scientific reports

Check for updates

OPEN

Deep neural operator-driven real-time inference to enable digital twin solutions for nuclear energy systems

Kazuma Kobayashi[1] & Syed Bahauddin Alam[1,2]✉

This paper focuses on the feasibility of deep neural operator network (DeepONet) as a robust surrogate modeling method within the context of digital twin (DT) enabling technology for nuclear energy systems. Machine learning (ML)-based prediction algorithms that need extensive retraining for new reactor operational conditions may prohibit real-time inference for DT across varying scenarios. In this study, DeepONet is trained with possible operational conditions and that relaxes the requirement of continuous retraining - making it suitable for online and real-time prediction components for DT. Through benchmarking and evaluation, DeepONet exhibits remarkable prediction accuracy and speed, outperforming traditional ML methods, making it a suitable algorithm for real-time DT inference in solving a challenging particle transport problem. DeepONet also exhibits generalizability and computational efficiency as an efficient surrogate tool for DT component. However, the application of DeepONet reveals challenges related to optimal sensor placement and model evaluation, critical aspects of real-world DT implementation. Addressing these challenges will further enhance the method's practicality and reliability. Overall, this study marks an important step towards harnessing the power of DeepONet surrogate modeling for real-time inference capability within the context of DT enabling technology for nuclear systems.

A reliable and sustainable energy is essential to support and drive economic activity in the modern world. As the urgent need for carbon-neutral solutions becomes increasingly evident, nuclear energy is projected to assume a significant role. With the ongoing global increase in energy demands, nuclear power is poised to emerge as a critical and environmentally friendly alternative to conventional energy sources.

Furthermore, nuclear research actively explores novel technologies and approaches to advance the field. Among these exciting research areas, using digital twin (DT) technology in nuclear systems has gained substantial attention. The United States Nuclear Regulatory Commission (U.S. NRC) and Department of Energy (DOE) recognize the potential of DT technology and has identified it as a key area for future research[1,2–5]. The NRC/DOE highlighted several potential benefits of DTs in nuclear energy applications, including increased operational efficiencies, enhanced safety and reliability, reduced errors, faster information sharing, and improved predictive capabilities[1,6]. However, it is imperative to recognize that the evolution of DT technology within the realm of nuclear systems is still at its inception, bringing forth a range of intricate challenges that necessitate diligent resolution and strategic overcoming[7]. These challenges span multifarious domains, encapsulating crucial aspects like the integrating data from various sources, the modeling & and simulation (M &S) of complex nuclear systems, the synchronization in real-time between the digital replica and physical asset, and the critical domains of cybersecurity and safeguarding data privacy. Furthermore, developing and deploying advanced sensors and network architecture is paramount, as they are pivotal to ensuring an uninterrupted data flow[1,7]. Among these challenges, this study focuses on potentially utilizing deep learning methods in M &S for nuclear systems.

A digital replica is created by capturing and integrating various data sources, such as sensor data, operational data, and design specifications, to generate a highly detailed and accurate representation of the physical system[8]. This virtual replica serves multiple purposes: monitoring, analysis, simulation, and prediction. While traditional analysis and simulation codes can fulfill these roles, it can be challenging to balance accuracy and speed, especially in systems requiring rapid analysis and response prediction. Dedicated codes are employed for

[1]Nuclear, Plasma & Radiological Engineering, University of Illinois Urbana-Champaign, Suite 100 Talbot Laboratory, 104 South Wright Street, Urbana, IL 61801, USA. [2]National Center for Supercomputing Application, 205 W Clark Street, Urbana, IL 61801, USA. ✉email: alams@illinois.edu

analysis in certain domains like nuclear reactor systems[9,10]. However, these codes often prioritize high analysis accuracy, which can result in high computational costs. Relying solely on expensive simulations based on real-time information from various sensors installed in the reactor system and then making control decisions based on the results can lead to sluggish operator response times and, in extreme cases, potentially catastrophic nuclear accidents. This situation calls for a new approach in DT technology for nuclear and energy systems[11–13] that simultaneously addresses the need for high calculation accuracy and speed.

Deep learning, particularly through neural networks (NNs), has transformed the landscape of modeling techniques for nonlinear systems. Its significant contribution lies in its remarkable ability to capture intricate and complex relationships within data, making it exceptionally well-suited for approximating nonlinear phenomena. Within nonlinear systems, NNs excel at approximating functions that map inputs to outputs without imposing explicit assumptions about underlying dynamics. It is particularly advantageous when traditional modeling approaches falter due to the system's nonlinear, dynamic, or stochastic nature. The applications of NNs within the nuclear field further highlight their versatility. Instances include leveraging Deep Neural Networks (DNNs)[14] and Recurrent Neural Networks (RNNs)[15] for predicting neutron flux distribution in reactor cores. Additionally, NNs have been employed for surrogate modeling of nuclear reactor dynamic equations. It has been facilitated through the use of Convolutional Neural Networks (CNNs)[16] and Physics-Informed Neural Networks (PINNs)[17].

The advancement of NNs has been instrumental in shaping the landscape of data-driven modeling techniques, including hybrid approaches like PINNs. However, as these deep learning models progress to the deployment phase, they encounter a formidable challenge termed "dataset shift." This challenge is rooted in the dynamic nature of real-world data. While these models are trained on specific datasets, they might only partially encompass the diverse scenarios they will face during practical applications. Environmental changes, shifts in user behavior, or other external factors can introduce variations in data distribution that the model has yet to encounter during its training. This discrepancy between the distributions of training and deployment data can lead to a decline in model performance, resulting in less reliable predictions and potentially compromising the system's overall functionality. The model might need help to adapt to novel situations insufficiently represented in its training data.

To illustrate, consider the behavior of neutrons within a nuclear system. This behavior is pivotal in reactor core analysis, shielding design, and criticality assessments. It can be effectively described by the neutron transport equation[18], featuring a source term that encompasses contributions from processes like radioactive decay and fission. This neutron source term is far from constant in practical scenarios, exhibiting varying energies and spatial distributions over time. Just as the neutron source term evolves, the dataset shift phenomenon reflects the evolving nature of real-world data, creating a parallel challenge for deploying neural network models effectively in intricate systems such as nuclear reactors.

