Computer Modeling in Engineering & Sciences

Tech Science Press

https://doi.org/10.32604/cmes.2026.084608

ARTICLE

Linear–Nonlinear Fusion Neural Operator for Partial Differential Equations

Heng Wu[1,2], **Junjie Wang**[1,2] **and Benzhuo Lu**[1,2,*]

[1]State Key Laboratory of Mathematical Sciences (SKLMS), Institute of Computational Mathematics and Scientific/Engineering Computing (ICMSEC), National Center for Mathematics and Interdisciplinary Sciences (NCMIS), Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China

[2]School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing, China

*Corresponding Author: Benzhuo Lu. Email: bzlu@lsec.cc.ac.cn

Received: 26 April 2026; Accepted: 02 July 2026; Published: 28 August 2026

ABSTRACT: Neural operator learning directly constructs the mapping relationship from the equation parameter space to the solution space, enabling efficient direct inference in practical applications without the need for repeated solution of partial differential equations (PDEs)—an advantage that is difficult to achieve with traditional numerical methods. In this work, we investigate a two-path formulation that combines affine and nonlinear computational components within such operator mappings to improve learning efficiency. This yields a novel network structure, namely the Linear–Nonlinear Fusion Neural Operator (LNF-NO), which models operator mappings via the multiplicative fusion of a linear component and a nonlinear component, thus achieving a lightweight and structurally transparent representation. This two-path formulation is designed to capture complex solution features at the operator level while retaining architectural simplicity. LNF-NO naturally supports multiple functional inputs and is applicable to both regular grids and fixed irregular-node discretizations. Across a diverse suite of PDE operator-learning benchmarks, including nonlinear Poisson–Boltzmann equations and multi-physics coupled systems, LNF-NO is typically substantially faster to train than several representative neural operator baselines, while achieving comparable or improved accuracy across most tested cases. On the tested three-dimensional Poisson–Boltzmann case, LNF-NO achieves competitive accuracy while requiring substantially less training time than the three-dimensional Fourier Neural Operator and Transolver baselines.

KEYWORDS: Neural operator; partial differential equations; scientific machine learning; operator learning; Poisson–Boltzmann equation; multi-physics systems

1 Introduction

Neural operator learning has emerged as a central paradigm in scientific machine learning for approximating solution operators of partial differential equations (PDEs). By learning mappings between function spaces, neural operators aim to amortize the cost of repeatedly solving PDEs under varying boundary conditions, source terms, or coefficients. Such many-query settings arise ubiquitously in physical, chemical, and biological modeling, where repeated numerical solves constitute a major computational bottleneck [1–3]. Neural operators provide fast surrogate models that deliver approximate PDE solutions with accuracy often sufficient for downstream analysis and decision-making, while substantially reducing computational cost compared with repeated high-fidelity simulations [4,5].

Among existing approaches, the Deep Operator Network (DeepONet) [4] and the Fourier Neural Operator (FNO) [6] represent the most widely used and representative neural operator models. DeepONet employs a branch–trunk decomposition to learn operator mappings from input functions to solution

fields, while FNO leverages global convolution in the Fourier domain to efficiently capture long-range dependencies on regular grids. Beyond these baselines, a growing family of neural operator architectures has been proposed, including geometry-informed operators [7], the U-shaped Neural Operator (UNO) [8], and transformer-inspired or attention-based formulations such as Transolver [9] and PDEformer [10].

More recently, attention-based and geometry-aware neural operators have been further developed for irregular meshes, multiple input functions, and complex geometries. Representative examples include the General Neural Operator Transformer (GNOT) [11], Geometry-Informed Neural Operator Transformer (GINOT) [12], and signed distance function (SDF)-based neural operator transformers [13]. Other related directions include latent-space neural operators such as Latent Spectral Models (LSM) [14], non-uniform-data neural operators such as NUNO [15], multigrid-based operator parameterizations such as Multigrid Neural Operators (MgNO) [16], and attention- or equivariance-based variants aimed at improved operator learning and generalization, such as Orthogonal Neural Operators (ONO) [17] and group-equivariant Fourier Neural Operators (G-FNO) [18]. These methods mainly focus on attention backbones, explicit geometry encoders, latent representations, non-uniform data processing, multigrid parameterizations, or symmetry/resolution generalization, whereas the present work focuses on a lightweight linear–nonlinear fusion core. Accordingly, we include Transolver as a representative attention-based baseline in our experiments, while exhaustive benchmarking against all recent neural-operator variants is left for future work.

Despite this progress, several challenges remain in practical operator-learning settings. First, nonlinear interactions in PDE operators can be difficult to represent or optimize using purely additive architectural compositions. Second, many real-world problems involve multiple functional inputs, such as boundary traces, source terms, or coefficient fields, as well as coupled multi-field outputs. Third, training efficiency and optimization stability become increasingly important as problem complexity grows, particularly for three-dimensional PDEs and large-scale benchmarks.

Non-additive interactions have proven effective across a wide range of deep learning architectures. Feature-wise modulation mechanisms, such as Feature-wise Linear Modulation (FiLM), apply learned, input-dependent scaling to intermediate representations and can enhance expressivity with minimal computational overhead [19]. More broadly, attention mechanisms [20] and mixture-of-experts models [21] illustrate that structured, conditional interactions between multiple components can improve representational capacity and training efficiency. These developments motivate exploring multiplicative fusion as a general design principle for neural operator architectures.

Our architectural design is further motivated by numerical insights into semi-linear elliptic PDEs and coupled systems. For instance, in biomolecular electrostatics and ion transport, solutions to nonlinear equations such as Poisson–Boltzmann or Poisson–Nernst–Planck are often viewed as being predominantly governed by their corresponding linear approximations, modulated by nonlinear correction terms [22,23]. Inspired by this, we hypothesize that combining separate affine and nonlinear computational paths within the neural architecture provides a useful PDE-motivated inductive bias.

In this work, we propose the Linear–Nonlinear Fusion Neural Operator (LNF-NO), which models operator behavior through the multiplicative interaction of a linear component and a nonlinear component. LNF-NO treats these components symmetrically: the two branches interact directly through element-wise fusion, rather than being combined through hierarchical or purely additive structures. This design provides a lightweight and structurally transparent mechanism for capturing nonlinear effects while maintaining architectural simplicity. LNF-NO naturally supports multiple functional inputs by encoding each input function independently and fusing their latent representations within the operator core. The architecture can be paired with a lightweight grid-based decoder for standard benchmark problems, or used in a decoder-free setting for fixed irregular-node discretizations where solutions are predicted on prescribed node sets.

We evaluate LNF-NO on a diverse suite of PDE operator-learning benchmarks, including nonlinear Poisson–Boltzmann equations, coupled multi-field systems, and fixed irregular-domain discretizations. Across these tasks, LNF-NO consistently improves training efficiency relative to representative neural operator baselines, while achieving comparable or improved accuracy across most tested cases. In particular, we present a three-dimensional Poisson–Boltzmann case study in which LNF-NO achieves competitive accuracy while requiring substantially less training time than the three-dimensional Fourier Neural Operator and Transolver baselines.

