EUROPEAN JOURNAL OF PURE AND APPLIED MATHEMATICS
2026, Vol. 19, Issue 1, Article Number 6334
ISSN 1307-5543 – ejpam.com
Published by New York Business Global

Scientific Evolution of Physics-Informed Neural Networks: A Comprehensive Review of Recent Architectural Variants and Optimization Strategies

Ilyas Khan[1,*], Ahmad[2], Husna Zafar[2], Aneeqa Zafar[2], Muhammad Jawad[2], Mohd Anul Haq[3], Abdulaziz Saeed Alqahtani[3], Wei Sin Koh[4]

[1] *Department of Mathematics, College of Science Al-Zulfi, Majmaah University, Al-Majmaah 11952, Saudi Arabia*
[2] *School of Mathematics, Hohai University, Nanjing 210098, Jiangsu, China*
[3] *Department of Information Technology, College of Computer and Information Sciences, Majmaah University, Al Majmaah 11952, Saudi Arabia*
[4] *INTI International University, Persiaran Perdana BBN Putra Nilai, 71800 Nilai, Negeri Sembilan, Malaysia*

Abstract. Physics-Informed Neural Networks (PINNs) are a machine learning technique that directly incorporates the governing physics of problems, such as partial differential equations (PDEs) and ordinary differential equations (ODEs), into the neural network architecture. The primary goal of PINNs is to approximate solutions while satisfying given constraints and minimizing the residuals of the differential equations. PINNs have been employed to solve various problems, including integro-differential equations, fractional differential equations, and stochastic PDEs. Over the past two years, significant advancements have addressed the challenges associated with PINNs, resulting in notable improvements in accuracy and performance. This review provides a comprehensive summary of the latest methodologies contributing to these advancements, focusing on innovations in hyperparameter optimization and novel PINN variants inspired by other neural networks. Examples include MultiInNet-PINN, Transformer-based PINNs such as Tr-PINN and PINNsFormer, as well as PINNs incorporating attention mechanisms and recurrent neural network (RNN) architectures (PIANN). Additionally, this review highlights recent research on domain decomposition techniques in PINN architectures.By consolidating recent architectural and algorithmic advances, this review identifies critical research opportunities for enhancing the reliability, efficiency, and broader applicability of PINNs in scientific computing.

2020 Mathematics Subject Classifications: 65M75, 68T07, 35R30

Key Words and Phrases: Physics-informed neural networks, smart microgrid, pinn, deep learning, mathematical modeling, pdes, recent innovations

*Corresponding author.
DOI: https://doi.org/10.29020/nybg.ejpam.v19i1.6334

Email addresses: i.said@mu.edu.sa (I. Khan)

 Copyright: © 2026 The Author(s). (CC BY-NC 4.0)

1. Introduction

With time, the world is advancing towards sophistication to comply with the contemporary requirements of handling complex tasks with higher output efficiency in various fields of technology. Keeping in view the myriad applications of conventional mathematical models in several fields of science and technology, deep learning techniques have evolved immensely as a powerful tool to capture complex phenomena. By employing different activation functions in each hidden layer in the ANN, an accurate and efficient approximations of nonlinear complex functions can be obtained. These achievements inspire many researchers to solve nonlinear PDEs which generally don't have an exact solution [1]. Some of the earliest research reported in literature where ANN has been employed to solve differential equations is the research of I.E. Lagaris et al. [2] in which differential equations are solved with the use of a trial function based on the initial/boundary conditions of the governing equation. This trial function becomes challenging to construct for high-dimensional complex PDEs [3]. To overcome this challenge, Raissi et al. [4] introduced a Physics-informed neural network (PINN). PINN is a deep learning framework in which an appropriate loss function is constructed for incorporating all the physical information into the neural network described by governing PDEs with initial and boundary conditions [5]. The main idea behind PINNs is that the NN satisfies the given constraints and approximates the solution in a way that minimizes the loss function [6]. The great advantage of PINN lies in its adaptability which means it can handle forward problems without the presence of any data when the given equations are fully known and utilize existing data to handle inverse problems even in the absence of model parameters or other physical quantities [7]. In simple words, PINNs fill the gap between the scarcity of the data and the data-intensive nature of deep neural networks [8] and have successfully been applied to many physical domains including heat transfer problems [**?**], fluid mechanics [9], and biomedical engineering [10, 11] etc. Even after remarkable success in recent years, there are still some challenges that PINN faces in solving complex problems such as the calculation of high-order differential terms makes the training approach time-consuming [12]. It is easy for the PINN to fall into local optima particularly the gradient pathologies, in solving high-frequency problems because of the presence of multiple terms in the loss function [13]. An appropriate choice of network architecture and its hyperparameters is crucial for attaining accurate results in PINN. However, it is difficult to choose a suitable combination of hyperparameters that yields strong generalization performance because of the extensive number and broad range of hyperparameters [14]. Moreover, high-order PDEs require an increasing depth of neural network which causes a slow learning rate because of vanishing gradients [15]. It is crucial to optimize the network in such a way that it produces accurate results with low computational cost and less running time. If the network's prediction is not accurate, it will cause an increase in loss function which will eventually transverse the learning path that reduces that loss [16]. In the context of a Physics-Informed neural network, an accuracy is generally defined by how well our network model aligns with underlying physical laws governed by differential equations and the precision ensures robustness of the model against noise and uncertainties in the data.

Researchers used different error metrics to assess the computational efficiency and the accuracy of the designed model which includes root mean square error (RMSE), mean absolute error, relative L2 error, and so on. Using these metrics, we can analyze the results and conclude the feasibility of utilizing physics-informed neural networks (PINNs) for intricate dynamical systems specifically about their ability to produce computationally efficient results[17]. To enhance the adaptability of PINN to specific problems along with its accuracy and generalization capabilities, continuous improvements and optimizations in its architecture are essential. Continuous modifications can bring about significant improvements in PINN's performance, aligning them to play a more crucial role in the broader spectrum of scientific and engineering applications. Researchers are continuously doing modifications in PINN architectures by using different hyperparameter optimization techniques , domain decomposition strategies and hybrid network architectures to get accurate results for better interpretation of complex science, engineering, and biomedical-related problems. In this review, we have conducted an in-depth analysis of different techniques to find optimal hyperparameters, innovations based on domain decomposition strategies and variants of PINNs recently reported in the literature. Our objective is not only to present the current state of research in this domain but also to provide valuable insights into key pathways for enhancing precision and accuracy in PINNs. This review intends to assist researchers in effectively applying PINN in different domains, offering practical guidance for improved utilization. We aim to contribute to the ongoing advancement of physics-informed neural networks by exploring the emerging trends and the methodologies that lead to excellence and innovation in the adaptability of PINN. This review aims to systematically evaluate the advancements in PINNs, particularly focusing on their application in solving high-dimensional PDEs and enhancing computational efficiency. Specifically, we aim to address the following key questions:

1. What are the state-of-the-art techniques to optimize hyperparameters of PINN, and how do they influence the model's performance?
2. How do domain-decomposition strategies enhance PINN's accuracy and efficiency for solving complex high-dimensional PDEs?
3. What are the most emerging variants of PINN, and how do they overcome the limitations of conventional PINN architecture?
4. In what ways can the flexibility of PINNs be improved to better address the broader range of scientific and engineering problems?