In order to address these challenges, this study introduces a potential solution in the form of data-driven surrogate modeling, utilizing the Deep Operator Network (DeepONet)[19] to fulfill these demanding prerequisites effectively as a real-time prediction algorithm component of DT system.Unlike conventional ML methods that usually deal with input-output mappings, DeepONet focuses on the mapping between functions[19]. By training over function spaces, the DeepONet network learns a mapping between inputs and outputs without requiring retraining for various conditions and scenarios. Once trained, DeepONet can evaluate this mapping very quickly for any new input, bypassing the iterative processes typically involved in high-fidelity solvers. The main advantage of this approach is the ability to generalize and predict outcomes for inputs outside the distribution (extrapolation). This is particularly beneficial in dynamic reactor environments where conditions can evolve rapidly, but the underlying physical processes or systems that the DT is modeling remain consistent. Utilizing DeepONet's unique attribute, the study constructs a surrogate model for the intricate spatial distribution of neutron flux. This approach employs the neutron source's spatial distribution as an input function. This utilization of function-based modeling demonstrates the potential of DeepONet in capturing the intricate dynamics of nuclear systems[19] as a faster surrogate model instead of conventional nuclear code.

It is worth addressing that this research constitutes an extension of earlier published investigations on DT-enabling technologies by the lead author[20,21]. Illustrative instances of DeepONet's generalization capability is demonstrated in[20]. Moreover, explainability of the prediction algorithm for DT systems is carried out in[21]. This paper delves deeper into the practical application of the DeepONet in nuclear engineering problems, providing additional insights and discussions. The current work also introduces new experimental data and presents further validations to reinforce the robustness of the proposed approach.

Deep operator network (DeepONet)

DeepONet[19], an advanced neural network, is built upon the Universal Approximation Theorem but focuses on the Universal Approximation Theorem for Operators[19,22]. While traditional neural networks map inputs to a function space, DeepONet is designed to map information from functions to an operator that can be observed in any domain.

Suppose G is an operator that takes an input function u; this assumption makes $G(u)$ the output function corresponding to the input one. As explained in[19], in the domain of $G(u)$, the real number at any sampling point y can be expressed as $G(u)(y)$. To handle an input function in calculations, it must be discretized in the input space V. In the concept of DeepONet[19], input functions are discretized by sampling at the fixed positions $\{x_1, x_2, \ldots, x_m\}$ where m represents the number of discretized points. Therefore, DeepONet can handle the two network inputs; $[u(x_1), u(x_2), \ldots, u(x_m)]^T$ and $y \in \mathbb{R}^d$. Based on the Universal Approximation Theorem for the Operator, the operator G can be expressed by the Generalized Universal Approximation Theorem for the Operator by[19,23] as follows:

$$\left| G(u)(y) - \langle \underbrace{\mathbf{g}(u(x_1, u_2, \cdots, u(x_m))}_{\text{branch}}, \underbrace{\mathbf{f}(y)}_{\text{trunk}} \rangle \right| < \epsilon \tag{1}$$

where $\epsilon > 0$, $\mathbf{g}$ and $\mathbf{f}$ are continuous vector functions $\mathbf{g} : \mathbb{R}^m \longrightarrow \mathbb{R}^p$ and $\mathbf{f} : \mathbb{R}^d \longrightarrow \mathbb{R}^p$, $\langle \cdot, \cdot \rangle$ represents the dot product in $\mathbb{R}^p$. The function $\mathbf{g}$ and $\mathbf{f}$ are replaced with neural networks. Here, the NNs that handle the input function are distinguished as "branch" and those that handle the input vector $y \in \mathbb{R}^d$ as "trunk." Since the main topic of this study is the applied use of DeepONet, please refer to[19,24] for more detailed mathematical proof.

DeepONet employs the branch-trunk architecture, where the "branch" and "trunk" networks are subnetworks implemented using NNs[19,25]. The trunk network deals with domain information, while the branch network encodes sensor information from the function. Internal filtering allows DeepONet to handle data efficiently. This architecture is harnessed to approximate the system's operator, and the entire model is trained using loss associated with the predictions. Figure 1 illustrates the branch-trunk architecture. For a more in-depth mathematical proof, refer to[19,24].

DeepONet is proven to be versatile in diverse domains, including multiphysics scenarios such as electric convection[23], bubble growth dynamics[26], and fluid dynamics[27]. As DeepONet continues to prove its efficacy in handling complex nonlinear and computationally intensive problems, its applicability is anticipated to expand across a wide array of fields in the future.

Experiments

In order to showcase the capabilities of DeepONet, a surrogate model is constructed for calculating the 2-dimensional spatial distribution of neutron flux in a maze. The training and test datasets used for training the DeepONet model are prepared using Particle and Heavy Ion Transport code System (PHITS) version 3.24[28]. This section elaborates on the methodology employed for data generation and the simulation setup utilized in this study.

Particle transport code

PHITS, a Monte Carlo particle transport simulation code, has the capability to accurately simulate particle transport across a wide range of energy spectra through the utilization of sophisticated nuclear reaction models and comprehensive data libraries[28]. Its versatility allows for its application in various research fields, including