The main contributions of this work are summarized as follows:

- We introduce a PDE-motivated linear–nonlinear multiplicative fusion mechanism that combines an affine response path with input-dependent nonlinear modulation. This design provides an efficiency-oriented architectural inductive bias, yielding substantially improved training efficiency on a broad range of benchmarks while achieving comparable or improved accuracy across most tested cases.
- Our simple and flexible architecture naturally accommodates multiple functional inputs and predicts multiple coupled output fields within a unified framework and can be applied to both regular-grid and fixed irregular-node discretizations while preserving the same linear–nonlinear fusion core.
- We evaluate the empirical applicability and computational performance of LNF-NO on a diverse suite of benchmarks, including three-dimensional systems and strongly nonlinear equations (e.g., Poisson–Boltzmann), where the proposed architecture shows favorable empirical performance and training efficiency.

2 Methodology

2.1 Operator Learning with Input/Output Discretization

Let $\mathcal{G} : \mathcal{X} \to \mathcal{Y}$ denote the solution operator of a PDE posed on a bounded domain $\Omega \subset \mathbb{R}^d$, mapping input functions (e.g., boundary traces, coefficient fields, or source terms) to solution fields. In practice, operator learning is carried out after discretizing both inputs and outputs.

Input discretization. We consider multi-function inputs with M components

$$x = \left(x^{(1)}, \ldots, x^{(M)}\right), \qquad x^{(m)} \in \mathcal{X}^{(m)}.$$

Each component is discretized by a sampling or projection operator $\mathcal{I}_h^{(m)} : \mathcal{X}^{(m)} \to \mathbb{R}^{d_{\text{in}}^{(m)}}$, yielding $x_h^{(m)} := \mathcal{I}_h^{(m)}(x^{(m)})$. Stacking all components gives $x_h := \mathcal{I}_h(x) \in \mathbb{R}^{d_{\text{in}}}$, where $d_{\text{in}} = \sum_{m=1}^{M} d_{\text{in}}^{(m)}$. In general, $\mathcal{I}_h$ is not invertible.

Data-generating family and lifting. The dataset is generated from a restricted family $\mathcal{A} \subset \mathcal{X}$. Accordingly, we only require the discrete learning target to be well defined on the set of discrete inputs observed in data. Let $K_h := \mathcal{I}_h(\mathcal{A}) \subset \mathbb{R}^{d_{\text{in}}}$. We fix a (possibly non-unique) selection map $\mathcal{L}_h : K_h \to \mathcal{A}$ such that $\mathcal{I}_h(\mathcal{L}_h(x_h)) = x_h$ for all $x_h \in K_h$. The choice of $\mathcal{L}_h$ reflects the data-generation procedure and does not require uniqueness.

Output discretization. Let $\mathcal{I}_h^{\text{out}} : \mathcal{Y} \to \mathbb{R}^{d_{\text{out}}}$ denote an output sampling or projection map, such as evaluation on a fixed grid or a fixed point set for irregular domains. For C_{out} output fields, typically $d_{\text{out}} = C_{\text{out}} N_{\text{out}}$.

Discrete target operator. We define the discrete operator learning target

$$F_h^\star := \mathcal{I}_h^{\text{out}} \circ \mathcal{G} \circ \mathcal{L}_h, \qquad F_h^\star : K_h \to \mathbb{R}^{d_{\text{out}}}. \tag{1}$$

All models are trained and evaluated in the discrete output space $\mathbb{R}^{d_{\text{out}}}$. In this discrete setting, the output coordinates are prescribed by $\mathcal{I}_h^{\text{out}}$ and are not treated as sample-dependent coordinate inputs to the operator core.

2.2 LNF-NO Architecture (Multi-Input, Multi-Output)

Fig. 1 overviews the LNF-NO architecture, generally formulated to handle multiple functional inputs (e.g., boundary conditions and source terms) and predict multiple coupled output fields (multi-output). Given $x_h = \left(x_h^{(1)}, \ldots, x_h^{(M)}\right)$, each input component is encoded independently:

$$z^{(m)} = \Phi_\theta^{(m)}\left(x_h^{(m)}\right) \in \mathbb{R}^{d_m}, \qquad m = 1, \ldots, M,$$
$$z = \left[z^{(1)}; \ldots; z^{(M)}\right] \in \mathbb{R}^{d_z}.$$

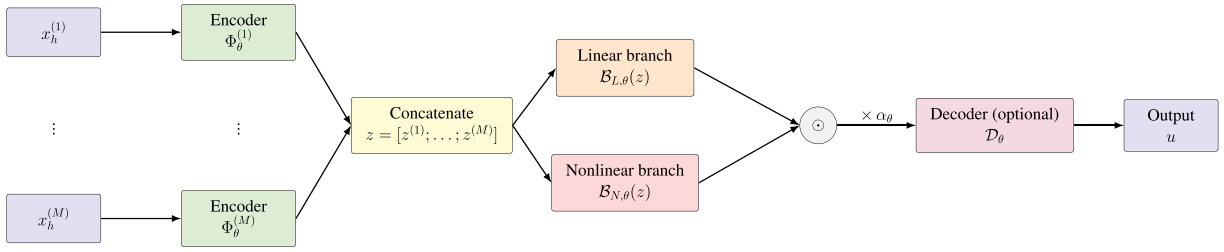


Figure 1: Architecture of the proposed Linear–Nonlinear Fusion Neural Operator (LNF-NO). Each input component (typically a discretized function, e.g., boundary traces or source fields) is encoded separately and concatenated into a latent representation. An operator core then fuses a linear branch and a nonlinear branch via element-wise multiplication ($\odot$), producing a raw prediction that can be optionally refined by a lightweight decoder. The current implementation is formulated as a discrete neural operator on a prescribed output discretization, such as a fixed grid, space–time grid, or finite-element node set. Therefore, spatial or spatio-temporal coordinates are fixed by the output discretization and are not shown as sample-wise coordinate inputs. Arrows indicate the data flow.

To balance expressivity and optimization efficiency, the operator core consists of a linear branch and a nonlinear branch fused multiplicatively:

$$u_{\text{raw}} = \alpha_\theta \cdot \left(\mathcal{B}_{L,\theta}(z) \odot \mathcal{B}_{N,\theta}(z)\right) \in \mathbb{R}^{d_{\text{out}}}, \tag{2}$$

where $\odot$ denotes element-wise multiplication and α_θ is a learnable scalar (see the Supplementary Materials for initialization details). Both branches map the latent representation to the discrete output space, $\mathcal{B}_{L,\theta}, \mathcal{B}_{N,\theta} : \mathbb{R}^{d_z} \to \mathbb{R}^{d_{\text{out}}}$, with $\mathcal{B}_{L,\theta}$ being affine and $\mathcal{B}_{N,\theta}$ a nonlinear multilayer perceptron (MLP).

Intuition

The proposed fusion strategy is motivated by both numerical treatments of nonlinear PDEs and modulation-based deep learning architectures. For nonlinear equations such as Poisson–Boltzmann and Poisson–Nernst–Planck, linearized formulations are often used to provide a baseline or initialization, which is subsequently modified through nonlinear treatment [22,23]. LNF-NO translates this numerical perspective into an architectural inductive bias: the affine branch provides a direct response path, while the nonlinear branch produces input-dependent modulation.

