation, storms, etc. Such digital twins can assist mankind in its fight against climate change and global warming.

Methods: Wavelet Neural Operator for parametric partial differential equations

Our aim is to perform the kernel integration in Eq. (3) in the neural network framework by parameterizing the kernel $k(a(x,y),x,y)$ with $\phi \in \boldsymbol{\theta}$. Before we construct the framework for parameterization in wavelet space, let us revisit the convolution integral. Let us drop the term $a(x,y)$ in Eq. (3) and modify the kernel as $k_\phi(x-y)$. By doing this we rewrite the Eq. (3) as,

$$\Big(K(\phi) * v_j\Big)(x) := \int_{D\in\mathbb{R}^d} k\left(x-y;\phi\right) v_j(y)\mathrm{d}y; \quad x \in D, \quad j \in [1,l] \tag{6}$$

which is consistent with the convolution operator. In this work, we propose to learn the kernel k_ϕ by parameterizing it in the wavelet domain. By learning kernel in wavelet domain we intend to perform the above convolution on the coefficients of wavelet decomposition rather than direct convolution in physical space. Let $\psi(x) \in \mathcal{L}^2(\mathbb{R})$ be an orthonormal mother wavelet that is localised both in the time and frequency domain. Let, $\mathcal{W}(\Gamma)$ and $\mathcal{W}^{-1}(\Gamma_W)$ be the forward and inverse wavelet transforms of an arbitrary function $\Gamma : D \mapsto \mathbb{R}^{d_v}$. Then the forward and inverse wavelet transforms of the function Γ with the scaling and translational parameters $\alpha \in \mathbb{R}^{+*}$ and $\beta \in \mathbb{R}$ are given by the following integral pairs,

$$(\mathcal{W}\Gamma)_j(\alpha,\beta) = \int_D \Gamma(x)\frac{1}{|\alpha|^{1/2}}\psi\left(\frac{x-\beta}{\alpha}\right)dx, \quad \& \quad (\mathcal{W}^{-1}\Gamma_w)_j(x) = \frac{1}{C_\psi}\int_0^\infty\int_D (\Gamma_w)_j(\alpha,\beta)\frac{1}{|\alpha|^{1/2}}\tilde{\psi}\left(\frac{x-\beta}{\alpha}\right)d\beta\frac{d\alpha}{\alpha^2}, \tag{7}$$

where $(\Gamma_w)_j(\alpha,\beta) = (\mathcal{W}\Gamma)_j(\alpha,\beta)\,\psi((x-\beta)\alpha) \in \mathcal{L}^2(\mathbb{R})$ is a scaled and translation of mother wavelet, often called as the daughter wavelet. By scaling and shifting, the desired wavelets can be obtained from the mother wavelet. The term C_ψ is called as the admissible constant whose range is given as $0 < C_\psi < \infty$. The expression for C_ψ is given as,

$$C_\psi = 2\pi\int_D \frac{|\psi(\omega)|^2}{|\omega|}d\omega, \tag{8}$$

where $\psi(\omega)$ denotes the Fourier transform of $\psi(x)$. Applying the convolution theorem, we therefore obtain that,

$$\Big(K(a;\phi) * v_j\Big)(x) = \mathcal{W}^{-1}\Big(\mathcal{W}(k_\phi)\cdot\mathcal{W}(v_j)\Big)(x); \quad x \in D. \tag{9}$$

Since we propose to parameterize the kernel k_ϕ directly in wavelet space instead of performing wavelet transform of k_ϕ, we represent $R_\phi = \mathcal{W}(k_\phi)$, where R_ϕ represents the wavelet transform of the kernel function $k : D \mapsto \mathbb{R}^{d_v\times d_v}$. With the above information, we define the wavelet neural operator as,

$$\left(\mathcal{K}(\phi) * v_j\right)(x) = \mathcal{W}^{-1}\left(R_\phi \cdot \mathcal{W}(v_j)\right)(x); \quad x \in D. \tag{10}$$