By addressing these questions, we aim to present a comprehensive analysis of the current state of PINN research and highlight the key areas for future development in this advancing field.

2. Physics-Informed Neural Networks:

In this section, we provide an overview of the basic concepts related to PINN. The subsequent sections entail the advances and limitations in PINNs as well as different architectural designs, learning strategies, and optimization methods used in recent research.

2.1. Background:

A physics-informed neural network is a deep-learning technique in which NN is trained to approximate the solution by incorporating all the physics into the loss function. This loss function is formulated based on terms present in the governing PDE such as initial/boundary conditions and residual of the governing equation. The crux of PINN is that it approximates the solution that satisfies the given constraints so that the loss function gets minimized. The PINN algorithm is a mesh-free technique that transforms the original equation into a loss optimization problem and generates the solution. The earliest research in which artificial neural networks were employed to solve differential equations includes the work of Isaac Lagaris et al. [2] in 1998. This approach assumes a trial function as a solution that consists of the sum of two independent terms, one for initial and boundary conditions and the other for neural network approximation. In 2011 Ladislav Zjavka [18] proposed a Differential Polynomial Neural network (D-PNN) which allows each neuron to contribute to the network's output, unlike ANN. Later, In 2017, Raissi et al. [19] proposed hidden physics models, a machine learning technique to extract patterns from high-dimensional data, incorporating underlying physics. Based on this, Raissi, Perdikaris, and Karniadakis [20] proposed Physics-informed neural networks that integrate the physical laws into neural networks for supervised learning tasks including nonlinear PDEs. Discrete and continuous time models were developed based on the availability of the data type. After that, he advanced his work to multistep neural networks [21] merging numerical techniques with deep learning to study nonlinear dynamical systems. He also proposed Deep-Hidden Physics Models [22] to solve nonlinear PDEs using deep learning which further enhances the ability to learn and predict complex systems more efficiently. In 2019, Raissi et al. [4] proposed a full version of PINN to solve both forward and inverse problems.

2.2. Basic Concepts of PINN:

Consider the general mathematical representation of PDE as:

$$f(x,t,u^{'},u^{'}_{x},u^{'}_{t},\lambda)=0 \tag{2.2.1}$$

$$u^{'}(x,t_0)=\beta_0, x\in\phi \tag{2.2.2}$$

$$u^{'}(x,t)=\beta_{\ulcorner}(t), x\in\partial\phi \tag{2.2.3}$$

Where $u^{'}$ represents the PDE solution at x and t coordinates with given initial and boundary conditions (could be Dirichlet, Neumann, mixed one). f and λ show residual of PDE and trainable parameters, respectively.
In Vanilla PINN [4], a fully connected feed-forward NN with multiple hidden layers to approximate the solution of the PDE $u^{'}(x,t)$ in which (x,t) embeds as an input into the neural network. Suppose ω^k is a hidden variable in a k^{th} hidden layer, the flow of the network is represented as follows:

$$\omega^0 = (x, t)$$
$$\omega^k = \varphi(W^k M^{k-1} + b^k),\ for\ 1 \leq k \leq L-1,$$
$$\omega^L = W^L M^L + b^L$$

The last (output) layer ω^L is used to approximate the solution i.e, $\omega^L \approx u^{'}$. The linear combination of weight and bias matrix of the k^{th} layer (W^k and b^k respectively), are passing through a nonlinear activation function φ. The governing PDE in eq 2.2.1 along with its initial and boundary conditions is transformed into a loss optimization problem in which loss L gets minimized by iteratively updating tuneable parameters θ. The loss function L is defined as:

$$L = \xi_1 L_{\text{PDE}} + \xi_2 L_{\text{data}} + \xi_3 L_{\text{IC}} + \xi_4 L_{\text{BC}}$$

Where, L_{PDE} penalizes the PDE's residual and the remaining terms L_{data}, L_{IC}, L_{BC} ensure the prediction of the model satisfies data points, initial, and boundary conditions [**?**]. ξ_i, $i = 1, 2, 3, 4$, are the weight coefficients corresponding to each loss term. Generally, the mean square error (MSE) computes the losses in equation (4) and ADAM [23] is employed as an optimizer to update the trainable parameters θ. The loss function is problem-dependent, meaning some terms may be canceled based on the problem. Automatic Differentiation is employed to compute the partial derivatives present in the loss function. The detailed architecture of PINN is shown in Figure 1.

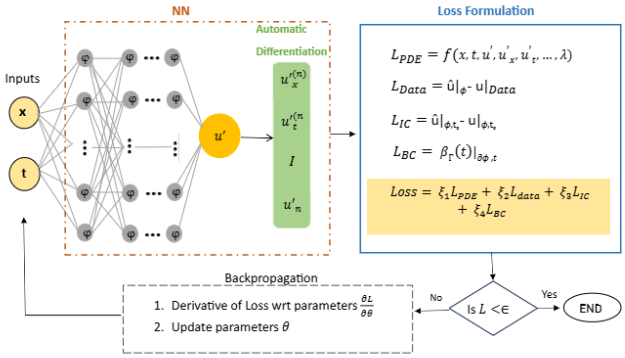

Figure 1: Framework of PINN

To compare the performance of PINNs with ANN, Son Seho et al. [24] experimented using an equation of motion of a 1-DOF mass-spring-damper system by setting the same architecture for both networks except for the learning rate. Table 1 below shows the hyperparameters used for training PINN and ANN.

Figure 2 compares the exact solution, represented by a blue line, and the predicted solution obtained by PINN and ANN. Figure 2(a) shows that the ANN only predicts the solution (red dashed line) for points (green marks) used in the training, and fails to predict

	PINN	**ANN**
Input Nodes	1	1
Hidden Layers	3	3
Hidden Nodes	35	35
Activation Function	Tanh	Tanh
Output Nodes	1	1
Learning Rate	0.0001	0.001
Epoch	18,000	18,000

Table 1: Comparison between PINN and ANN

the solution for other data points. This limitation underscores the need for PINN to solve engineering problems requiring large training data. In contrast, PINN also predicts the solution in regions outside the training data, demonstrating better generalization ability than ANN.