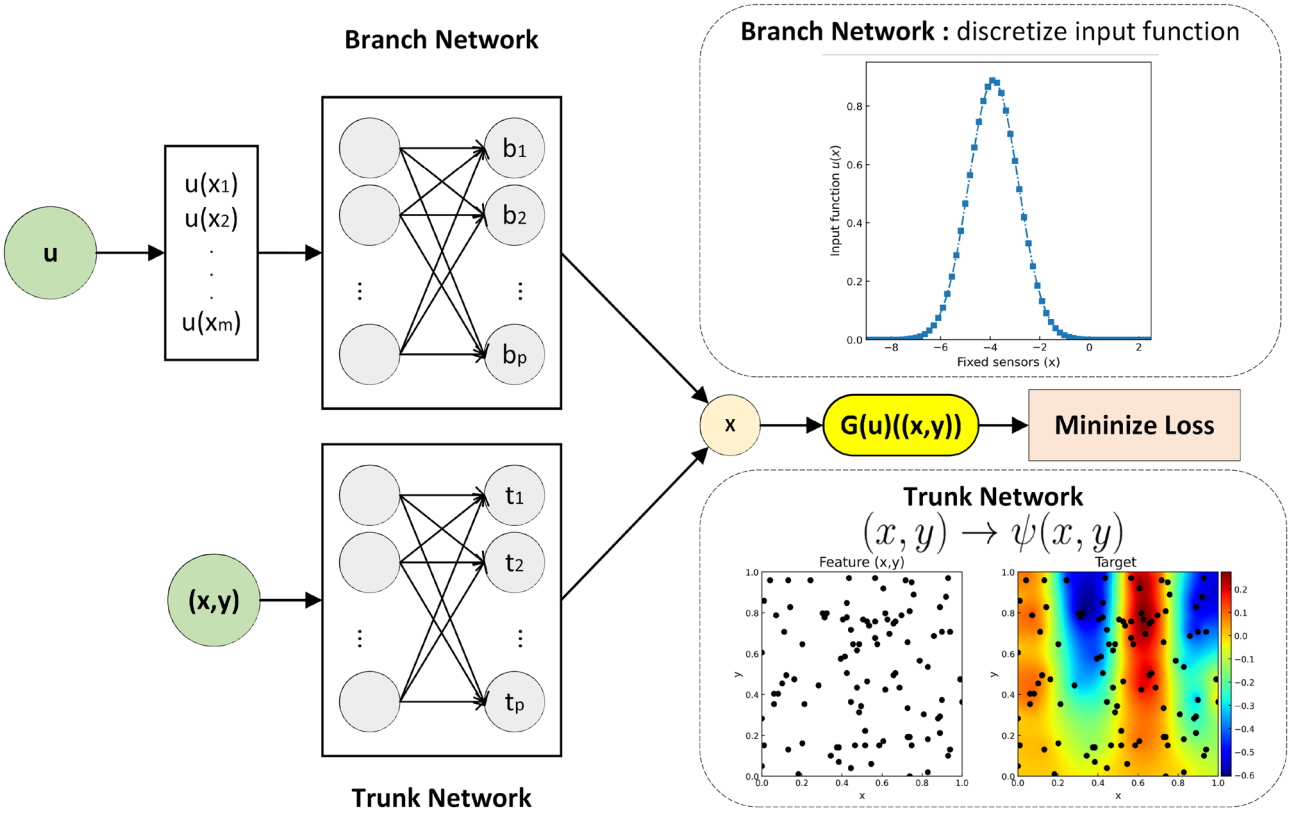

Figure 1. Illustration of DeepONet Branch-Trunk architecture employing fully-connected neural networks. It is an example when the input vector is composed of 2-dimension (i.e., $\mathbf{y} = [x,y]$). The discretized input functions and input vector y are individually fed into fully connected neural networks. Then, the dot product is computed using their network outputs. Note: This figure was completely redrawn by the authors based on the concept of DeepONet presented in[19] to fit the problem setting of this study. This figure is further modified following[20,21].

accelerator technology, radiotherapy, space radiation, and other domains involving particle and heavy ion transport phenomena. In this study, we leverage the power of PHITS to simulate the 2D spatial distribution of neutron flux within a maze. To accomplish this, we have customized and employed the sample code provided by the PHITS Development Group as a foundation for our research. For particle transport calculations in PHITS, the definition of geometry, material, radiation sources, and the phenomenon to be analyzed, referred to as Tally, is necessitated.

Geometry
In our simulation, a hypothetical maze with concrete walls enclosing an air-filled interior is chosen as the geometry, as depicted in Figure 2a. During the configuration of the geometry, there is no need to specify the spatial resolution, such as the grid size, as it is a critical factor affecting the accuracy of the calculation and is determined when defining the Tally, as elucidated later.

Material
The material densities of concrete and air are set to $2.2\,\mathrm{g}/cm^3$ and $1.2 \times 10^{-3}\,\mathrm{g}/cm^3$, respectively. For each defined element, nuclear reaction cross-sections are bundled from the nuclear data library JENDL-4.0[29]. Moreover, the detail of material compositions used in this study is provided in Supplementary Material A.

Neutron Source
The simulation employs a Gaussian distribution neutron source, where the spatial distribution of neutrons is modeled as independent Gaussian distributions along each component of the spatial vector $\mathbf{x} = [x_1, x_2, x_3]$. Given the independence of the variables, the covariance matrix Σ is a diagonal matrix with variances σ_j^2 for each spatial dimension, as illustrated in Eq. (2):

$$\Sigma = \begin{pmatrix} \sigma_1^2 & 0 & 0 \\ 0 & \sigma_2^2 & 0 \\ 0 & 0 & \sigma_3^2 \end{pmatrix}. \tag{2}$$

In this representation, the subscript j corresponds to each spatial direction. Consequently, the 3-dimensional Gaussian distribution $\phi(\mathbf{x})$ is defined by Eq. (3):

$$\phi(\mathbf{x}) = \prod_{j=1}^{3} \frac{1}{\sqrt{2\pi\sigma_j^2}} \exp\left\{ -\frac{1}{2\sigma_j^2} (x_j - \mu_j)^2 \right\}. \tag{3}$$

The neutron source $u(E, \mathbf{x})$ is characterized by a mono-energetic distribution, where all neutrons possess the same energy level E. This leads to the introduction of a new quantity, $\phi(\mathbf{x}) \cdot E$, which represents the energy density distribution of neutrons in space. The neutron source distribution is thus simplified to:

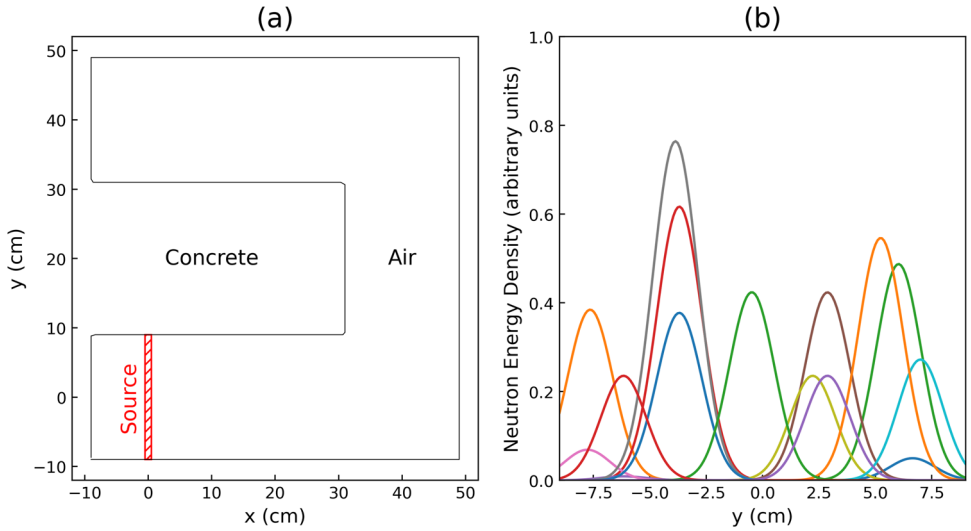

Figure 2. (**a**) the layout of the concrete maze used in the particle transport simulation. The maze is enclosed by concrete walls, and the interior is filled with air. The neutron source position y is randomly selected, while the x location is fixed at $x = 0$. (**b**) example of randomly generated distribution neutron sources (neutron energy density). The peak value and the Gaussian center position in the y-axis are randomly generated in each simulation run.

$$u(E,\mathbf{x}) = \phi(\mathbf{x}) \cdot E = E \cdot \prod_{j=1}^{3} \frac{1}{\sqrt{2\pi\sigma_j^2}} \exp\left\{-\frac{1}{2\sigma_j^2}\left(x_j - \mu_j\right)^2\right\}, \tag{4}$$

In this equation, $\phi(\mathbf{x})$ represents the spatial distribution of neutron intensity and E is the constant energy level of each neutron. The product $\phi(\mathbf{x}) \cdot E$ thus describes the distribution of neutron energy density across the spatial dimensions, providing a comprehensive view of both the spatial and energetic characteristics of the neutron source. In this equation, μ_j represents the mean position component of the mean vector μ. This model allows for the simulation of the spatial and energetic distribution of neutrons, providing insights into their behavior in a controlled environment.