This multiplicative interaction is different from an additive residual correction, such as the residual connection structure commonly used in ResNets [24]. In an additive formulation, the two branches independently contribute terms that are summed at the output. In LNF-NO, the nonlinear branch acts directly

on the affine response through element-wise multiplication, thereby changing its magnitude and spatial pattern in an input-dependent manner. This mechanism is related in spirit to feature-wise modulation such as FiLM [19], but is formulated here at the discrete operator-output level to reflect the linearized-response and nonlinear-modulation structure motivating the model.

The terms "linear" and "nonlinear" refer to the mathematical forms of the two learned branches. They encode a PDE-motivated structural hypothesis, but we do not require the trained branches to coincide exactly, component by component, with a unique analytical decomposition of the underlying continuous PDE solution operator. Accordingly, this interpretation should be understood as a heuristic structural hypothesis rather than as a rigorously identified branch-wise physical decomposition.

Optionally, a lightweight decoder $\mathcal{D}_\theta$ refines local consistency on regular grids:

$$F_\theta(x_h) = \begin{cases} u_{\text{raw}}, & \text{decoder-free,} \\ \mathcal{D}_\theta(u_{\text{raw}}), & \text{with decoder.} \end{cases}$$

2.3 Approximation Guarantee

The following result concerns the expressivity of LNF-NO under the fixed input/output discretization introduced in Section 2; it is not intended as a convergence or discretization theorem for the underlying continuous PDE operator.

Theorem 1 (Universal approximation): *Assume $K_h \subset \mathbb{R}^{d_{\text{in}}}$ is compact and $F_h^\star \in C(K_h; \mathbb{R}^{d_{\text{out}}})$.*

If the feedforward components and latent representation of LNF-NO are allowed sufficient width and employ an activation function satisfying a standard finite-dimensional universal approximation theorem, then for any $\varepsilon > 0$ there exist parameters θ such that

$$\sup_{x_h \in K_h} \|F_\theta(x_h) - F_h^\star(x_h)\| \le \varepsilon.$$

Proof Sketch: By choosing a decoder-free realization, allowing the encoders and latent representation to preserve the discrete input information, setting $\mathcal{B}_{L,\theta}(z) \equiv \mathbf{1}$ and $\alpha_\theta = 1$, and allowing the nonlinear branch sufficient width, the architecture contains a standard feedforward approximator from $\mathbb{R}^{d_{\text{in}}}$ to $\mathbb{R}^{d_{\text{out}}}$ as a special case.

Universal approximation then follows from classical results for non-polynomial activations [25,26]. □

The purpose of Theorem 1 is to show that the multiplicative fusion constraint does not reduce the universal approximation capability of the discrete architecture. It does not by itself provide operator-specific convergence rates, discretization consistency, mesh-resolution generalization, stability bounds, or PDE-aware error estimates, which require additional assumptions on the continuous solution operator and the discretization.

2.4 Optimization

We train LNF-NO by supervised learning on pairs (x_h, y_h) with $y_h = F_h^\star(x_h)$. We use the averaged relative ℓ_2 error across output fields as the loss:

$$\mathcal{L}(\theta) = \frac{1}{B} \sum_{b=1}^{B} \frac{1}{C_{\text{out}}} \sum_{c=1}^{C_{\text{out}}} \frac{\|\hat{y}_{h,b}^{(c)} - y_{h,b}^{(c)}\|_2}{\|y_{h,b}^{(c)}\|_2 + \varepsilon}. \tag{3}$$

All losses and metrics are computed on the decoded physical scale after inverse normalization. We optimize using AdamW [27]; task-specific hyperparameters, including learning rates and training schedules, are provided in the Supplementary Materials. Following common practice, we exclude bias terms and the scalar fusion scale α_θ from weight decay, since they act as calibration parameters rather than capacity-controlling weights.

3 Results

Evaluation protocol. All models are trained and evaluated under a common evaluation protocol. For each task, we use identical training and test sets across all methods with a fixed 9:1 split, and report the mean relative ℓ_2 error on the test set. DeepONet is trained for 5000 epochs, while the other neural operator baselines and LNF-NO are trained for 500 epochs. Training time refers to the wall-clock training time measured under the same hardware setting and includes the full training process. Task-specific hyperparameters, including learning rates, batch sizes, and architectural settings, are provided in the Supplementary Materials.

Task definitions. In all benchmarks, we learn solution operators that map boundary conditions and/or input fields to solution fields evaluated on a fixed set of points. We summarize the PDE tasks considered in this work below.

Laplace. We study the two-dimensional (2D) Laplace equation $\Delta u = 0$ in $\Omega = (0,1)^2$ with Dirichlet boundary condition $u|_{\partial\Omega} = g$, and learn the operator mapping the boundary trace g to the solution field u on a fixed grid (see the Supplementary Materials). Here, Ω denotes the computational domain, $\partial\Omega$ is its boundary, u is the scalar solution field, g is the prescribed boundary trace, and Δ denotes the Laplace operator.

Burgers. We consider the one-dimensional (1D) viscous Burgers equation

$$\partial_t u + u\,\partial_x u = \nu\,\partial_{xx} u \quad \text{on } x \in [0, 2\pi),\ t \in [0, T], \tag{4}$$

with periodic boundary conditions. Here, $u = u(x,t)$ denotes the time-dependent solution field, ν is the viscosity coefficient, and the subscripts t, x, and xx denote partial derivatives with respect to time and space. The operator maps the initial condition $u(\cdot, 0)$ to the solution trajectory $u(\cdot, t)$ on a fixed space–time grid (see the Supplementary Materials).

Darcy flow. We benchmark the elliptic Darcy equation $-\nabla \cdot (a(x)\nabla u) = f$ in $\Omega = (0,1)^2$, where $a(x)$ denotes the permeability field. Here, u is the solution field, f is the forcing term, Ω is the computational domain, and $\nabla\cdot$ and ∇ denote the divergence and gradient operators, respectively. Both smooth and piecewise constant permeability regimes are considered (see the Supplementary Materials).

Poisson–Boltzmann (PB). We evaluate the nonlinear Poisson–Boltzmann operator

$$-\Delta u + k\,\sinh(u) = f \quad \text{in } \Omega, \qquad u|_{\partial\Omega} = g, \tag{5}$$

where $k > 0$ partially controls the strength of the nonlinearity. Here, u denotes the scalar PB solution field, f is the source term, g is the prescribed Dirichlet boundary condition, Ω is the computational domain, $\partial\Omega$ is its boundary, and Δ denotes the Laplace operator. The source-free case corresponds to $f = 0$, while the source-driven case uses $f \neq 0$ (see, e.g., [22] for background on numerical PB models and the Supplementary Materials for benchmark details).

Irregular-domain PB. To evaluate the applicability of LNF-NO to irregular node-based discretizations, we further consider source-free PB problems posed on irregular planar domains with multiple boundary components, discretized using finite element methods and represented in a node-based format. In these benchmarks, we do not use a dedicated geometry encoder such as boundary point-cloud attention or a

signed distance function (SDF). Instead, for each irregular-domain benchmark, the geometry is represented implicitly by the prescribed finite-element node set, and the model predicts the PB solution values on this fixed set of nodes (see the Supplementary Materials).