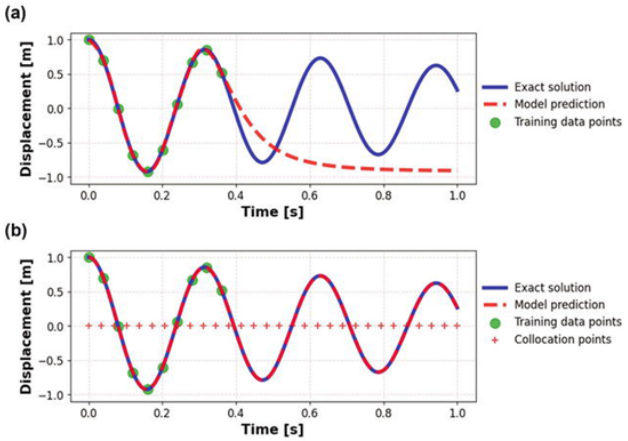

Figure 2: Comparison between exact and predicted solution by (a) ANN and (b) PINN [24]

3. Significance of Accuracy and Precision in PINN:

The significance of accuracy and precision cannot be overlooked in the realm of physics-informed neural networks. These two are the fundamental aspects to critically analyze the performance and reliability of any predictive model. Precision assures the predictions of the PINN model to be consistent and robust which means it produces stable and reliable results regardless of the variations in data inputs and scenarios being considered. Meanwhile, accuracy evaluates how well our model predicts the solution of the problem as compared to its true solution highlighting its ability to capture all the true physics of the governing differential equation. In other words, accuracy ensures the predictions made by the PINN model match real-world observations. There are different ways to check the

precision and accuracy in PINNs which include quantitative metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Absolute Error, etc. These metrics calculate the average magnitude of the errors. Also, we can compare PINN's predictions with present baseline methods to calculate the relative error. In summary, precision and accuracy play a vital role in analyzing the outcomes because wrong predictions lead to false interpretations of real-world phenomena. Therefore, the exploration of strategies to enhance the precision and accuracy of PINNs is not only essential but also imperative to unveil its full potential to study problems across various fields of science and technology.

4. Limitations of PINNs:

While PINNs have gained significant popularity for their ability to produce promising outcomes across various phenomena, the initial formulation of PINN proposed by Raissi [4] often encounters many challenges in constructing an accurate approximation of the exact solution. A common reason for failure in PINNs arises from the unbalanced gradients during the process of backpropagation. This challenge persists even in simple settings such as solving linear elliptic equations [25]. This imbalance may stem from disparities in gradients across different parts of the neural network, which ultimately leads the model to return inaccurate predictions. Additionally, soft regularization in physics-informed neural networks, which typically incorporates differential operator based on partial differential equation (PDE) into the loss function, can give rise to many problems, potentially causing the problem to become more ill-conditioned. Importantly, some of the limitations observed in PINN are not due to the inability of the neural network's architecture to handle complex functions well, but rather the inherent complexities associated with optimizing the loss landscape within the PINN framework [26]. The ill-conditioned differential operators in the residual term make the loss function challenging to optimize. Moreover, PINN also faces difficulties in effectively capturing the simulations of those dynamical systems that possess turbulent, multi-scale, or chaotic solutions [27]. These systems often exhibit complex spatial-temporal dynamics and nonlinear phenomena which need to be analyzed accurately. To overcome these challenges, the development of innovative approaches to PINN's architecture, including novel training strategies, designing new activation functions, and efficient structures for the loss function is highly needed. In summary, these limitations emphasize the importance of addressing the difficulties in the development of PINN to unveil its full potential. In the following sections, we will explore the potential solutions to these challenges, along with recent advancements in PINN architectures, training strategies, and optimization techniques aimed at enhancing the accuracy and returning accurate predictions to the underlying problem.

5. Recent Innovations in Physics-Informed Neural Networks:

Researchers recently innovated Physics-Informed Neural networks to amplify their accuracy and overcome training challenges. This section overviews some recent modifications in PINNs, including innovative architectural considerations via hybrid PINN models, differ-

ent domain decomposition strategies, and optimal tuning of hyperparameters. A detailed analysis of these advancements will give readers valuable insights into PINN research to revolutionize computational modeling and simulation in various fields of science and technology.

6. Innovations in PINN for Optimal Architecture:

The selection of appropriate hyperparameters in PINN such as no. of hidden layers with no. of neurons in each layer, activation function, learning rate, and optimizer, is crucial in improving the prediction of solution. However, finding the optimal hyperparameters within a high-dimensional search space is challenging, and slow training remains a concern in PINNs [28, 29]. To address these challenges, hyperparameter optimization (HPO), also known as (HP) tuning [30], offers a solution. While many researchers have employed grid search (resp. random) [30] to provide flexibility in choosing hyperparameters, these approaches suffer from a curse of dimensionality and become inefficient with a random search. To explore the optimal network parameters of PINN, there are few studies present in the literature that involve tuning hyperparameters. Gaussian processes-based Bayesian optimization via HPO tuning has been used to tune PINNs hyperparameters for the Helmholtz operator [31]. This method used GP-based regression on previously evaluated hyperparameters to predict the next most likely optimal configuration. The same approach was used to optimize stochastic PINN for advection-diffusion reaction problems [32]. Manual tuning of hyperparameters to solve discontinuous heat conduction problems is also employed in [33]. In [34], the authors used a genetic algorithm to determine the most appropriate network design and optimization function for dynamical systems. Later, Wang et al. [5] proposed NAS-PINN by incorporating the NAS algorithm into PINN to search for optimal hyperparameters. NAS is an algorithm that aids in finding optimal architecture in specific search spaces [35]. NAS-PINN is trained using bi-level optimization with the inner one focusing on optimizing weights and biases of the NN while the outer one optimizes the parameters of network architecture. Auto-PINN proposed by Yicheng Wang et al. [36] utilizes the step-by-step decoupling strategy to reduce the search space by fixing other hyperparameters and searching for one at a time for a few sets of options. A generally efficient and fully automatic method to optimize PINN architecture was presented by Kaplarević-Mališić, Ana, et al. [14]. The authors proposed a PINN/GA framework which is based on a Genetic Algorithm (GA). This framework starts with the initial population of candidate solutions, called individuals, and evolves the population through selection, crossover, and mutation. It keeps the search space computationally manageable by evolutionary strategy from simple to more complex network architectures, as presented in [37] while leveraging the scalable computational design to handle large-scale optimization efficiently. Table 2 highlights the summary of research studied in this section.

Architecture	**Problem Type**	**PINN Optimal Configuration**	**Performance Metrics and Computational Cost**	**Comparison**
Neural Architecture Search [5]	2D Poisson Equation (Regular)	Depth: 4 Width: [50, 70, 1]	L2 Error: 4.46×10^4 -Time to find optimal architecture = 1.57 hours	-take less time to find optimal hyperparameters compared to SMAC NAS-PINN can search in a larger and more flexible search space compared to SMAC.
	2D Poisson Equation (Irregular: Circle, L, and flower shaped)	Circle:[2, 110, (50, 50, 50, 30,) 1] L-shaped: [2, 110, 110, (10,) 1] Flower:[2, 50, 70, (70, 70,) 70, 110, 1]	L2 Error Circle: 2.25×10^{-7} L-shaped: 2.05×10^{-6} Flower: 6.91×10^{-6}	Less parameters compared to Dumpy Architecture -Outperformed Giant, Dumpy, and Slender
	Burger's	[2, 90, 50, 110, 1] for v=0.1 [2, 90, 70, 30, 110, 1] for v=0.07 [2, 110, 110, 70, 110, 1] for v=0.04	L2 error $8.87 \times 10^{-7}(for\ v = 0.1)$ $1.41 \times 10^{-6}(for\ v = 0.07)$ $1.51 \times 10^{-6}(for\ v = 0.04)$	Outperformed all manually designed architectures like Giant, Dumpy, and Slender
	Advection's	[2, 110, 110, 1] for $\beta = 1$ [2, 90, 90, 90, 90, 110, 1] for $\beta = 0.4$ [2, 110, 50, 50, 70, 30, 90, 1] for $\beta = 0.1$	L2 error $1.49 \times 10^{-4} for\ \beta = 1$ $1.30 \times 10^{-6} for\ \beta = 0.4$ $2.56 \times 10^{-6} for\ \beta = 0.1$	NAS-PINN shows similar performance with Dumpy architecture (2 hidden layers with 110 neurons) for $\beta = 1$ -NAS-PINN demonstrates the most consistent high performance across all β values