For every simulation, the single neutron energy E and mean position μ_2 are randomly generated within the ranges of E and y are defined by $E \in (0, 1)$ in MeV and $y \in [-9, 9]$ in cm, respectively. These values are then employed in Eq. (4) to prepare the neutron source. It should be noted that the variances $\sigma_1^2 = \sigma_2^2 = 1, \sigma_3^2 = 0$, and mean values $\mu_1 = \mu_3 = 0$ remain fixed throughout the process. The possible regions of the mean position y are indicated by the red boxed area in Fig. 2a. Figure 2b shows an example of 15 randomly generated patterns of neutron sources projected onto the y-plane, which confirms that the peak value and peak position are each a random combination.

Tally

The flux within the geometry is calculated using the [T-Track] tally. To cover potential peak energies up to 1 MeV, the energy mesh range is defined from 1 keV to 10 MeV. For spatial resolution, the geometry is divided into 80 divisions in both the x- and y-directions, ranging from − 12 to 52 cm, while a single division is used in the z-direction, covering a range from − 10 cm to 10 cm. This results in the definition of 6400 sub-regions on the xy-plane, and the neutron flux values are computed for each of these regions. In this calculation, a normalization factor of 1000 is set to avoid the default per-particle normalization in PHITS, which would result in a significantly smaller value.

Executing the simulations

A total of 1900 random combinations of energy E and the mean value of y positions are generated, and simulations are carried out for each combination. Each simulation includes 10^5 neutrons per batch, with a total of 10^2 batches per combination. This setup ensures that the relative error of the neutron flux remains below 10%.

In order to provide a comprehensive comparison between the simulation and surrogate model runtimes, it is important to include information about the computational environment. The simulations are performed on a machine running the Ubuntu 22.04 operating system, equipped with an AMD Ryzen9 3900X CPU (12 Cores/24 Threads) and 64GB DRAM (3,200 MHz). PHITS version 3.24, compiled by our group with the Intel Fortran Compiler (Intel(R) Parallel Studio XE 2020 Update 1 for Linux), is used for hybrid MPI and OpenMP parallel computing. The number of threads (or cores) to be used for OpenMP is set to 2 threads per core, and one for MPI is 11 cores. Based on the log data, the time required per simulation was 30.54 ± 3.76 seconds.

Preprocessing

A systematic multi-stage protocol is implemented for preprocessing the data to trainining and test the DeepONet model.

Initially, an ensemble of n functions $u^1, u^2, \ldots, u^n$ is generated, each representing a hypothetical neutron source distribution scenario within our study. These functions are discretized by sampling them at m uniformly distributed spatial coordinates across the domain of interest. This process is graphically represented in Fig. 3a, providing a visual understanding of the discretization methodology. This study evaluates each function at $m = 190$ points along the y-axis, maintaining a constant interval width of 0.095 cm between points to ensure uniform domain coverage. Following the discretization, these functions are partitioned into training and test datasets in an 8:2 ratio. This data division is designed to optimize the training process while comprehensively evaluating the model's predictive capability on unseen data.

As Fig. 3b represents, the discretized functional data are subsequently used as inputs for particle transport simulations conducted via PHITS on an 80×80 Cartesian grid. This simulation step produces a matrix dataset of 6400 grid point coordinates (x, y), each paired with its corresponding simulated neutron flux values $\psi(x, y)$. This process effectively transforms the abstract functional forms into concrete, quantifiable simulation outcomes.

Furthermore, as demonstrated in Fig. 3c, while the entire set of 6400 (x, y) coordinate pairs with corresponding simulation results $\psi(x, y)$ is available for surrogate model construction, we have methodically created multiple sub-datasets, labeled as Set1 through Set5, to evaluate the impact of data volume on DeepONet's performance. Each dataset is generated by randomly sampling a specific proportion of the total data in increments of 10%, starting from 50% and extending up to 90% of the 6400 pairs. This stratified sampling strategy facilitates a comprehensive evaluation, allowing us to methodically analyze how the variation in the volume of training data affects the model's predictive accuracy. Creating these subsets, Set1 to Set5 enables a systematic investigation into the relationship between the quantity of training data and the fidelity of the DeepONet model.

Finally, the data is meticulously formatted to the unique capabilities of DeepONet, as delineated in Fig. 3d. Unlike conventional neural network models that rely predominantly on fixed-size feature vectors, DeepONet is uniquely equipped to learn mappings between functional inputs and their corresponding outputs across continuous domains. This specialized formatting process involves integrating the evaluations of the discretized input functions with their paired observation coordinates and neutron flux values. Thus, the data is transformed