Poisson–Nernst–Planck (PNP). Finally, we benchmark a multi-field coupled operator corresponding to the Poisson–Nernst–Planck system (see, e.g., [23] for the classical PNP formulation and the Supplementary Materials for dataset details). A dimensionless form is taken where physical parameters are set to unity for operator-learning purposes:

$$-\Delta\phi = c_+ - c_-, \qquad \nabla \cdot (\nabla c_\pm \pm c_\pm \nabla\phi) = 0. \tag{6}$$

Here, ϕ denotes the electric potential, c_+ and c_- denote the positive and negative ion concentration fields, respectively, $c_\pm$ denotes the two concentration fields collectively, and $\nabla\cdot$, ∇, and Δ denote the divergence, gradient, and Laplace operators. The operator maps boundary traces to the electric potential ϕ and ion concentration fields $c_\pm$ on a fixed grid.

3.1 Baseline Benchmarks on Regular Grids

3.1.1 Single-Input Operators

We first consider standard single-input operator-learning benchmarks on regular grids, including Laplace, Burgers, Darcy flow (smooth and discontinuous coefficients), and the source-free Poisson–Boltzmann equation ($k = 1$). These tasks span linear and nonlinear elliptic and parabolic operators with varying degrees of complexity.

The quantitative results are summarized in Table 1. Across these single-input benchmarks, LNF-NO consistently exhibits a clear advantage in training efficiency, and it attains the best accuracy on Laplace, Darcy flow with smooth coefficients, and the source-free Poisson–Boltzmann benchmark. On the Darcy flow (Smooth) benchmark, for example, LNF-NO achieves the lowest error (1.79×10^{-3}) among the compared models, while requiring substantially less training time than the competing baselines.

Table 1: Baseline benchmarks on regular grids (single-input operators). We report the test mean relative ℓ_2 error and wall-clock training time under the unified protocol. Bold values indicate the best result within each equation.

Equation	Model	#Params	Epochs	Training Time (s)	Test Rel. ℓ_2
Laplace	DeepONet	973,313	5000	1470.08	1.16e−02
	FNO	9,133,866	500	999.22	1.66e−02
	UNO	9,207,722	500	1393.00	2.05e−02
	Transolver	2,158,538	500	7322.38	1.04e−02
	LNF-NO	2,891,956	500	**257.57**	**7.43e−03**
Burgers	DeepONet	938,497	5000	1760.09	1.05e−01
	FNO	8,413,953	500	2316.45	1.20e−02
	UNO	8,487,809	500	3143.00	**1.10e−02**
	Transolver	1,438,625	500	43,598.27	2.43e−02
	LNF-NO	4,037,346	500	**911.87**	3.89e−02

(Continued)

Table 1 (continued)

Equation	Model	#Params	Epochs	Training Time (s)	Test Rel. ℓ_2
Darcy(Smooth Coeff.)	DeepONet	5,182,209	5000	4981.13	5.65e−02
	FNO	8,413,953	500	3889.17	2.58e−03
	UNO	9,587,201	500	6830.23	2.39e−03
	Transolver	1,438,625	500	302,922.64	3.82e−03
	LNF-NO	5,922,948	500	**603.17**	**1.79e−03**
Darcy(Piecewise Coeff.)	DeepONet	15,790,849	5000	36,945.35	4.87e−02
	FNO	8,413,953	500	11,127.40	1.24e−02
	UNO	9,587,201	500	119,219.69	**8.65e−03**
	Transolver	1,438,625	500	534,539.25	8.78e−03
	LNF-NO	106,488,900	500	**2893.28**	2.74e−02
PB ($k = 1$)	DeepONet	1,024,513	5000	1685.50	6.89e−02
	FNO	11,138,266	500	3700.89	8.29e−02
	UNO	11,212,122	500	5602.36	9.10e−02
	Transolver	4,228,730	500	32,475.36	8.30e−02
	LNF-NO	8,027,156	500	**671.30**	**1.99e−02**

This optimization advantage is further illustrated in Fig. 2, where LNF-NO shows markedly faster error reduction on the source-free PB benchmark, both as a function of training epoch and of wall-clock time.

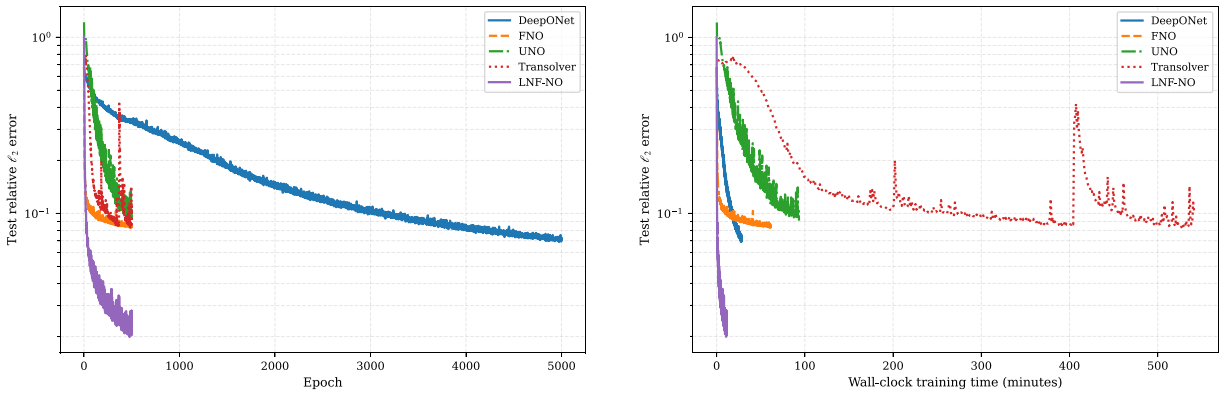


Figure 2: Convergence comparison on the source-free Poisson–Boltzmann benchmark ($k = 1$). Left: test relative ℓ_2 error vs. training epoch. Right: test relative ℓ_2 error vs. wall-clock training time. LNF-NO exhibits consistently faster error reduction than the compared baselines, especially when measured against actual training time.

3.1.2 Multi-Input Operators

We next consider multi-input operator-learning problems, where the input consists of multiple function-valued components. These include a source-driven Poisson–Boltzmann equation and a coupled Poisson–Nernst–Planck system.

Table 2 presents the performance on these multi-input tasks. LNF-NO naturally accommodates multi-function inputs through independent encoders and a shared operator core. For the source-driven PB

equation, LNF-NO attains the best accuracy among the compared models (1.47×10^{-2}) while requiring substantially less training time than FNO, UNO, and Transolver. For the coupled PNP system, the lowest error is achieved by Transolver, whereas LNF-NO remains competitive in accuracy and offers a clear advantage in training efficiency.

Table 2: Baseline benchmarks on regular grids (multi-input operators). Each task takes multiple function-valued inputs, such as boundary traces and a source field, and predicts fields on a fixed grid. Bold values indicate the best result within each equation.