Architecture	**Problem Type**	**PINN Optimal Configuration**	**Performance Metrics and Computational Cost**	**Comparison**
PINN/Genetic Algorithm [14]	1D Stefan Problem with time-dependent Dirichlet BC's	Hidden layers=2, no.of neurons per layer =[3,22], Optimizer=NADAM, Activation=Tanh	Achieved the lowest Mean Squared Error (MSE) compared to Random Search and Hyperopt for both PDEs	Random Search demonstrated good solutions in the early stages but lacked a systematic refinement process, whereas Hyperopt TPE achieved rapid initial progress but exhibited stagnation in optimization
	2D Helmholtz Equation	Hidden Layers=5, neurons per hidden layer= [30,30,30,30,30], Activation=sin, Optimizer=NADAM	Optimization took approximately $1-2$ hours to converge for both cases	PINN/GA consistently showed best results, steadily improving accuracy and stability over time compared to Random Search and Hyperopt TPE.
GP-based Bayesian HPO [31]	Helmholtz (Dirichlet)	Learning rate $= 10^{-4}$ Width = 275 Depth = 2 Activation = sin	Training Loss: 1.13×10^{-3} Testing Error: 1.19×10^{-4} Training Time: 196.6s Parameters: 77,001	-Achieved 10x lower error than FEM for low-frequency Helmholtz problems (e.g., $\omega = 2$). -PINNs training was 100x slower than FEM.
	Helmholtz (Neumann)	Learning rate $= 1.54 \times 10^{-4}$ Width = 10 Depth = 292 Activation = sin Weights for boundary term error = 19.1	Training Time: 9.3×10^2 Parameters: 1.7×10^5	-Higher metric loss compared to FEM. -PINNs are slower; FEM (LU solver) completes in 1.2×10^2 seconds (approx. 2 minutes), and GMRES is even faster (3.9 seconds)

Architecture	**Problem Type**	**PINN Optimal Configuration**	**Performance Metrics and Computational Cost**	**Comparison**
Manual hyperparameters tuning [33]	Discontinuous heat conduction problems (2D/3D)	Architecture = FCNN, DGM Depth = 20 Activation: SiLU, GELU, Tanh, Sin Learning Rates: 7.5×10^{-3}, 5×10^{-3}, 2.5×10^{-3}, 10^{-3}, 7.5×10^{-4}	Mean Rel. L2 Error: 5.60% Max Rel. Error: 13.9%	-FCNN outperformed MFNN and DGM. -SiLU, GELU, Tanh, Sin (consistent accuracy) compared to ReLU and SELU. -Moderate rates (10^{-3} to 7.5×10^{-4}) offer optimized convergence than large learning rates (10^{-2}). -Increased depth (up to 20 layers) improved accuracy but required more iterations
Auto-PINN [36]	Heat Equation with Dirichlet BCs	Width: 80–512, Depth: 5–8, Activation: Swish or Tanh, Changing Point: 0.4–0.5	Not explicitly mentioned in the study for each configuration but it consistently achieved lowest l2 errors across different sampling techniques	-Uniform sampling is generally more effective, regardless of the search method. -Moderate learning rates (lr = 1e-4) paired with sufficient training epochs (15,000) produce the best results -Optimal architectures generally have high width (384–512) and moderate-to-high depth (6–10) -Lower widths and depths fail to capture sufficient complexity. In contrast, higher learning rates lead to instability in training.

Architecture	**Problem Type**	**PINN Optimal Configuration**	**Performance Metrics and Computational Cost**	**Comparison**
	Heat Equation with Neumann BCs	Width: 232–252, Depth: 8–10, Activation: Tanh, Changing Point: 0.4–0.5	Not explicitly mentioned	-Random sampling strategies are more effective in demonstrating the strong adaptability of Auto-PINN compared to uniform sampling techniques. - Networks with greater width ($>$ 160 units) and depth ($>$ 6 layers) exhibited better testing error values
	Wave Equation	Width:40-180, Depth=3-7, Activation=Swish or Tanh, Changing Point: 0.4-0.5	Not Mentioned in the study	-Moderate network configurations tend to produce optimal performance, contrasting with the Heat equations. -Most effective architectures were identified with widths of 160–224 units and depths of 4–6 layers, highlighting that shallower networks can efficiently model wave dynamics. -Uniform sampling strategies yield better performance than random samplings -Auto-PINN precision in finding optimal architectures outperforms both HyperOpt and Random Search.

Table 2: Performance Comparison of Hyperparameter Tuning in Recent PINN Architecture

7. Innovations in PINN via Other Neural Networks:

Inspired by residual networks, Yan, Liangliang, et al. [38] proposed a multi-input residual network to modify the PINNs backbone network, named MultiInNet PINNs. The multi-step training with fixed parameters in MultiInNet PINNs greatly improves the stability and convergence speed of traditional PINNs.Traditional PINN sometimes fails to capture the temporal dependencies present in physical systems. To overcome this issue, Transformer based PINN (PINNsFormer) was proposed by Zhiyuan Zhao et al. [39]. PINNsFormer uses the multi-head attention-based mechanism which transforms the inputs and PINN loss from pointwise to sequential, making it able to capture these dependencies and hence outperform traditional PINN in dealing with failure modes of high dimensional PDEs. Tr-PINN [40] is proposed by combining the transformer model with PINN architecture. It uses a series of transformer blocks with a self-attention mechanism to capture the non-linearity features and also includes an additional input vector for mobility ratio. This model uses only the encoder part of transformer blocks and weighs the previous transformer block's output to increase the computational efficiency. A new modification of PINN, combining the recurrent neural networks (RNNs) and attention mechanisms, is proposed and termed as Physics-informed attentional-based neural network (PIANN) [41]. The attention mechanism captures the nonlinear features of the solution while the encoder-decoder GRU-based network extracts the most relevant information from fully encoded data. Hence, the model integrates gated recurrent units (GRUs) with attention mechanisms. Multilayer Perceptrons (MLPs) also called fully connected FFNN, serve as a building block for neural network architecture. Recently, Liu, Ziming, et al. [42] proposed Kolmogorov-Arnold Networks (KANs), inspired by Kolmogorov-Arnold representation theorem. KAN showed better performance and efficiency with few parameters compared to MLPs, primarily due to their learnable activation functions. Wang, Yizheng, et al. [43] proposed Kolmogorov-Arnold-Informed Neural Network (KINN), a KAN variant designed for different forms of PDEs including strong, energy, and weak forms, to compare the performance of KAN with MLP across various PDEs. The outperformance of KAN unlocks the potential to modify MLP-based PINN. Recently, Rigas, Spyros, et al. [44] developed a fast implementation of Physics-Informed Kolmogorov-Arnold Networks (PIKANs) using JAX called jaxKAN which is also available on PYPI as a Python package. PIKANs, a new type of PINN, with KAN as its foundation block for network architecture. Sun et al. [**?**] introduced a new architecture based on LSTM networks, termed as physical-informed memory networks (PIMNs) which differ from fully-connected PINN architecture. PIMNs used difference schemes as convolutional filter to approximate the differential operator. This convolutional structure is only to speed up the computations and doesn't have an impact on the training process. In 2019, Lu et al. [45] proposed a new network architecture inspired by the universal approximation theorem to operators [46], called DeepONets. Two DNNs, branch and trunk networks, are employed in the DeepONets architecture. The branch one encodes the input functions at fixed sensors (i.e., function values at different locations), while the trunk encodes the location information of the output function. The solution operator is then defined as the element-wise mul-