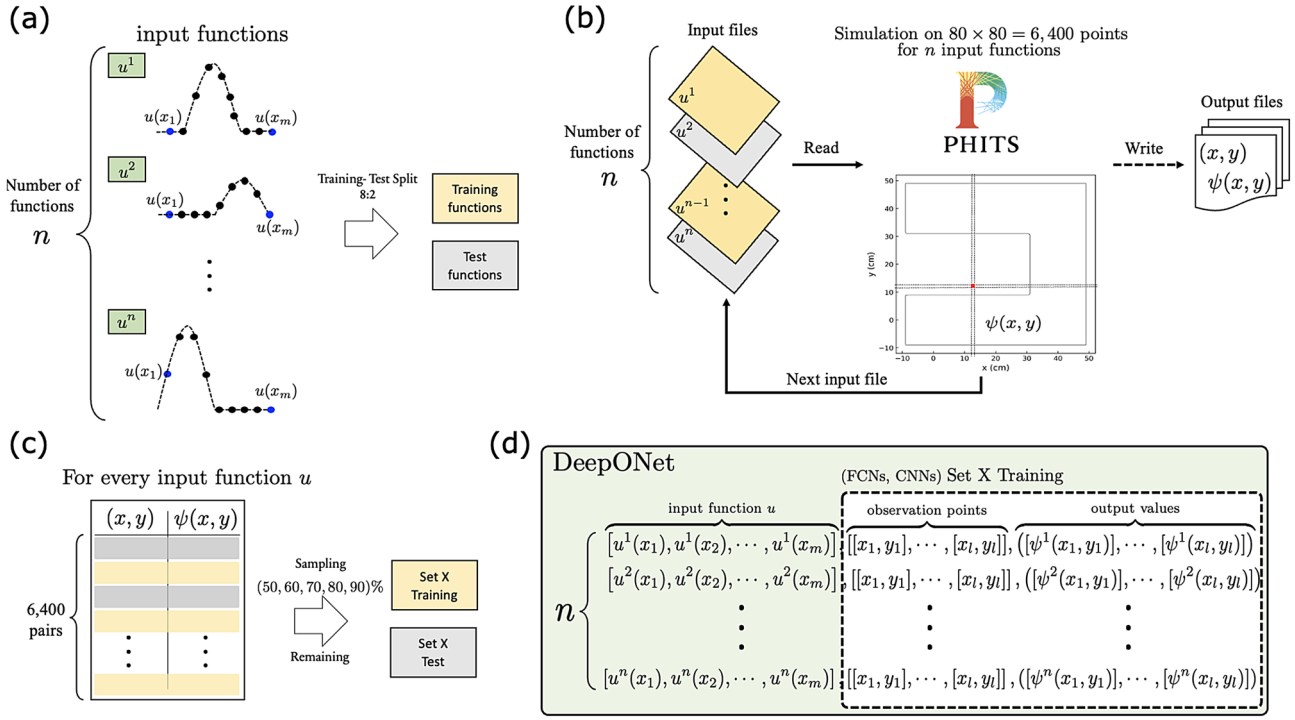

Figure 3. Overview of the data preparation and simulation pipeline for DeepONet and conventional neural network models: (**a**) Input functions u^i are sampled at m points, with the total number of functions being n. These functions are then split into training and test sets using an 8:2 ratio. (**b**) Each input function is read and processed through a PHITS simulation on a grid of 80 x 80 points to generate output files containing values of $\psi(x, y)$ at corresponding coordinates. (**c**) The simulation results for every input function u form a dataset comprising 64,00 (x, y) pairs, which are then sampled into subsets for training and testing purposes. (**d**) The DeepONet framework utilizes the input functions, observation points, and output values for model training, distinguishing it from the training data structure used in FCNs and CNNs, which are represented in a more traditional feature-target format. The subscripts l in the observation points and output values equal to the length of sampled dataset X in panel (**c**).

into a format that optimally aligns with DeepONet's architecture, enabling it to learn from these functional relationships efficiently.

DeepONet model

The desired function of a DeepONet model is to obtain the operator that maps between the distribution of neutron source and the 2-dimensional spatial distribution of neutron flux in the maze. In this case, the operator G can be represented as $G : u(E, \mathbf{y}) \mapsto \psi(\mathbf{y})$, where $\mathbf{y} \in \mathbb{R}^2$ denotes the position vector in the x-y plane. DeepONet network architecture, model training, and evaluation methods are provided in this section to build the model. The implementation of DeepONet is done using the scientific machine learning library DeepXDE[24].

DeepONet utilizes fully connected neural networks for both the branch and trunk networks. The branch network has a layer size of [190, 80, 80], while the trunk network has a layer size of [2, 80, 80] to accommodate the two-dimensional input (x, y).

The Adam optimization algorithm is employed as the optimization method during the training process. The mean L^2 relative error is the evaluation metric to assess the model's performance. The model is trained for 10,000 iterations with a learning rate of 0.001. The learning rate determines the step size during gradient descent and influences the convergence speed and accuracy of the training process. These choices of optimization algorithm, evaluation metric, and learning rate are consistent across all training datasets, ensuring a fair and comparable evaluation of the models.

In order to evaluate the performance of DeepONet models trained on each training dataset, four indices are employed: the R^2 score, root-mean-squared error (RMSE), mean absolute error (MAE), and the ratio of RMSE to MAE (RMSE/MAE). These indices provide insights into the trained models' accuracy, precision, and general performance. Please refer to Supplementary Material D.

Results and discussions

Overall performance of the DeepONet models

The evaluation of DeepONet models using different datasets demonstrated their performance on the test dataset in terms of R^2 score, RMSE, MAE, and RMSE/MAE, with Table 1 presenting the mean and standard deviation of each metric. The results revealed that larger training datasets led to improved model performance. Even with

Dataset	Metrics			
	R^2 ($\times 10^{-1}$)	RMSE ($\times 10^{-2}$)	MAE ($\times 10^{-2}$)	RMSE/MAE
Set1 (50%)	9.93 ± 0.02	6.61 ± 1.49	4.27 ± 1.20	1.56 ± 0.13
Set2 (60%)	9.93 ± 0.02	6.57 ± 1.47	4.47 ± 1.22	1.49 ± 0.10
Set3 (70%)	9.93 ± 0.02	6.43 ± 1.35	4.17 ± 1.15	1.56 ± 0.12
Set4 (80%)	9.95 ± 0.02	5.71 ± 1.40	3.84 ± 1.20	1.51 ± 0.12
Set5 (90%)	9.96 ± 0.02	5.14 ± 1.33	3.45 ± 1.16	1.52 ± 0.12

Table 1. Performance of DeepONet models evaluated with R^2, RMSE, MAE, and RMSE/MAE.

the smallest dataset, the DeepONet models achieved impressive results, with R^2 values reaching 0.99 and RMSE and MAE staying within 10%. For more detailed metrics for all model test data, please refer to Supplementary Material B.

Ideally, a well-constructed model should capture the general trends in the data, with only random noise following a normal distribution being represented as errors. In such cases, the ratio of RMSE to MAE should be close to 1.253. However, the observed ratio of approximately 1.53 in all models indicates the presence of data points significantly deviating from the model predictions, suggesting the influence of outliers.

To address this issue, two potential solutions are considered. Firstly, the removal of outliers from the training dataset can improve model generalization and performance by focusing on representative data. Secondly, hyperparameter tuning allows the DeepONet model to adapt more effectively to the dataset's complexities, fine-tuning its performance.

Combining both approaches can lead to a more robust and accurate DeepONet model, enabling reliable predictions even in the presence of challenging data points. Implementing these improvements will enhance the model's effectiveness and applicability, making it a valuable tool in the context of this study.

Comparisons with FCN and CNN

This section compares the model's performance trained with Set1 on the test data against FCN and CNN. As mentioned earlier, DeepONet takes functions as inputs, and the model test evaluates its response to unseen input functions. In this study, 380 test input functions were provided to the model, and the obtained model outputs were compared to the true values. The same metrics used in the previous section were calculated for each input function.