Equation	Model	#Params	Epochs	Training Time (s)	Test Rel. ℓ_2
PB with source ($k = 1$)	DeepONet	4,096,769	5000	2549.46	2.64e−01
	FNO	11,138,330	500	3751.44	3.71e−02
	UNO	11,212,186	500	5683.71	3.63e−02
	Transolver	4,228,986	500	28,264.52	4.85e−02
	LNF-NO	9,662,852	500	**988.78**	**1.47e−02**
PNP system	DeepONet	1,841,667	5000	12,299.65	6.01e−02
	FNO	8,409,987	500	8197.63	2.21e−02
	UNO	8,483,843	500	9873.26	3.64e−02
	Transolver	1,439,395	500	47,281.55	**8.19e−03**
	LNF-NO	35,954,602	500	**2872.09**	1.88e−02

3.2 Varying Nonlinear Strengths in Poisson–Boltzmann Equations

We further investigate empirical performance under increasing nonlinearity using the source-free Poisson–Boltzmann equation with $k = 0.01$, 1, and 100. As k increases, the operator becomes increasingly stiff and exhibits sharp boundary layers.

The results under varying nonlinearity strengths are detailed in Table 3. Across all three nonlinearity regimes, LNF-NO attains the lowest test error among the compared models while maintaining a clear advantage in training efficiency. For the stiffest case ($k = 100$), LNF-NO achieves a relative error of 2.28×10^{-2}, significantly outperforming all compared baselines, suggesting an empirical benefit of the proposed multiplicative fusion in this stiff PB setting.

Table 3: Poisson–Boltzmann regimes with varying nonlinear strengths (source-free). We vary the nonlinearity strength k in (5) with $f = 0$ and report test mean relative ℓ_2 error and training time. Bold values indicate the best result within each equation.

Equation	Model	#Params	Epochs	Training Time (s)	Test Rel. ℓ_2
PB ($k = 0.01$)	DeepONet	1,024,513	5000	1821.46	5.72e−02
	FNO	11,138,266	500	3747.64	6.24e−02
	UNO	11,212,122	500	5862.01	6.76e−02
	Transolver	4,228,730	500	28,642.89	4.27e−02
	LNF-NO	8,027,156	500	**668.07**	**1.65e−02**

(Continued)

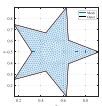
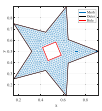
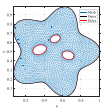
Table 3 (continued)

Equation	Model	#Params	Epochs	Training Time (s)	Test Rel. ℓ_2
PB (k = 1)	DeepONet	1,024,513	5000	1685.50	6.89e−02
	FNO	11,138,266	500	3700.89	8.29e−02
	UNO	11,212,122	500	5602.36	9.10e−02
	Transolver	4,228,730	500	32,475.36	8.30e−02
	LNF-NO	8,027,156	500	**671.30**	**1.99e−02**
PB (k = 100)	DeepONet	1,024,513	5000	2280.09	1.01e−01
	FNO	11,138,266	500	3735.93	1.01e−01
	UNO	11,212,122	500	5901.40	1.18e−01
	Transolver	4,228,730	500	28,294.86	1.07e−01
	LNF-NO	8,027,156	500	**735.33**	**2.28e−02**

3.3 Irregular Geometries

Table 4 compares the performance on three non-rectangular geometries. LNF-NO consistently attains the best accuracy on all three irregular geometries while requiring substantially less wall-clock training time than the compared baselines. On the Star domain, for example, LNF-NO attains a test relative error of 1.05×10^{-2}, substantially improving over both DeepONet and Transolver while requiring much less training time. These results highlight the applicability of LNF-NO to the tested fixed irregular-node discretizations: LNF-NO predicts solution values on prescribed unstructured node sets, whereas grid-based spectral operators such as FNO are not directly applicable in this setting.

Table 4: Performance on irregular domains for the Poisson–Boltzmann equation. The problems are posed on non-rectangular geometries with complex boundaries and discretized using unstructured node-based representations, for which grid-based spectral operators such as FNO are not directly applicable. Bold values indicate the best result for each geometry.

Geometry	Model	#Params	Epochs	Training Time (s)	Test Rel. ℓ_2
	DeepONet	973,313	5000	1050.88	3.47e−02
	Transolver	1,014,924	500	3137.79	2.92e−02
	LNF-NO	1,912,695	500	**114.79**	**1.05e−02**
	DeepONet	973,313	5000	925.12	3.46e−02
	Transolver	1,004,644	500	3012.29	2.71e−02
	LNF-NO	1,892,135	500	**109.16**	**1.12e−02**
	DeepONet	973,313	5000	1025.40	3.20e−02
	Transolver	1,411,732	500	10148.53	2.70e−02
	LNF-NO	2,706,311	500	**117.66**	**1.23e−02**

3.4 Extension to Three-Dimensional Problems

Quantitative comparisons for the three-dimensional (3D) benchmark are provided in Table 5. Although Transolver attains the lowest error, LNF-NO achieves competitive 3D accuracy (4.32×10^{-2}) while requiring substantially less training time than both 3D FNO and Transolver. Qualitatively, Fig. 3 visualizes the

central cross-sections of the predicted solution field $u(x, y, z)$, demonstrating that LNF-NO can accurately reconstruct the 3D potential distribution from boundary data.

Table 5: Results on the 3D Poisson–Boltzmann benchmark on a 33^3 grid. We report test mean relative ℓ_2 error and wall-clock training time under the unified protocol. Bold values indicate the lowest error.

Model	#Params	Epochs	Time (s)	Rel. ℓ_2
DeepONet	2.5M	5000	1349	3.34e−01
FNO (3D)	15.0M	500	7559	7.71e−02
UNO	13.0M	500	10,612	8.53e−02
Transolver	11.8M	500	53,311	**3.66e−02**
LNF-NO	21.7M	500	3253	4.32e−02