One issue with continuous wavelet transform (CWT) is that it estimates the wavelet coefficients at all possible scales i.e., they contain information along an infinite number of wavelets, which may be redundant and computationally expensive. However, the same equivalent accuracy with computationally speedup can be achieved by performing the discrete wavelet transform (DWT). In DWT, we modify the mother wavelet to calculate the wavelet coefficients at scales with powers of two. In this case, the wavelet $\phi(x)$ is defined as,

$$\psi_{m,\tau}(x) = \frac{1}{\sqrt{2^m}}\psi\left(\frac{x - \tau 2^m}{2^m}\right) \tag{11}$$

where the parameter $m \in \mathbb{Z}^+$ and $\tau \in \mathbb{Z}^{+*}$ are the scaling and translational parameters. The admissible constant in DWT is given by,

$$C_\psi = \int_0^\infty \frac{|\psi(\omega)|^2}{|\omega|} d\omega, \tag{12}$$

The forward DWT is given as,

$$(\mathcal{W}\Gamma)_j(m,\tau) = \frac{1}{\sqrt{2^m}} \int_D \Gamma(x)\psi\left(\frac{x - \tau 2^m}{2^m}\right) dx. \tag{13}$$

By fixing the scale parameter m at a particular integer and shifting the τ, only the DWT coefficients at level m can be obtained. Though there are many different flavours of DWT, we particularly chose the filterbank implantations of DWT. The filterbank implementation decomposes the signal into detail and approximation coefficients by passing the signal through a low-pass and a high-pass filter. If $r(n)$ and $s(n)$ denote the low-pass and high-pass filter, respectively, then two convolutions of the form $z_{low}(n) = (x * r)(n) \downarrow 2$ and $z_{high}(n) = (z * s)(n) \downarrow 2$ are executed, where n is the number of discretization points. While the detail coefficients $z_{high}(n)$ are retained, the approximation coefficients are recursively filtered by passing it through the low-pass and high-pass filters until the total number of decomposition levels are exhausted. At each level, the length of the signal is halved by a factor 2 due to the conjugate symmetry. At the end of entire operation, the length of the support width of wavelet coefficients is given by the formula: $\zeta = n_s/2^m + (2n_w - 2)$, where, n_s is the length of the signal, m is the decomposition level in DWT and n_w is the length of the vanishing moment of the undertaken wavelet.

In the DWT, the coefficients in smaller scales such as 2^0 and 2^1 are usually associated with high frequencies; thus, they are not the canonical choice for kernel parameterization. On the other hand, the high scale wavelet coefficients generally corresponds to low frequency information thus containing the most important features of the input signal. We thus, choose to parameterize R_ϕ on higher scales of discrete wavelet transform, possibly the last level. For this purpose, a finite-dimensional parameterization space is obtained by keeping the information from only the highest scale wavelet coefficients. For $n \in D$, we have $v_t \in \mathbb{R}^{n \times d_v}$, $\mathcal{W}(v_t) \in \mathbb{R}^{n \times d_v}$ and $R_\phi(m) \in \mathbb{R}^{d_v \times d_v}$. Since we want to parameterize R_ϕ only on the highest level of decomposition, we have $\mathcal{W}(v_t) \in \mathbb{R}^{n/2^m \times d_v}$ in a m-level of DWT. In general, the length of wavelet coefficients are also influenced by the number of vanishing moments of the orthogonal mother wavelet. Thus, it is more appropriate to write that $\mathcal{W}(v_t) \in \mathbb{R}^{\zeta \times d_v}$. The direct parameterization of R_ϕ in the wavelet space is therefore carried out by integrating out the parameter ϕ as convolution of $(\zeta \times d_v \times d_v)$-valued tensor. Rephrasing Eq. (10), we can write the convolution of weight tensor $R \in \mathbb{R}^{\zeta \times d_v \times d_v}$ and $\mathcal{W}(v_t) \in \mathbb{R}^{\zeta \times d_v}$ as,

$$\left(R \cdot \mathcal{W}(v_j)\right)_{I_1, I_2}(x) = \sum_{I_3=1}^{d_v} R_{I_1, I_2, I_3} \mathcal{W}(v_j)_{I_1, I_3}; \quad I_1 \in [1, \zeta], \quad I_2, I_3 \in d_v. \tag{14}$$