tiplication of outputs of branch and trunk networks (i.e., coefficients and basis functions respectively), summed over the total output values. However, DeepONet needs a large corpus of paired input/output observations obtained through multiple repeated experiments, making the training dataset computationally expensive. An alternative version, inspired by PINN, is PI-DeepONet [47] which demonstrates that DeepONets' outputs are differentiable wrt. input coordinates, allowing for the use of automatic differentiation to compute the derivatives present in underlying PDE. PI-DeepONet greatly enhances the generalization capabilities of PINN. Several works have focused on optimizing the computations needed to compute the derivatives which include: zero-coordinate shift to optimize reverse-mode automatic differentiation [48] and the use of factorizable coordinates to optimize forward-mode AD [49]. Based on pioneering work done for separable-PINN in [49], Mandl, Luis, et al. [50] proposed separable-PI-DeepONet. This approach leverages both separable PINN and PI-DeepONet methodologies by employing the factorized coordinates and separable subnetwork for each 1D coordinate to reduce the computational cost, using a forward-mode AD for efficient Jacobian matrix computation, and combining the outputs of branch and trunk network to further optimize the forward pass. Table 3 and 4 below summarizes the publications studied in this section.

Architecture	**Problem Type**	**ERR**†	**Key Advantages**
MultiNet [38]	Poisson Advection Convection-diffusion Helmholtz	97.0% (MSE) 95.1% (MSE) 73.9% (MSE) 68.0% (MSE)	- Fewer parameters (57% reduction) - Better convergence stability - Enhanced computational efficiency
PINNsFormer [39]	Navier-Stokes 1D-Wave Convection 1D-Reaction	97.1% (rMAE) 83.4% (rMAE) 97.0% (rMAE) 98.5% (rMAE)	- Superior in complex fluid flows - NTK enhancement compatibility - Stable training dynamics - Handles stiff problems well
PIANN [41]	Hyperbolic PDEs (Buckley–Leverett problem)	Residual decreased by$< 10^{-4}$	- Captures shock fronts without residual regularization or prior knowledge
PIKANs [44]	Diffusion Helmholtz Burger's Allen–Cahn	0.021% (rL2) 0.176% (rL2) 2.340% (rL2) 1.414% (rL2)	- Faster training with adaptive methods and efficient implementation - Scalable to smaller networks while maintaining performance
Physics-informed DeepONets [47]	1D parametric ODE Diffusion reaction Burger's	$\sim$0.33% (rL2) $\sim$0.45% (rL2) $\sim$1.38% (rL2)	- Accurate predictions for out-of-distribution test data - More accurate predictions with paired input-output training data, unlike conventional DeepONet - Rapid prediction of spatiotemporal solutions ($>$ 10 ms) and up to 3 orders of magnitude faster than conventional solvers
Separable PINN*[49]	Diffusion Helmholtz (2+1)-d Klein–Gordon (3+1)D Klein-Gordan	12.2% (rL2) 93.7% (rL2) 94.9% (rL2) 90.1% (rL2)	- Memory reduction - Less computational time
Separable PI-DeepONets [50]	Burgers Consolidation Biot's Parametrized Heat Poisson's	no significant reduction in relative L2 error compared to [50] for all test cases	- Significant runtime improvement - Scalable to higher dimensions - Maintains accuracy with less computations
PIMNs using LSTM [**?**]	1-soliton Schrodinger 2-soliton Schrodinger KdV-Burger's KdV Burgers Kuramoto	77.8% 86.1% (rL2) 97.7% (rl2) 0.021% (rL2)	- Enhanced Accuracy - Works well with high-gradient regions

Table 3: Comparison of various PINN architectures, their problem types, error metrics, and key advantages.

ERR† shows the error reduction rate compared to baseline Vanilla PINN performance except for [50] and [46]. $ERR = \frac{E_1 - E_2}{E1} \times 100$, where E_1 is error in baseline method and E_2 is error of newly proposed framework.
† Error metrics vary by study: MSE (Mean Square Error), rMAE (relative Mean Absolute Error), mrL_2 (mean relative L_2 error), and rL_2 (relative L_2 error).
*The values of error reduction are calculated by comparing the best results of PINN and SPINN-based architectures.

Architecture	**Key Features**	**Best Suited For**	**Limitations**
MultiInNet [38]	- Multiple independent networks - Parameter efficient - Parallel training possible	- Problems requiring domain decomposition - Multi-scale physics	- Higher memory requirements - Complex implementation
PINNsFormer [39]	- Transformer-based architecture - Self-attention mechanisms - Compatible with NTK enhancement	- Complex fluid dynamics - Problems with long-range dependencies	- Training time overhead - Requires careful hyperparameter tuning
PIANN [41]	- Attention mechanism detects shock locations automatically - GRU-based encoder-decoder for non-local correlations - Enforces boundary and initial conditions as hard constraints	- Problems with sharp discontinuities (e.g., shocks) - Problems requiring non-local relationship modeling - Systems where boundary/initial conditions must be strictly satisfied	- Memory-intensive and scales quadratically with domain size - Requires high-resolution training data for small residuals - Limited scalability to 3D domains without significant changes
PIKANs [44]	- JaxKAN: Open-source KAN framework in JAX and Flax for faster training - Loss re-weighting and collocation resampling for small PIKAN architectures - Grid-Dependent Basis Functions - Grid adaptations and linear interpolation to solve sharp peaks in the loss function	- Applications needing mesh-free solvers (e.g., fluid dynamics, quantum mechanics) - Complex PDEs in science and engineering	- Large training time - Computational overhead still higher than MLPs - Sensitive to hyperparameter fine-tuning
Physics-informed DeepONets [47]	- Dual-network architecture for continuous coordinates and fixed locations - Incorporates adjustable weights in the loss function - Handles multiscale behavior in PDEs for complex physical systems	- Complex physical systems and processes - Complex parametric PDEs with variable coefficients	- Accuracy drops in regions with steep PDE solution gradients - Training is generally slower compared to conventional DeepONet
SPINN [49]	- Forward-Mode Automatic Differentiation - Utilizes independent sub-networks for each one-dimensional coordinate - Low-Rank Tensor Approximation	- Complex partial differential equations in high-dimensional spaces such as 3D or 4D	- Increased Collocation Points Requirement - Performance relies on coordinate-based MLPs, limiting the utilization of high-dimensional input
Separable PI-DeepONets [50]	- Independent trunk networks for each coordinate axis - Decreases the number of forward passes through trunk and branch networks - Operator learning framework	- High-dimensional problems - Parametric PDEs	- Limited to problems compatible with parameter and coordinate factorization - Struggles with non-grid-based real-world domains - Limited flexibility for diverse input functions
PIMNs using LSTM [**?**]	- Integrates traditional difference schemes with Long Short-Term Memory (LSTM) networks - Regional decomposition to train multiple networks simultaneously - Adaptive functionality to enhance training in high-loss regions	- Nonlinear PDEs - Problems with high-gradient regions	- High memory usage - Requires more training time - Requires meshing, limiting scalability to high-dimensional systems

Table 4: Comparison of architectures, their key features, applications, and limitations.