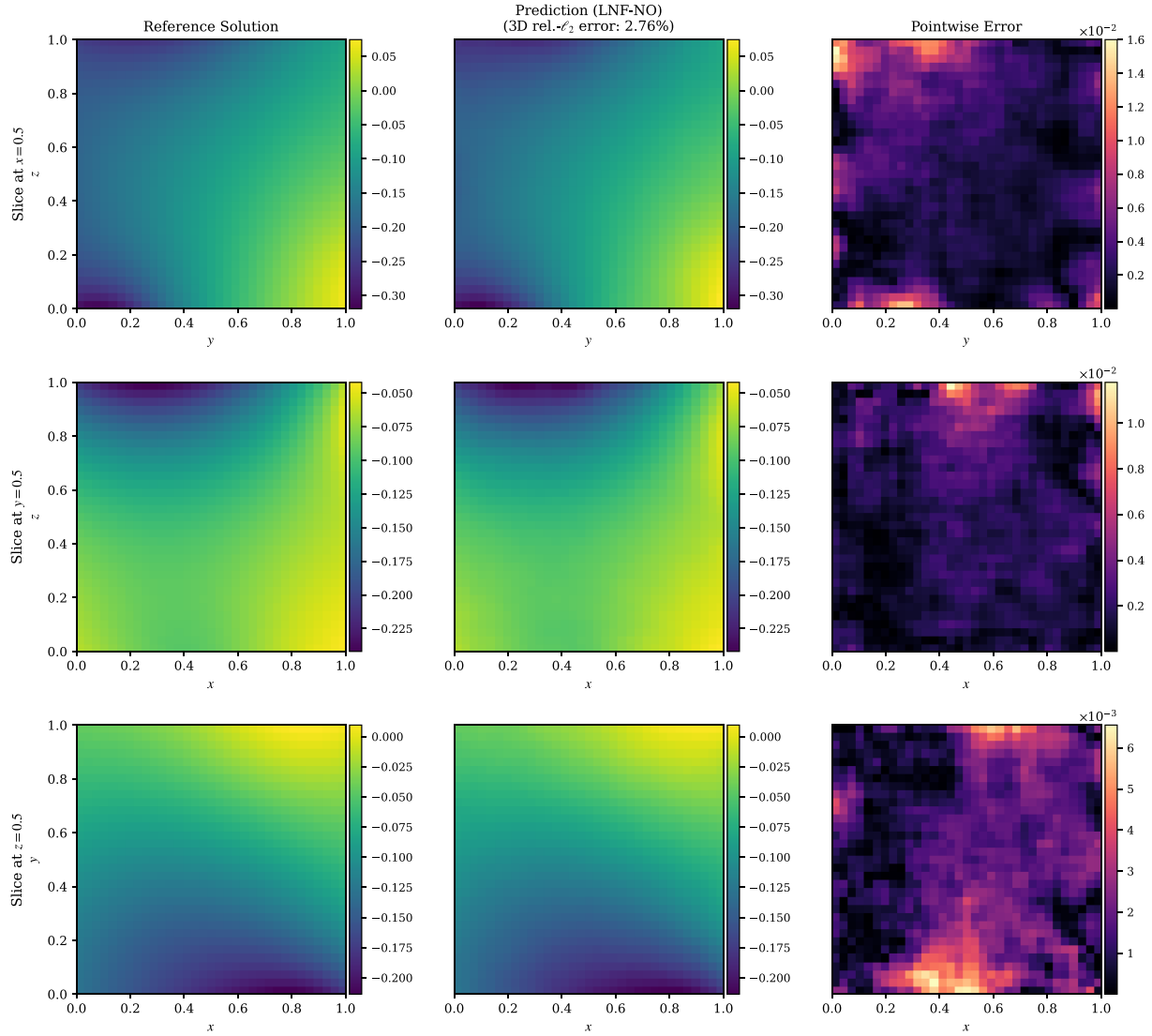


Figure 3: Three-dimensional Poisson–Boltzmann extension. Central cross-sections of the solution field $u(x, y, z)$ at $x = 0.5$, $y = 0.5$, and $z = 0.5$. The prediction is reconstructed from boundary data by LNF-NO.

Taken together, the experimental results across Tables 1–5 show a consistent pattern. On regular-grid benchmarks (Tables 1 and 2), LNF-NO achieves strong overall performance while substantially reducing wall-clock training time relative to the compared baselines. Beyond regular grids, as evidenced in Table 4, LNF-NO remains directly applicable to the tested fixed irregular-node discretizations, where grid-based spectral operators are not readily usable. In addition, on the 3D Poisson–Boltzmann benchmark (Table 5 and Fig. 3), LNF-NO shows favorable empirical optimization behavior and achieves competitive 3D accuracy while requiring substantially less training time than the 3D FNO and Transolver baselines. Overall, these results suggest that the proposed multiplicative fusion functions primarily as an efficiency-oriented inductive bias that is empirically applicable to the tested heterogeneous-input, fixed irregular-node, and three-dimensional settings, rather than a task-specific architectural specialization.

3.5 Ablation Study

To examine the role of the proposed linear–nonlinear fusion mechanism, we conduct ablation studies on the source-free PB benchmarks with $k = 0.01, 1, 100$. We first consider the full architecture with the convolutional encoder and decoder used in the regular-grid PB experiments. We compare the proposed multiplicative fusion $B_L \odot B_N$ with three variants: additive fusion $B_L + B_N$, a linear-only branch B_L, and a nonlinear-only branch B_N. All variants use the same data splits, learning rate, batch size, and training schedule as the corresponding PB experiments.

The full encoder–decoder rows of Table 6 show that the full multiplicative model achieves the lowest error under the complete encoder–decoder architecture. The improvement over the linear-only or additive variants is moderate in this setting, especially for the stiffest $k = 100$ case. This moderate difference may be partly explained by the additional nonlinear representation capacity introduced by the convolutional encoder and decoder on regular grids. Nevertheless, the full multiplicative-fusion formulation remains the best among the tested variants, suggesting that the multiplicative fusion is compatible with the encoder–decoder backbone and provides an additional empirical benefit in these experiments.

Table 6: Ablation studies of the linear–nonlinear fusion mechanism on source-free PB benchmarks. The reported values are mean relative ℓ_2 errors on the test set, and the best result in each row is highlighted in bold. The compared variants are multiplicative fusion $B_L \odot B_N$, additive fusion $B_L + B_N$, the linear-only branch B_L, and the nonlinear-only branch B_N.

Setting	PB Case	Multiplicative	Additive	Linear-Only	Nonlinear-Only
Full encoder–decoder	$k = 0.01$	**1.65e-02**	1.97e-02	1.76e-02	3.18e−02
Full encoder–decoder	$k = 1$	**1.99e−02**	2.53e−02	2.23e−02	3.98e−02
Full encoder–decoder	$k = 100$	**2.28e−02**	2.33e−02	2.30e−02	5.98e−02
Fusion core only	$k = 0.01$	**2.08e−02**	3.45e−02	7.05e−02	9.50e−02
Fusion core only	$k = 1$	**5.47e−02**	9.12e−02	1.99e−01	1.45e−01
Fusion core only	$k = 100$	**6.84e−02**	1.42e−01	2.19e−01	1.99e−01

To further isolate the contribution of the fusion core itself, we remove both the convolutional encoder and decoder and directly map the boundary vector to the discrete solution vector through the branch modules. This no-encoder/decoder setting reduces the additional nonlinear effects introduced by the convolutional backbone and is also relevant to irregular-domain or node-based problems, where grid-based convolutional encoder–decoder modules may not be directly applicable.

The fusion-core-only rows of Table 6 provide a clearer isolation of the proposed fusion mechanism. Across all three PB regimes, the multiplicative-fusion core consistently outperforms additive fusion, the linear-only branch, and the nonlinear-only branch. The gap between $B_L \odot B_N$ and $B_L + B_N$ suggests that, in these tested settings, the nonlinear branch is more effective when used as an input-dependent modulation of the affine response path rather than as a direct additive residual correction. The results also suggest that the observed gain is not solely attributable to the auxiliary convolutional encoder–decoder modules. Rather, the multiplicative interaction is associated with improved approximation performance of the core architecture in these tested settings. These comparisons are empirical evaluations of finite-capacity trained variants under the same protocol, and should not be interpreted as a statement about the exact representability of linear PDE solution operators by affine maps. These ablations do not, however, establish a unique physical interpretation of the two learned branches.

3.6 Boundary-Amplitude Extrapolation Diagnostic

To provide a limited diagnostic of extrapolation beyond the training boundary-amplitude distribution, we further evaluated LNF-NO on out-of-distribution (OOD) boundary-amplitude samples for the source-free PB benchmark with $k = 1$. Specifically, held-out boundary traces were scaled as $g_{\mathrm{OOD}} = sg$ with $s = 1.25$ and $s = 1.50$, and the corresponding reference solutions were recomputed using the finite-difference Newton solver. Data-driven surrogate models commonly face challenges when evaluated outside the training distribution. In this context, the present boundary-amplitude test provides a representative example showing that LNF-NO retains a certain degree of extrapolation capability under moderate boundary-amplitude shifts: as shown in Table 7, the error increases under boundary-amplitude extrapolation, as expected, but remains at the same order as the in-distribution test error for the tested scaling factors. This result should still be interpreted as a limited diagnostic rather than a systematic parameter-extrapolation benchmark across PDE regimes. More comprehensive extrapolation studies, including tests over PDE parameters, coefficient contrasts, boundary-condition amplitudes, and ionic-strength regimes, as well as possible improvements through appropriate physical constraints, mechanism-informed regularization, or regularity-preserving architectures, require a separately designed study and are important directions for subsequent work.