Relation with Fourier Neural Operator (FNO)

The proposed WNO is closely related to FNO; infact, we will show that the FNO is a special case of the proposed WNO. To understand the relation, let us consider the generalised wavelet transform that is given as,

$$\mathcal{W}(\alpha, \beta) = \int_D \Gamma(x)\psi_{\alpha,\beta}(x) dx, \tag{15}$$

where $\psi_{\alpha,\beta}(x) \in \mathcal{L}^2(\mathbb{R})$ is the orthonormal wavelet and α, β, $\Gamma(x)$ denote the same as above. In wavelet transform, the aim is to transforms the continuous function $\Gamma(x)$ into a continuous function of scale and translational variables α and

β. The transformation is generally done by projecting $\Gamma(x)$ into different wavelet coefficients using the following form of $\psi_{\alpha,\beta}(x)$,

$$\psi_{\alpha,\beta}(x) = \frac{1}{\sqrt{\alpha}}\psi\left(\frac{x-\beta}{\alpha}\right) \tag{16}$$

where $\psi_{\alpha,\beta}(x)$ will decide the nature of waveforms. Based on the requirements, if one chose to project the function $\Gamma(x)$ in terms of its frequency components, the translation parameter can be dropped and the orthonormal basis wavelet can be replaced using the triangular/harmonic basis functions. If one chose to consider only the sine and cosine waves as the orthonormal wavelets i.e., to replace the orthonormal wavelet $\psi_{\alpha,\beta}(x)$ by $e^{-2i\pi\langle x,\omega\rangle}$, Eq. (15) will result in a standard Fourier transform. If we denote $\mathcal{F}(\omega)$ as the Fourier transform and $\mathcal{F}^{-1}(x)$ to be its inverse operation, then for the function $\Gamma(x)$, we have,

$$(\mathcal{F}\Gamma)_j(\omega) = \int_D f_j(x)e^{-2i\pi\langle x,\omega\rangle}dx, \quad \& \qquad \left(\mathcal{F}^{-1}\Gamma\right)_j(x) = \int_D f_j(\omega)e^{2i\pi\langle x,\omega\rangle}\mathrm{d}\omega; \quad j = 1,\ldots,d_v \tag{17}$$

Since the Fourier transform can translate between the convolution of multiple functions, Eq. (9) can be written in Fourier space as,

$$\left(\omega(a;\phi)v_j\right)(x) = \mathcal{F}^{-1}\left(\mathcal{F}(k_\phi)\cdot\mathcal{F}(v_j)\right)(x); \quad x \in D. \tag{18}$$

The kernel k_ϕ therefore can be directly parameterized in Fourier space by choosing a sufficient number of Fourier modes $\omega_{\max} \in D$.