To distinguish each input function, identification numbers (Test ID) were assigned from 1 to 380. Notably, when Test ID 23 and 18 were used, the DeepONet model demonstrated the highest and lowest R^2 values, respectively. To compare with conventional ML methods, FCNs and CNNs were trained on these two cases, and the metrics were computed similarly. For more detailed information on the network architectures of FCNs and CNNs in these cases and the parameters used during training, please refer to Supplementary Material C.

The calculation results are summarized in Table 2, where DeepONet has been trained for a range of conditions, and FCNs and CNNs require retraining once operational conditions change with new dataset. In both cases of Test IDs 23 and 18, the DeepONet model outperformed FCN and CNN in terms of R^2 score, RMSE, and MAE. Particularly, for Test ID 23, the DeepONet model exhibited significantly better performance with RMSE and MAE values superior by order of magnitude compared to the other models. Additionally, in the challenging case of Test ID 18, where FCN and CNN struggled to build accurate models, DeepONet achieved an impressive R^2 value exceeding 0.9, while its RMSE and MAE demonstrated several times better performance than the other models. However, it is noteworthy that the ratio of RMSE to MAE for DeepONet was relatively higher than that of FCN and CNN when comparing the models.

In addition to quantitative model performance considerations, confirming the reproducibility of physical phenomena is crucial. In this study, we modeled the spatial distribution of neutron flux using the neutron source distribution in a two-dimensional space as an input function ($G : u(E, \mathbf{y}) \mapsto \psi(\mathbf{y})$). Since the system under

Test ID	Model	Metrics			
		R^2 ($\times 10^{-1}$)	RMSE ($\times 10^{-2}$)	MAE ($\times 10^{-2}$)	RMSE/MAE
Highest (ID: 23)	DeepONet	9.97	4.77	3.06	1.56
	FCN	9.01	25.41	20.31	1.25
	CNN	9.10	24.68	19.00	1.30
Lowest (ID: 18)	DeepONet	9.79	17.42	12.95	1.35
	FCN	5.07	86.16	73.26	1.18
	CNN	5.10	85.92	73.47	1.17

Table 2. Performances of DeepONet model (Set1), FCN, and CNN.

consideration lacks fissile material, the Gaussian center position of the neutron source distribution is expected to have the highest value. Based on this fact, we compared the predictions of DeepONet, FCN, and CNN models.

Figure 4 presents the ground truth from the simulation, along with the predictions of DeepONet, FCN, and CNN for Test ID 23. As expected, the highest values are obtained at the Gaussian center position of the neutron source (Fig. 4a). The DeepONet model can reproduce the overall distribution of neutron flux, although the peak value at the Gaussian center position is estimated to be small (Fig. 4b). In contrast, FCN and CNN fail to accurately reproduce both the Gaussian center location and the overall neutron distribution prediction (Fig. 4c and d).

Figure 5 shows the ground truth from the simulation and the predictions of DeepONet, FCN, and CNN for Test ID 18. The DeepONet model is capable of reproducing the overall distribution of neutron flux, although the Gaussian center position is slightly blurred (Fig. 5b). On the other hand, FCN and CNN models do not accurately reproduce the neutron flux distribution, especially in the upper-left regions farthest from the neutron source (Fig. 5c and d). Notably, in this test case, the neutron flux is overestimated relative to the true value, which could lead to potentially hazardous situations from a radiation protection perspective if underestimated.

Overall, these findings underscore the ability of the DeepONet model better to capture the intricate spatial distribution of neutron flux compared to FCN and CNN models, showcasing its potential for reliable and accurate predictions in nuclear engineering applications.

Key discussions

The study demonstrates the power of DeepONet, which takes functions as input data and constructs operator G in the system using training data. Notably, the prediction accuracy of DeepONet matches or surpasses that of conventional ML methods like FCN and CNN. The use of fixed sensors to extract features from input functions and their integration into the model through a branch network is a compelling concept. Training the model with historical data or high-confidence simulations allows it to handle diverse scenarios, including various accidents in the process.

An essential advantage of DeepONet, shared by many ML-based surrogate models, is its remarkably fast execution speed compared to conventional simulations. While a PHITS simulation took about 30 seconds, DeepONet performed the task in just 0.02 seconds. This remarkable speed makes DeepONet a potent modeling method for digital twin systems, enabling real-time predictions based on data from sensors installed on physical assets. Furthermore, DeepONet stands out in their approach to learning and adapting to dynamic reactor systems without the need for frequent retraining. They achieve this by treating functions, representing system changes, as inputs. This method allows the model to predict system responses across a wide range of scenarios based on a comprehensive initial training. By effectively learning from a vast array of potential situations, DeepONet can

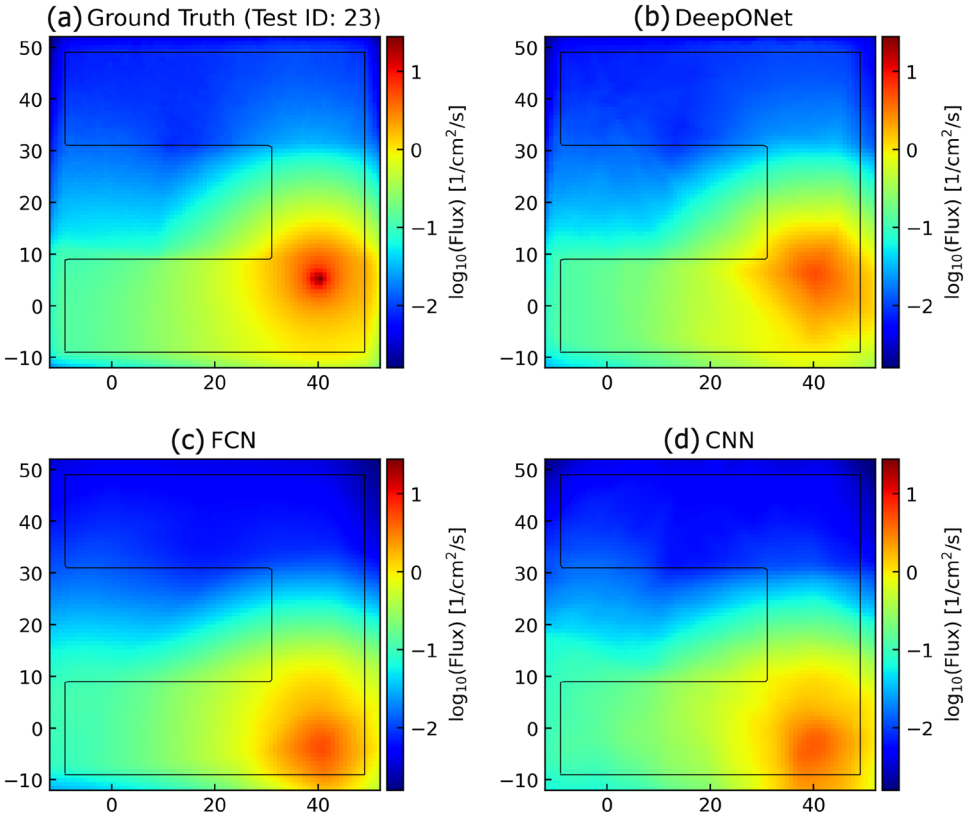

Figure 4. Comparisons between the ground truth, DeepONet, FCN, and CNN predictions for the Test ID of 23.