Table 7: Limited boundary-amplitude extrapolation diagnostic for LNF-NO on the source-free PB benchmark with $k = 1$. Boundary traces from the held-out test set are scaled as $g_{\mathrm{OOD}} = sg$, and new reference solutions are recomputed using the finite-difference Newton solver. The boundary root-mean-square error (RMSE) is also reported.

Setting	Full-Field Relative ℓ_2	Boundary Relative ℓ_2	Boundary RMSE
In-distribution, $s = 1.00$	$1.9905 \times 10^{-2} \pm 6.7160 \times 10^{-3}$	$2.6860 \times 10^{-2} \pm 8.6085 \times 10^{-3}$	$2.2943 \times 10^{-1} \pm 1.4633 \times 10^{-1}$
OOD, $s = 1.25$	$2.8232 \times 10^{-2} \pm 1.1826 \times 10^{-2}$	$3.7197 \times 10^{-2} \pm 1.5622 \times 10^{-2}$	$4.2089 \times 10^{-1} \pm 2.9542 \times 10^{-1}$
OOD, $s = 1.50$	$4.2268 \times 10^{-2} \pm 2.1059 \times 10^{-2}$	$5.3113 \times 10^{-2} \pm 2.6227 \times 10^{-2}$	$7.4527 \times 10^{-1} \pm 5.4624 \times 10^{-1}$

4 Conclusion

The experimental results indicate that, within the scope of the problem settings and equations investigated, LNF-NO offers a valuable trade-off between efficiency, versatility, and accuracy. Rather than claiming superiority across all possible PDE learning tasks, we position LNF-NO as a streamlined framework that shows favorable empirical performance on the tested multi-field problems and fixed irregular-domain settings, where specialized grid-based or highly task-specific baselines may require additional adaptations.

First, LNF-NO exhibits a high degree of architectural uniformity. While grid-based spectral approaches (e.g., FNO) are inherently restricted to regular meshes and require specific adaptations for complex geometries, LNF-NO employs a unified linear–nonlinear fusion core across regular grids, fixed

irregular-node discretizations, and the tested three-dimensional setting. Although point-based baselines like DeepONet also support irregular geometries, our experiments suggest that they may face optimization difficulties in higher-dimensional or strongly nonlinear settings. In contrast, LNF-NO shows consistent empirical optimization behavior and competitive performance across these tested scenarios using a single architectural paradigm. This design naturally accommodates multiple functional inputs and coupled multi-field outputs without requiring problem-specific architectural redesign.

Second, LNF-NO consistently demonstrates significantly improved training efficiency. Under the common evaluation protocol used in this work, our method exhibits a clear advantage in training efficiency on most benchmarks. The linear–nonlinear fusion structure acts as an efficiency-oriented inductive bias and empirically leads to faster error reduction, particularly in stiff nonlinear regimes. Unlike emerging Transformer-based foundation-model-oriented approaches [10], which often emphasize model capacity and broad pretraining, LNF-NO targets an explicitly efficiency-oriented design regime. This makes it particularly suitable as a lightweight baseline for rapid exploration and analysis.

Third, regarding predictive performance, LNF-NO achieves comparable or improved accuracy in most cases relative to the baselines. It performs particularly well on strongly nonlinear Poisson–Boltzmann problems, irregular-domain benchmarks, and several multi-input settings. We also note that on certain established benchmarks, such as Burgers, discontinuous Darcy flow, the PNP system, and the 3D Poisson–Boltzmann problem, the lowest error may be attained by other specialized baselines. Even in these cases, however, LNF-NO remains competitive in accuracy while retaining clear advantages in training efficiency and architectural uniformity.

The variation across the PB regimes provides one possible interpretation of when the proposed architectural prior may be useful. As k increases from 0.01 to 100, the source-free PB problem becomes increasingly stiff and develops sharper boundary layers, while the relative accuracy advantage of LNF-NO over the compared baselines becomes more pronounced. One possible interpretation is that the multiplicative affine–nonlinear interaction is better aligned with the stronger nonlinear modulation present in these PB regimes. This interpretation is qualitatively consistent with the motivating intuition of the architecture. However, the present experiments do not establish a causal mechanism or a universal equation-dependent performance theory. By contrast, in weakly nonlinear or more generic settings, such as the $k = 0.01$ and three-dimensional PB cases, general-purpose attention- or spectral-based models can also achieve strong predictive accuracy, and the principal advantage of LNF-NO is more clearly expressed through its accuracy–training-cost trade-off.

Although LNF-NO provides a favorable accuracy–efficiency trade-off, several specialized architectures still achieve lower predictive errors on selected benchmarks, and the current framework does not yet provide dedicated mechanisms for arbitrary-geometry encoding or resolution-independent generalization. The present study does not establish a branch-wise physical interpretation or latent-space decomposition of the learned affine and nonlinear representations. In addition, the current theoretical analysis does not provide operator-specific convergence, stability, discretization-consistency, or mesh-generalization guarantees. An important direction is therefore to further improve predictive accuracy while carefully controlling the additional computational cost, for example through more expressive but still lightweight encoders and decoders, adaptive or task-dependent fusion mechanisms, and problem-specific refinements. Since the proposed linear–nonlinear fusion core is lightweight and modular, it may also be integrated with advanced neural-operator backbones or other neural-network architectures to combine complementary representation capabilities, although such hybrid designs should be evaluated with explicit attention to the resulting accuracy–efficiency trade-off. Future studies will additionally investigate more complex geometries and realistic boundary conditions, as well as higher-resolution three-dimensional simulations where balancing

accuracy, flexibility, and computational cost becomes increasingly important. Broader systematic PDE-residual, conservation, boundary-condition, perturbation-stability, and parameter-extrapolation analyses across the full benchmark suite also remain for future investigation.

Acknowledgement: Not applicable.

Funding Statement: This research was funded by the Strategic Priority Research Program of the Chinese Academy of Sciences, grant number XDB0500000, the National Key R&D Program of China, grant number 2024YFA1012101, and the National Natural Science Foundation of China, grant number 12371413.

Author Contributions: The authors confirm contribution to the paper as follows: conceptualization, Heng Wu and Benzhuo Lu; methodology, Heng Wu; software, Heng Wu and Junjie Wang; validation, Heng Wu and Junjie Wang; formal analysis, Heng Wu; investigation, Heng Wu; resources, Benzhuo Lu; data curation, Heng Wu and Junjie Wang; writing—original draft preparation, Heng Wu; writing—review and editing, Heng Wu, Junjie Wang and Benzhuo Lu; visualization, Heng Wu; supervision, Benzhuo Lu; project administration, Benzhuo Lu; funding acquisition, Benzhuo Lu. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The datasets and implementation used in this study are available from the corresponding author upon reasonable request.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Supplementary Materials: The supplementary material is available online at https://www.techscience.com/doi/10.32604/cmes.2026.084608/s1. It includes dataset generation details, training and evaluation protocols, architecture configurations, baseline model descriptions, seed stability results, and qualitative visualization results [4,6,8,9,23,28,29].