Note on the complexity of the wavelet integral layer

Without loss of generality, we restrict the complexity analysis to a 1D problem. Recall, n is the number of discretization points in the domain D. In the discrete wavelet decomposing, the input signals are decomposed simultaneously using a low-pass filter $r(n)$ and a high-pass filter $s(n)$. In case of Daubechies wavelets, the filters $r(n)$ and $s(n)$ have a constant length, each of which performs the convolutions $(x * r)(n)$ and $(x * s)(n)$, thereby has the complexity $O(n)$. In the single tree format, the discrete wavelet decomposition uses these two filters for recursive multi-level decomposition of the output of the branch convolved with $r(n)$. This recursive filterbank implementation has the overall complexity of $O(n)$. Decomposition of the input using wavelet with a M level results in a signal of length $n/2^M$. Convolution of decomposed coefficients and weight tensor inside the wavelet integral layer takes $O(n/2^M)$ time. The convolution in the CNN with kernel size takes $O(n)$ time. Therefore, the majority of the computational demand arises from the convolution in wavelet integral. In general, there is a trade between the number of decomposition and the integral kernel size and their relation can be represented as a reciprocal relation, i.e., more the decomposition level less the integral kernel size and vice-versa. However, we observed that the level of decomposition play a significant role in speed-up of the proposed neural operator as compared to the size of integral kernel. For discrete wavelet decomposition we make use of the pytorch-wavelet and pywavelet libraries, more details can be found in Ref. [40, 41].

Acknowledgements

T. Tripura acknowledges the financial support received from the Ministry of Human Resource Development (MHRD), India in form of the Prime Minister's Research Fellows (PMRF) scholarship. S. Chakraborty acknowledges the financial support received from Science and Engineering Research Board (SERB) via grant no. SRG/2021/000467 and seed grant received from IIT Delhi.

Data availability

On acceptance all the used datasets in this study will be made public on GitHub by the corresponding author.

Code availability

On acceptance all the source codes to reproduce the results in this study will be made available to public on GitHub by the corresponding author.

References

[1] Michael Renardy and Robert C Rogers. *An introduction to partial differential equations*, volume 13. Springer Science & Business Media, 2006.

[2] Arnold Sommerfeld. *Partial differential equations in physics*. Academic press, 1949.

[3] Douglas Samuel Jones, Michael Plank, and Brian D Sleeman. *Differential equations and mathematical biology*. Chapman and Hall/CRC, 2009.

[4] Thomas JR Hughes. *The finite element method: linear static and dynamic finite element analysis*. Courier Corporation, 2012.

[5] John C Strikwerda. *Finite difference schemes and partial differential equations*. SIAM, 2004.

[6] Robert Eymard, Thierry Gallouët, and Raphaèle Herbin. Finite volume methods. *Handbook of numerical analysis*, 7:713–1018, 2000.

[7] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[8] Maziar Raissi. Deep hidden physics models: Deep learning of nonlinear partial differential equations. *The Journal of Machine Learning Research*, 19(1):932–955, 2018.

[9] Samuel H Rudy, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Data-driven discovery of partial differential equations. *Science Advances*, 3(4):e1602614, 2017.

[10] Kailiang Wu and Dongbin Xiu. Data-driven deep learning of partial differential equations in modal space. *Journal of Computational Physics*, 408:109307, 2020.

[11] N Navaneeth and Souvik Chakraborty. Koopman operator for time-dependent reliability analysis. *arXiv e-prints*, pages arXiv–2203, 2022.

[12] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[13] Somdatta Goswami, Cosmin Anitescu, Souvik Chakraborty, and Timon Rabczuk. Transfer learning enhanced physics informed neural network for phase-field modeling of fracture. *Theoretical and Applied Fracture Mechanics*, 106:102447, 2020.

[14] Souvik Chakraborty. Transfer learning based multi-fidelity physics informed deep neural network. *Journal of Computational Physics*, 426:109942, 2021.

[15] Tianping Chen and Hong Chen. Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE Transactions on Neural Networks*, 6(4):911–917, 1995.

[16] Lu Lu, Pengzhan Jin, and George Em Karniadakis. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv:1910.03193*, 2019.

[17] Lu Lu, Xuhui Meng, Shengze Cai, Zhiping Mao, Somdatta Goswami, Zhongqiang Zhang, and George Em Karniadakis. A comprehensive and fair comparison of two neural operators (with practical extensions) based on fair data. *Computer Methods in Applied Mechanics and Engineering*, 393:114778, 2022.

[18] Shailesh Garg, Harshit Gupta, and Souvik Chakraborty. Assessment of deeponet for reliability analysis of stochastic nonlinear dynamical systems. *arXiv preprint arXiv:2201.13145*, 2022.