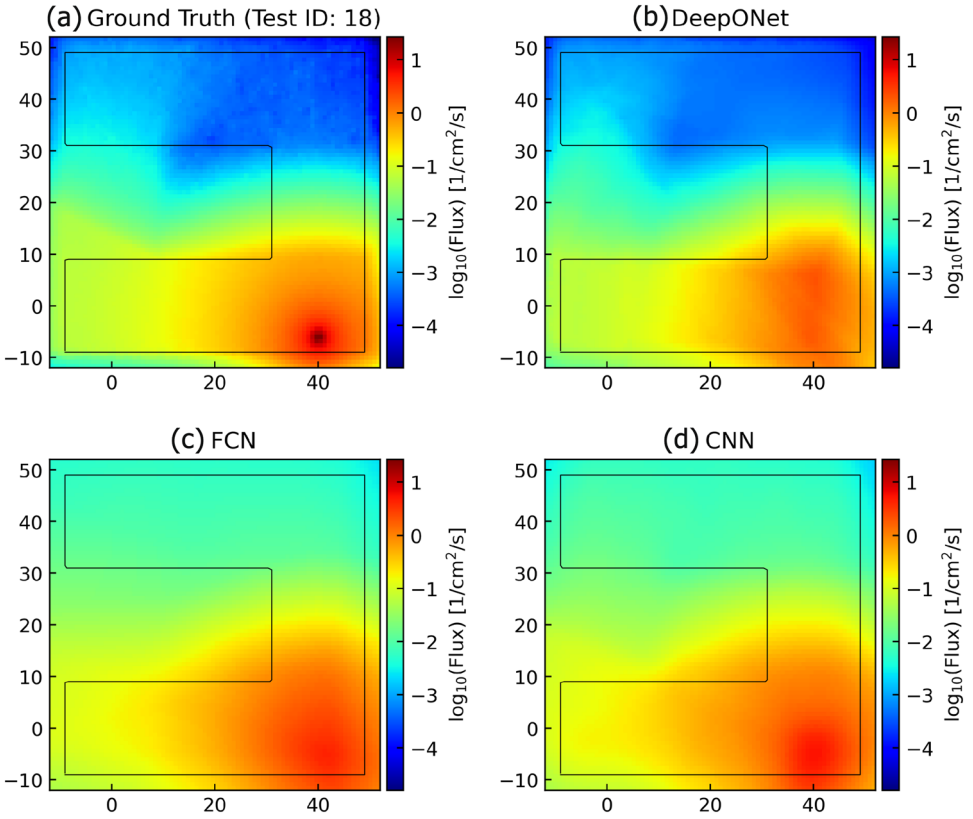

Figure 5. Comparisons between the ground truth, DeepONet, FCN, and CNN predictions for the Test ID of 18.

handle new, unseen conditions within its training domain, significantly reducing the need for retraining when compared to traditional neural network methods.

To further enhance the application of DeepONet, two important issues must be addressed. Firstly, understanding the impact of fixed sensors' number and location on model performance is crucial, considering the limited installation possibilities in harsh environments like nuclear power systems. Optimizing sensor placement and quantity under such constraints will be necessary for accurate and reliable modeling. Secondly, evaluating the DeepONet model requires attention. While overall metrics may indicate excellent performance, certain input functions might produce spurious predictions. Improving the model evaluation process, including hyperparameter tuning, is necessary to ensure robust and dependable predictions in all scenarios.

By addressing these challenges, DeepONet can be further optimized for digital twin systems, enhancing its potential to predict and analyze current and future systems based on real-time data from physical assets' sensors. This advancement opens new possibilities for various engineering applications, including nuclear engineering and beyond.

Conclusions

This paper highlights the significant potential of DeepONet as a robust surrogate modeling method with real-time prediction capability for digital twin (DT) for nuclear energy systems, showcased by its remarkable prediction accuracy and computational efficiency. The utilization of DeepONet allows for the accurate prediction of complex behaviors and spatial distributions of neutron flux, surpassing the performance of conventional ML methods like FCN and CNN. Its ability to handle functions as input data and construct operator G from training data makes it a valuable tool for capturing the intricate behavior of nuclear systems in real-time within the context of DT. The speed of execution, significantly faster than traditional simulations, positions DeepONet as a promising method to enable real-time predictions based on sensor data from physical assets. It also demonstrates consistent improvement in performance with increasing training datasets, making it a versatile and scalable solution for nuclear systems.

While DeepONet shows great promise, the study also sheds light on challenges that need further investigation. The impact of fixed sensors and the optimal sensor placement for improved model performance requires careful consideration, given the constraints in harsh operating environments like nuclear power systems. Additionally, developing more effective model evaluation methods is crucial to ensure reliable predictions and robustness.

Key areas for future research include: (1) Enhancing the multifidelity (including sparse and noisy dataset) DeepONet architecture to integrate diverse data sources for improved digital twin fidelity and performance. (2) Capturing sensor degradation over time into the DeepONet model to ensure reliable

predictions. (3) Developing on-the-fly temporal synchronization module with the physical asset. (4) Multiscale uncertainty quantification.

Data availability

The datasets used and/or analysed during the current study available from the corresponding author on reasonable request.