8. Innovations in PINN via Domain Decomposition:

The idea of combining several PINNs, each trained with a different network architecture, was proposed by Jagtap et al. [51] in 2020. Jagtap et al. presented a conservative PINN by dividing the computational domain into discrete subdomains and then patching the solutions using feasible interface conditions to get a complete solution. This spatial domain decomposition allows efficient network parallelization and flexibility in choosing hyperparameters for networks in each subdomain. To generalize domain decomposition for any PDE, eXtended PINNs [52] (XPINNs) were proposed. XPINN not only supports domain decomposition to any PDE irrespective of its physical nature but also endows both spatial and temporal parallelization which enhances the computational efficiency and reduces the training cost. The architecture of XPINN was improved by Hu, Zheyuan, et al. [53]. He proposed APINN which utilizes a trainable gate network for soft domain decomposition by obviating the interface losses. APINN allows the use of all the training data in each subnetwork and employs partial parameter sharing to capture common features across decomposed functions, averaging the outputs from multiple subnetworks. bc-PINN [54] involves solving the PDE sequentially by training the same network across different time segments, ensuring the solution already obtained is consistent for all preceding time segments. This approach is thus termed as backward compatible PINN (bc-PINN). XPINN suffers from slow convergence because of the simultaneous training of multiple subnetworks, and increasing the number of subdomains results in a large number of network parameters. In contrast, bc-PINN involves training one network at a time but it gathers points from previously trained subdomains each time it trains. Consequently, the training time for each subdomain increases due to a large number of training points. Therefore, Zhang, Dinglei et al. [55] introduce a transferable pre-trained NN depending on time domain decomposition. It uses a simpler network to solve each subdomain independently, by using the pre-trained network (Network trained on the previous subdomain) as a training model for subsequent subdomains. It not only reduces the training time but also improves the prediction accuracy and generalization better than XPINN and bc-PINN. Kharazmi et al. [56] proposed another domain decomposition based on the variational formulation of PINNs (VPINN), named as hp-VPINNs. The variational PINN [57] is based on the Galerkin method in which integration equivalent to the differential equation is used and then integration is computed at the computational domain. In hp-VPINNs, authors used domain decomposition via hp-refinement in VPINN architecture to enhance its performance. Later, the efficiency of hp-VPINNs for handling nonlinear integrands was greatly improved in the work of Liu Chuang and Wu HengAn [58]. They proposed the convolutional variational physics-informed neural network (cv-PINN) in which integration is calculated at quadrature points and employed convolutional filters to specify test functions and quadrature weights. Variational PINN faces slow convergence with stochastic gradient descent optimizers. To improve the convergence of VPINN, Uriarte, Carlos, et al. [59] introduced a hybrid Least Squares/Gradient-Descent optimizer that uses a Least Squares solver for weights of the output layer which significantly enhances the convergence. The authors also demonstrate that forward-mode AD or ultraweak-type

scheme are faster up to 100 times than traditional backward-mode automatic differentiation. Moreover, Finite basis physics-informed neural networks (FBPINNs) [60] were proposed inspired by finite element methods, to efficiently handle large domain problems by using smaller neural networks on overlapping domains, which mitigates spectral bias and reduces the optimization complexity. Later, Dolean, Victorita, et al. [61] extended FBPINNs with multilevel domain decomposition, inspired by the Schwarz domain decomposition method. Multilevel FBPINNs perform better in the case of a large number of subdomains. Recently, Anderson, Samuel, et al. [62] accelerated the training of FBPINNs by replacing subdomain networks with Extreme Learning Machines (ELMs). An improved version of Physical-informed memory networks (PIMNs) based on domain decomposition was introduced by Sun, Jiuyun, et al. [**?**]. This research focuses on splitting the domain into multiple overlapping subdomains and computing the loss in each subdomain independently. A network with adaptive parameters is employed to adjust the weights of loss in each subdomain, enabling efficient training in regions with high loss values. Table 4 summarizes the publications being studied in this section.

Architecture	**Problem Type**	**Architectural Details**	**Performance Evaluation**	**Comparative Analysis**
Conservative PINN [51]	-Nonlinear conservation laws (Burgers, KdV, Euler), Navier-Stokes equations (lid-driven cavity)	-Separate PINNs per sub-domain -Adaptive activation functions -Varying depth/width based on domain complexity	-Relative L2 error $< 10^{-2}$ -Faster convergence with sub-domain parallelization -cPINN achieved up to 10x better accuracy.	-cPINN significantly outperformed PINN in capturing high gradients and discontinuities -PINN struggled to generalize for conservation laws and high-gradient regions, while cPINN provided 10x better accuracy with efficient domain decomposition
eXtended PINN [52]	Viscous Burger's equation	A separate neural network is used for different subdomains, allowing independent hyperparameter optimization for each subdomain.	Relative L2 error of solution is 8.932×10^{-3} And along with interface, the error is 5.9267×10^{-3}.	-XPINN outperforms PINN due to its parallelization and efficient hyperparameter tuning -Provides flexibility in dividing the subdomain for any type of differential equation.

Architecture	**Problem Type**	**Architectural Details**	**Performance Evaluation**	**Comparative Analysis**
Augmented PINN [53]	1D Burger's equation, 2D Helmholtz equation, Klein–Gordon equation, Wave equation, Boussinesq-Burger equation	-Soft and trainable domain decomposition using a gating network -Shared parameters across sub-networks -Eliminates interface loss conditions by using a soft weighted average of sub-network outputs -different variants of APINN are used.	**Burger's**: Mean Relative L2-error = $9.10910^4 3.68910^4$ (best with APINN-X) **Helmholtz**: $1.27510^3 \pm 4.71010^4$ (best with APINN-X) **Klein–Gordon equation**: $2.84610^3 \pm 8.56810^4$ (best with APINN-M) **Wave equation**: $1.29910^3 2.94110^4$ (best with APINN-M) **Boussinesq-Burger equation**:for u is $1.09110^2 \pm 4.58810^3$ and for function v, its $8.18510^2 \pm 2.97310^2$ (best with APINN-M)	-APINN significantly outperforms PINN and XPINN, particularly near interface regions -XPINN fails due to large interface errors, while APINN-X achieves the best accuracy -trainable gate network utilizes all training samples in APINN, leading to enhance its performance than XPINN. - APINN-M (with MPINN-type gate network pretraining) and APINN-X (with XPINN-type gate network pertaining) perform the best, demonstrating superior results compared to other configurations.