[19] Chensen Lin, Martin Maxey, Zhen Li, and George Em Karniadakis. A seamless multiscale operator neural network for inferring bubble dynamics. *Journal of Fluid Mechanics*, 929, 2021.

[20] Somdatta Goswami, Minglang Yin, Yue Yu, and George Em Karniadakis. A physics-informed variational deeponet for predicting crack path in quasi-brittle materials. *Computer Methods in Applied Mechanics and Engineering*, 391:114587, 2022.

[21] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Graph kernel network for partial differential equations, 2020.

[22] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations, 2020.

[23] George Bachman, Lawrence Narici, and Edward Beckenstein. *Fourier and wavelet analysis*, volume 586. Springer, 2000.

[24] Albert Boggess and Francis J Narcowich. *A first course in wavelets with Fourier analysis*. John Wiley & Sons, 2015.

[25] Karlton Wirsing. Time frequency analysis of wavelet and fourier transform. In *Wavelet Theory*. IntechOpen London, UK, 2020.

[26] Stephane Mallat and Wen Liang Hwang. Singularity detection and processing with wavelets. *IEEE transactions on information theory*, 38(2):617–643, 1992.

[27] Seong-Won Cho. Method of recognizing human iris using daubechies wavelet transform, October 17 2002. US Patent App. 09/946,714.

[28] Ehsan Sheybani. Dimensionality reduction and noise removal in wireless sensor networks. In *2011 4th IFIP International Conference on New Technologies, Mobility and Security*, pages 1–5. IEEE, 2011.

[29] Eladio Martin. Novel method for stride length estimation with body area network accelerometers. In *2011 IEEE Topical Conference on Biomedical Wireless Technologies, Networks, and Sensing Systems*, pages 79–82. IEEE, 2011.

[30] Jie Liu. Shannon wavelet spectrum analysis on truncated vibration signals for machine incipient fault detection. *Measurement Science and Technology*, 23(5):055604, 2012.

[31] Jun Zhang, Gilbert G Walter, Yubo Miao, and Wan Ngai Wayne Lee. Wavelet neural networks for function learning. *IEEE transactions on Signal Processing*, 43(6):1485–1497, 1995.

[32] Bingbing Xu, Huawei Shen, Qi Cao, Yunqi Qiu, and Xueqi Cheng. Graph wavelet neural network. *arXiv preprint arXiv:1904.07785*, 2019.

[33] Navid Shervani-Tabar and Nicholas Zabaras. Physics-constrained predictive molecular latent space discovery with graph scattering variational autoencoder, 2020.

[34] Gaurav Gupta, Xiongye Xiao, and Paul Bogdan. Multiwavelet-based operator learning for differential equations. *Advances in Neural Information Processing Systems*, 34, 2021.

[35] Ivan W Selesnick, Richard G Baraniuk, and Nick C Kingsbury. The dual-tree complex wavelet transform. *IEEE signal processing magazine*, 22(6):123–151, 2005.

[36] Lee A Ray and Reza R Adhami. Dual tree discrete wavelet transform with application to image fusion. In *2006 Proceeding of the Thirty-Eighth Southeastern Symposium on System Theory*, pages 430–433. IEEE, 2006.

[37] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[38] Ingrid Daubechies. *Ten lectures on wavelets*. SIAM, 1992.

[39] Yves Meyer. Wavelets: algorithms & applications. *Philadelphia: SIAM (Society for Industrial and Applied Mathematics*, 1993.

[40] Gregory Lee, Ralf Gommers, Filip Waselewski, Kai Wohlfahrt, and Aaron O'Leary. Pywavelets: A python package for wavelet analysis. *Journal of Open Source Software*, 4(36):1237, 2019.

[41] Fergal Cotter. *Uses of Complex Wavelets in Deep Convolutional Neural Networks*. PhD thesis, University of Cambridge, 2020.