Received: 5 October 2023; Accepted: 11 January 2024
Published online: 24 January 2024

References

1. Yadav, V. *et al.* Technical challenges and gaps in digital-twin-enabling technologies for nuclear reactor applications (2021).
2. Yadav, V. *et al.* Project Summary of Digital Twin Regulatory Viability in Nuclear Energy Applications. U.S. Nuclear Regulatory Commission (2022).
3. Yadav, V. *et al.* State-of-Technology and Technical Challenges in Advanced Sensors, Instrumentation, and Communication to Support Digital Twin for Nuclear Energy Application. U.S. Nuclear Regulatory Commission (2023).
4. Yadav, V. *et al.* Digital Twins for Nuclear Safeguards and Security: Assessment of Challenges, Opportunities, and Current State-of-Practice. U.S. Nuclear Regulatory Commission (2023).
5. Yadav, V. *et al.* Technical Challenges and Gaps in Integration of Advanced Sensors, Instrumentation, and Communication Technologies with Digital Twins for Nuclear Application. U.S. Nuclear Regulatory Commission (2023).
6. Kobayashi, K. *et al.* Non-intrusive uncertainty quantification for U_3Si_2 and UO_2 fuels with SiC/SiCcladding using BISON for digital twin-enabling technology. arXiv preprint arXiv:2211.13687 (2022).
7. Kazuma, K., Daniell, J., Sakib, M. N., Kumar, D. & Alam, S. Components of intelligent digital twin framework for complex nuclear systems. In *13th Nuclear Plant Instrumentation, Control & Human–Machine Interface Technologies (NPIC &HMIT 2023)* (2023).
8. IBM. What is a digital twin. https://www.ibm.com/topics/what-is-a-digital-twin. Accessed: 01.09.2023.
9. Alam, S. *et al.* Neutronic feasibility of civil marine small modular reactor core using mixed D_2O+H_2O coolant. *Nucl. Eng. Des.* **359**, 110449. https://doi.org/10.1016/j.nucengdes.2019.110449 (2020).
10. Alam, S. *et al.* Neutronic investigation of alternative & composite burnable poisons for the soluble-boron-free and long life civil marine small modular reactor cores. *Sci Rep.* **9**, 19591. https://doi.org/10.1038/s41598-019-55823-2 (2019).
11. Rahman, M. *et al.* Leveraging industry 4.0–deep learning, surrogate model and transfer learning with uncertainty quantification incorporated into digital twin for nuclear system. In *Handbook of Smart Energy Systems. Springer, Cham.* https://doi.org/10.1007/978-3-030-72322-4_192-1 (2022).
12. Daniell, J. *et al.* Physics-informed multi-stage deep learning framework development for digital twin-centred state-based reactor power prediction. arXiv preprint arXiv:2211.13157 (2022).
13. Khan, A. *et al.* Digital twin and artificial intelligence incorporated with surrogate modeling for hybrid and sustainable energy systems. In *Handbook of Smart Energy Systems. Springer, Cham.* https://doi.org/10.1007/978-3-030-72322-4_147-1 (2022).
14. Moloko, L. E., Bokov, P. M., Wu, X. & Ivanov, K. N. Prediction and uncertainty quantification of SAFARI-1 axial neutron flux profiles with neural networks. *Ann. Nucl. Energy* **188**, 109813. https://doi.org/10.1016/j.anucene.2023.109813 (2023).
15. Cadini, F., Zio, E. & Pedroni, N. Simulating the dynamics of the neutron flux in a nuclear reactor by locally recurrent neural networks. *Ann. Nucl. Energy* **34**, 483–495. https://doi.org/10.1016/j.anucene.2007.02.013 (2007).
16. Hadad, K. & Piroozmand, A. Application of cellular neural network (CNN) method to the nuclear reactor dynamics equations. *Ann. Nucl. Energy* **34**, 406–416 (2007).
17. Wang, J. *et al.* Surrogate modeling for neutron diffusion problems based on conservative physics-informed neural networks with boundary conditions enforcement. *Ann. Nucl. Energy* **176**, 109234. https://doi.org/10.1016/j.anucene.2022.109234 (2022).
18. Lamarsh, J. R. *et al. Introduction to Nuclear Engineering* Vol. 3 (Prentice hall, 2001).
19. Lu, L., Jin, P., Pang, G., Zhang, Z. & Karniadakis, G. E. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nat. Mach. Intell.* **3**, 218–229 (2021).
20. Kobayashi, K., Daniell, J. & Alam, S. B. Improved generalization with deep neural operators for engineering systems: Path towards digital twin. *Eng. Appl. Artif. Intell.* (2024).
21. Kobayashi, K. & Alam, S. B. Explainable, interpretable, and trustworthy AI for an intelligent digital twin: A case study on remaining useful life. *Eng. Appl. Artif. Intell.* **129**, 107620 (2024).
22. Chen, T. & Chen, H. Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE Trans. Neural Netw.* **6**, 911–917 (1995).
23. Cai, S., Wang, Z., Lu, L., Zaki, T. A. & Karniadakis, G. E. DeepM &Mnet: Inferring the electroconvection multiphysics fields based on operator approximation by neural networks. *J. Comput. Phys.* **436**, 110296. https://doi.org/10.1016/j.jcp.2021.110296 (2021).
24. Lu, L., Meng, X., Mao, Z. & Karniadakis, G. E. DeepXDE: A deep learning library for solving differential equations. *SIAM Rev.* **63**, 208–228. https://doi.org/10.1137/19M1274067 (2021).
25. Goswami, S., Yin, M., Yu, Y. & Karniadakis, G. E. A physics-informed variational DeepONet for predicting crack path in quasi-brittle materials. *Comput. Methods Appl. Mech. Eng.* **391**, 114587. https://doi.org/10.1016/j.cma.2022.114587 (2022).
26. Lin, C. *et al.* Operator learning for predicting multiscale bubble growth dynamics. *J. Chem. Phys.* **154**, 104118 (2021).
27. Mao, Z., Lu, L., Marxen, O., Zaki, T. A. & Karniadakis, G. E. DeepM &Mnet for hypersonics: Predicting the coupled flow and finite-rate chemistry behind a normal shock using neural-network approximation of operators. *J. Comput. Phys.* **447**, 110698. https://doi.org/10.1016/j.jcp.2021.110698 (2021).
28. Sato, T. *et al.* Features of particle and heavy ion transport code system (PHITS) version 3.02. *J. Nucl. Sci. Technol.* **55**, 684–690 (2018).
29. Shibata, K. *et al.* JENDL-4.0: A new library for nuclear science and engineering. *J. Nucl. Sci. Technol.* **48**, 1–30 (2011).

Acknowledgments

The authors also acknowledge that ChatGPT 4 has been solely used for minor language editing.

Author contributions

Conceptualization, S.A. and K.K.; methodology, K.K. and S.A.; Writing - original draft, K.K.; Writing - review & editing, S.A.; Formal analysis, K.K.; Visualization, K.K.; Data curation, K.K.; Supervision, S.A.; Project administration, S.A.; All authors have read and agreed to the published version of the manuscript.

Funding
The computational part of this work was supported in part by the National Science Foundation (NSF) under Grant No. OAC-1919789.

Conflicts of interest
The authors declare no competing interests.

Additional information
Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-024-51984-x.

Correspondence and requests for materials should be addressed to S.A.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.