Architecture	**Problem Type**	**Architectural Details**	**Performance Evaluation**	**Comparative Analysis**
Backward Compatible PINN(bc-PINN) [54]	1D and 2D Allen-Cahn equations, 1D and 2D Cahn-Hilliard equations	-Sequentially trains over time segments using a single neural network. -Ensures backward compatibility by satisfying solutions from all previous time segments -Uses Initial Condition Guided Learning (ICGL) to accelerate convergence by aligning solutions with initial conditions and Transfer Learning (TL) to reduce training time by reusing learned features across segments. -Phase space representation is adopted to approximate higher-order derivatives.	**1D Cahn-Hillard equation**: Relative total error with bc-PINN is 0.0186 and std-PINN is 0.8594 **1D Allen-Cahn**:Total relative error with bc-PINN =0.0068 **2D Allen-Cahn**: Total prediction error for bc-PINN with ICGL is 2.5% and without ICGL is more than 95%.	-std-PINN does not work for the Allen Cahn equation comprising a strongly non-linear term whereas bc-PINN can accurately predict the solution for the entire domain. -Std-PINN also provides inaccurate solutions because of a higher derivative term in Cahn-Hillard equation. -bc-PINN can achieve high accuracy by using fewer collocation points than std-PINN.

Architecture	**Problem Type**	**Architectural Details**	**Performance Evaluation**	**Comparative Analysis**
hp-VPINN [56]	1D and 2D Poisson equations, Linear Advection-Diffusion equations	-Each subdomain uses a separate neural network -Legendre Polynomials are used as a test function, enhancing accuracy by increasing polynomial order within each subdomains	1D-Poisson with boundary-layer solution: Point-wise error of hp-VPINN is $O(10^7)$(superior to PINN with $O(10^4)$) 1D-Poisson with steep solution: Point-wise error: $O(10^{-5})$ for hp-VPINN with 3 subdomains and $O(10^{-3})$ for PINN	-hp-VPINNs enhance the accuracy and provide faster convergence for sharp-steep solutions compared to PINN. -Combining h-refinement(domain decomposition) and p-refinement (polynomial order) leads to further accuracy.

Architecture	**Problem Type**	**Architectural Details**	**Performance Evaluation**	**Comparative Analysis**
Optimizing VPINN [59]	1D and 2D Poisson's equations with smooth, singularities, and high-frequency components.	-Combines Least square with gradient descent (GD) to enhance the training of Variational PINN. -Forward model differentiation is employed during least square matrix construction to compute spatial derivatives, reducing the computational cost of backward Automatic differentiation. -An alternate formulation (UltraPINN) avoids differentiating trial functions, further cuts the computational cost.	**For Smooth 1D problem** : The relative error is 2.72% (with ADAM) and 0.19% (with LS/ADAM) **High-Frequency 1D problem**: After 1000 iterations, relative error with ADAM is 99.99% and LS/ADAM is 0.66% **Singular 1D**: For discretization M=32 Relative error with ADAM =23.20%, and with LS/ADAM =19.25% For discretization M=64, Relative error with ADAM =22.52%, and with LS/ADAM =16.87% For discretization M=128, Relative error with ADAM =22.33%, and with LS/ADAM =14.98% **For Smooth 2D problem**: Relative error with ADAM =2.15%,and with LS/ADAM =0.44%	-LS/GD significantly reduces the iterations compared to GD alone. -GD optimizes all trainable parameters simultaneously, but LS/GD computes hidden layer's parameters with GD and output layer coefficients with LS. -For singular problems, LS/ADAM reduces error but it requires more discretization in test space which will eventually increase the computational cost.

Architecture	**Problem Type**	**Architectural Details**	**Performance Evaluation**	**Comparative Analysis**
FB-PINNS [60]	-low and high frequency 1D and 2D sinusoidal problems -Viscous time-dependent Burger's equation -Wave equation	-uses finite basis functions to express the solution of a differential equation, inspired by FEM. - computational domain is divided into small, overlapping domains. -networks are trained in parallel, via separate input normalization for each subdomain -The total output is obtained by summing all the output from these sub-networks, weighted by a window function.	-FB-PINN has similar performance as PINN for 1D low frequency problem (L1 error approx. 10^{-3}) -For high-frequency 1D problem, FBPINN outperforms PINN having faster convergency rate and L1 error of 10^{-4} with less training steps. -For 2D problem, L1 of FBPINN was reduced to 10^{-3} within approx. 10,000 training steps while PINN after 40,000 training steps, reduced the L1 error of 10^{-2} -In the case of Burger's equation, FBPINN-coinciding (subdomain interfaces coincide with solution discontinuities) performs slightly worse than PINN, but without coinciding, FBPINN performs better than PINN.	- Require the same number of training points as standard PINNs, leading to a high computational workload. -time domain decomposition reduces the complexity of the optimization problem. -FBPINN is effective across all of the 1D, 2D, and 3D problems -number of overlapping models in the hyper-rectangular subdivision grows exponentially with dimensions, which affects the FBPINN optimization. -a subdomain network size of 2 layers and 16 hidden units is effective for every problem except the wave equation -FBPINN achieves higher accuracy in fewer training steps than PINN in case of high-frequency problems.

Architecture	**Problem Type**	**Architectural Details**	**Performance Evaluation**	**Comparative Analysis**
Multilevel FBPINNs [61]	2D Homogeneous and multiscale Laplace equation	-it advances the FBPINNs by introducing multiple levels of overlapping domains -all multilevel FBPINNs use an exponentially increasing number of subdomains per level -It employs uniform rectangular domain decomposition	-Accuracy of FBPINNs doesn't significantly depend on number of levels in case of 2D homogenous Laplace problem as L1 loss is reducing to approx. 10^{-4}–10^{-5}. -For multi-scale Laplacian problem, Accuracy of FBPINNs increases with increase in levels, achieving normalized L1 loss as low as 10^{-4}	- Multi-level FBPINN outperforms one-level FBPINN and traditional PINNs in terms of computational efficiency and accuracy. -Multilevel FBPINN is more efficient than PINN but its training time is slower than traditional FDM and FEM.
ELM-FBPINN [62]	1D damped harmonic oscillator	-replaces subdomain networks in FBPINN with Extreme Learning Machine (ELM), a network where only weights and biases of the last layer get trained, and other parameters are left initialized (untrained).	L1 loss: PINN = 0.226 FBPINN = 3.1110^{-3} ELM-FBPINN = 5.8710^{-3}	-ELM-FBPINN and FBPINN both models accurately approximated the exact solution but it converges faster and in significantly less time than FB-PINN.