References

1. Quarteroni A, Manzoni A, Negri F. Reduced basis methods for partial differential equations: an introduction. Cham, Switzerland: Springer; 2015.
2. Benner P, Gugercin S, Willcox K. A survey of projection-based model reduction methods for parametric dynamical systems. SIAM Rev. 2015;57(4):483–531. doi:10.1137/130932715.
3. Willcox K, Peraire J. Balanced model reduction via the proper orthogonal decomposition. AIAA J. 2002;40(11):2323–30. doi:10.2514/3.15326.
4. Lu L, Jin P, Pang G, Zhang Z, Karniadakis GE. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. Nat Mach Intell. 2021;3(3):218–29. doi:10.1038/s42256-021-00302-5.
5. Kovachki N, Li Z, Liu B, Azizzadenesheli K, Bhattacharya K, Stuart A, et al. Neural operator: learning maps between function spaces. J Mach Learn Res. 2023;24(89):1–97. doi: 10.48550/arxiv.2108.08481.
6. Li Z, Kovachki N, Azizzadenesheli K, Liu B, Bhattacharya K, Stuart A, et al. Fourier neural operator for parametric partial differential equations. In: Proceedings of the International Conference on Learning Representations; 2021 May 3–7; Virtual Conference.
7. Li Z, Kovachki N, Choy C, Li B, Kossaifi J, Otta S, et al. Geometry-informed neural operator for large-scale 3D PDEs. In: Proceedings of the Thirty-Seventh Conference on Neural Information Processing Systems; 2023 Dec 10–16; New Orleans, LA, USA. Vol. 36, p. 35836–54.
8. Rahman MA, Ross ZE, Azizzadenesheli K. U-NO: u-shaped neural operators. Trans Mach Learn Res. 2023.
9. Wu H, Luo H, Wang H, Wang J, Long M. Transolver: a fast transformer solver for PDEs on general geometries. In: Proceedings of the 41st International Conference on Machine Learning; 2024 Jul 21–27; Vienna, Austria. Cambridge, MA, USA: PMLR; 2024. p. 53681–705.

10. Ye Z, Huang X, Chen L, Liu H, Wang Z, Dong B. PDEformer: towards a foundation model for one-dimensional partial differential equations. In: Proceedings of the ICLR 2024 Workshop on AI4 Differential Equations in Science; 2024 May 11; Vienna, Austria.
11. Hao Z, Wang Z, Su H, Ying C, Dong Y, Liu S, et al. GNOT: a general neural operator transformer for operator learning. In: Proceedings of the 40th International Conference on Machine Learning; 2023 Jul 23–29; Honolulu, HI, USA. Cambridge, MA, USA: PMLR; 2023. p. 12556–69.
12. Liu Q, Zhong W, Meidani H, Abueidda D, Koric S, Geubelle P. Geometry-informed neural operator transformer for partial differential equations on arbitrary geometries. Comput Methods Appl Mech Eng. 2026;451(5):118668. doi:10.1016/j.cma.2025.118668.
13. Liu Q, Koric S, Abueidda D, Meidani H, Geubelle P. Toward signed distance function based metamaterial design: neural operator transformer for forward prediction and diffusion model for inverse design. Comput Methods Appl Mech Eng. 2025;446(3):118316. doi:10.1016/j.cma.2025.118316.
14. Wu H, Hu T, Luo H, Wang J, Long M. Solving high-dimensional PDEs with latent spectral models. In: Proceedings of the 40th International Conference on Machine Learning; 2023 Jul 23–29; Honolulu, HI, USA. Cambridge, MA, USA: PMLR; 2023. p. 37417–38.
15. Liu S, Hao Z, Ying C, Su H, Cheng Z, Zhu J. NUNO: a general framework for learning parametric PDEs with non-uniform data. In: Proceedings of the 40th International Conference on Machine Learning; 2023 Jul 23–29; Honolulu, HI, USA. Cambridge, MA, USA: PMLR; 2023. p. 21658–71.
16. He J, Liu X, Xu J. MgNO: efficient parameterization of linear operators via multigrid. In: Proceedings of the Twelfth International Conference on Learning Representations; 2024 May 7–11; Vienna, Austria.
17. Xiao Z, Hao Z, Lin B, Deng Z, Su H. Improved operator learning by orthogonal attention. In: Proceedings of the 41st International Conference on Machine Learning; 2024 Jul 21–27; Vienna, Austria. Cambridge, MA, USA: PMLR; 2024. p. 54288–99.
18. Helwig J, Zhang X, Fu C, Kurtin J, Wojtowytsch S, Ji S. Group equivariant fourier neural operators for partial differential equations. In: Proceedings of the 40th International Conference on Machine Learning; 2023 Jul 23–29; Honolulu, HI, USA. Cambridge, MA, USA: PMLR; 2023. p. 12907–30.
19. Perez E, Strub F, de Vries H, Dumoulin V, Courville A. FiLM: visual reasoning with a general conditioning layer. In: Proceedings of the AAAI Conference on Artificial Intelligence; 2018 Feb 2–7; New Orleans, LA, USA.
20. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. Adv Neural Inf Process Syst. 2017;30:6000–10. doi:10.65215/ctdc8e75.
21. Shazeer N, Mirhoseini A, Maziarz K, Davis A, Le QV, Hinton GE, et al. Outrageously large neural networks: the sparsely-gated mixture-of-experts layer. In: Proceedings of the International Conference on Learning Representations; 2017 Apr 24–26; Toulon, France.
22. Lu B, Zhou Y, Holst MJ, McCammon JA. Recent progress in numerical methods for the Poisson-Boltzmann equation in biophysical applications. Commun Comput Phys. 2008;3(5):973–1009. doi:10.4208/cicp.2008.v3.p973.
23. Zhang Q, Gui S, Li H, Lu B. Model reduction-based initialization methods for solving the Poisson–Nernst–Planck equations in three-dimensional ion channel simulations. J Comput Phys. 2020;419(2):109627. doi:10.1016/j.jcp.2020.109627.
24. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2016 Jun 27–30; Las Vegas, NV, USA. p. 770–8.
25. Cybenko G. Approximation by superpositions of a sigmoidal function. Mathem Cont Sig Syst. 1989;2(4):303–14. doi:10.1007/bf02551274.
26. Hornik K. Approximation capabilities of multilayer feedforward networks. Neural Netw. 1991;4(2):251–7. doi:10.1016/0893-6080(91)90009-t.
27. Loshchilov I, Hutter F. Decoupled weight decay regularization. In: Proceedings of the International Conference on Learning Representations; 2019 May 6–9; New Orleans, LA, USA.
28. Wu H, Lu B. Mathematical artificial data for operator learning. arXiv:2507.06752. 2025.
29. Jin P, Meng S, Lu L. MIONet: learning multiple-input operators via tensor product. SIAM J Sci Comput. 2022;44(6):A3490–514. doi:10.1137/22m1477751.