Table 5: Performance Comparison of Different PINN models based on Domain Decomposition

9. Real-World Applications of PINN:

Physics-informed neural networks have gained significant popularity in recent years for solving a wide range of real-world problems. The development of PINN is driven by several salient advantages including improved convergence and interpretability, enhanced training performance with physically consistent results, and reduced search space for weights along with less reliance on large training datasets. This section will highlight some of the notable real-world applications of PINNs across various disciplines during the period from 2023 to 2024. In medical imaging, PINN was employed to reconstruct super-resolution medical images for better analysis and diagnostics [63]. A novel approach based on physics-informed neural networks termed PINQI was proposed by utilizing differentiable optimization layers with learned regularization for quantitative MRI images [64]. A detailed survey of PINN for analysis of medical images is also presented in [65]. Breast cancer remains a growing concern among women, especially in low or middle-income countries. Early detection and diagnosis are crucial for patient recovery and reducing mortality rates in the world. Mukhmetov, Olzhas, et al. [66] used PINN for stimulating temperature distribution in cancerous breast tissues by solving heat transfer PDEs, aiding in the identification of abnormal regions for the detection of tumors. Later, a novel approach based on PINN and temperature data of breast surface is presented to detect breast cancer by predicting tumor heat source through bioheat transfer modeling [67]. A comparative analysis of different network architectures used for modeling 3D blood flow using PINN is shown in [68]. In the field of renewable energy, an improved version of PINN, named Residual Attention-based PINN[69], was used to monitor the spatiotemporal thermal insulation aging of transformers operating in renewable power plants. Wind energy generation is rapidly growing across the world. To measure the wind speed and its direction, a residual-connected PINN [70] is employed by using the Navier-Stokes equation and real-world noisy LIDAR measurements to propose an anti-noise wind field reconstruction algorithm. Moreover, PINN is also used to estimate the remaining lifetime of power electronic devices for their maintenance and safety purposes [71]. In image processing, Namaki, N. et al. [72] proposed blending the PDE-based PINN model to remove noise from images. In the food industry, where drying is the most common preservation method, a PINN-MT model is proposed to predict mass loss and the change in moisture during low-temperature drying [73]. Moreover, the PINN has been used to predict the gasification products of biomass, such as nitrogen, hydrogen, carbon oxide, carbon dioxide, and methane [74], as well as to analyze the transient flow of natural gas inside the pipelines [75]. These applications of PINN highlight their effectiveness in addressing complex real-world problems across various fields, including medical imaging, renewable energy, image processing, and the food industry. The progress of PINN is not limited to these applications, as ongoing research continues to explore new frontiers. In this section, we have highlighted some notable applications from the past two years to provide readers with insights into the potential of PINN architecture.

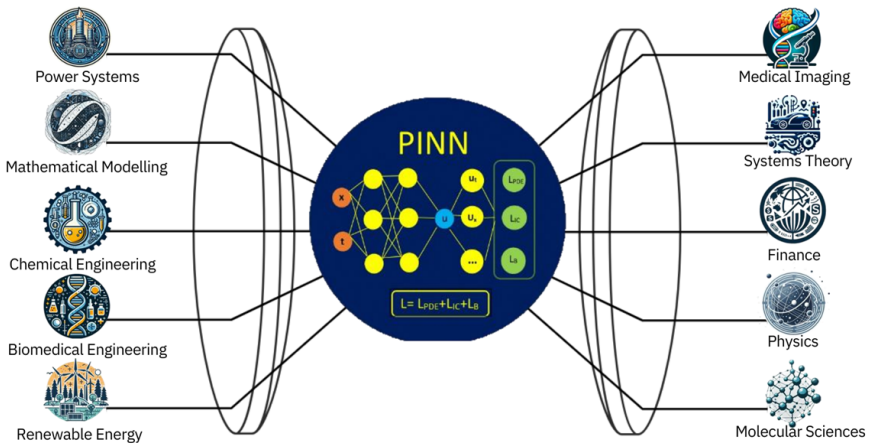

Figure 3: Few Applications of PINNs in Different Fields of Science and Technology. (All the Logos were created using Microsoft Copilot [76]

10. Future Directions:

This section will highlight incomplete results of previous research and propose new directions for future work. Several key areas where advancements can be made to further enhance the training performance will be explored. Activation function plays a crucial role in training because the derivative of the loss function relies on optimization parameters, which are, in turn, influenced by the activation function's derivative. Recent studies have explored improving PINN's accuracy by modifying the activation functions. Physical Activation Functions (PAFs) [77] represent a mathematical approach for embedding physics into the model. The expression of PAFs can be based on terms present in analytical solution, Initial/Boundary conditions. These functions improve the training by minimizing the residual loss and also reducing the size of PINNs by over 75% while maintaining the same accuracy as sigmoid and tanh activation functions. Uddin, Ziya, et al. [6] improved PINN accuracy using Wavelets (Morlet, Mexican Hat, and Gaussian Wavelets) as activation functions, outperforming Tanh. Similarly, PINN trained with periodic (sine) activation function achieved more accurate results and trained up to twice as fast as those using tanh [78]. The effect of activation functions on the convergence of PINN has also been explored [79]. Some research also focused on introducing a scalable parameter in activation functions, such as self-scalable Tanh (Stanh) [80], and Adaptive activation functions [81], which allows gradients to flow easily to compute the derivatives, leading to faster training and better accuracy compared to traditional functions. As introducing a scalable hyperparameter in the activation function increases the convergence and accuracy, further research is needed on the initialization of this trainable parameter along with sensitive analysis. The theoretical perspective of gradient flow in these activation functions should also be explored. To further improve the hyperparameter tuning in PINNs, advanced HPO techniques such as multi-fidelity optimization (MFO) could be investigated. This

approach uses a low-fidelity model to estimate the system's state while high-fidelity further refines the estimation, achieving desirable accuracy with less computational cost. Low-fidelity stills provide the estimation in the absence of high-fidelity data. This approach has proven to be successful in tuning the hyperparameters like merging MFO with recurring learning rate in CNNs, termed MORL [82]. Moreover, combining multi-fidelity strategies with evolutionary algorithms like CMA-ES showed an acceleration in hyperparameters' optimization by up to 15% while maintaining the quality of the solution [83]. In recent years, different domain decomposition strategies integrating into PINN have gained much attention in scientific computing because they improve parallel generalization capability and solution representation [84], [85]. To further enhance the performance, one can employ machine learning algorithms to identify and cluster domain regions that share similar features, treating these regions as a separate subdomain. This will allow the model to focus more on homogeneous subdomains, leading it to capture local behaviors more efficiently. In cases where a previously trained subdomain is used as a pre-trained model for the preceding subdomain [55], collecting training points from hard-to-fit regions can be challenging. Future work could explore employing adaptive sampling strategies within subdomains by concentrating training points into the regions that are challenging to fit. This will allow the model to capture intricate regions of subdomains, thereby improving their accuracy. We anticipate substantial PINN improvements can further improve its performance and applicability in real-world scientific and engineering problems.

11. Conclusions:

This comprehensive review encompasses an in-depth analysis of recent innovations mainly over the past two years which have broadly contributed to improving performance and overcoming training challenges in PINN architecture. The pioneering work of PINN, proposed by Raissi et al. [4] embeds physical laws into neural networks which effectively overcomes the limitations of traditional methods. Over the past few years, new directions have been explored in refining the PINN architecture such as introducing the concept of segmenting large computational domains into manageable subdomains, termed as domain decomposition. Advanced versions of XPINN such as A-PINN and bc-PINN, along with more robust versions of FBPINN demonstrate enhanced computational efficiency and accuracy. Some researchers also explored pathways to optimize network hyperparameters. This review also highlighted new variants of PINN which integrates residual networks, transformers, LSTMs, and Kolmogorov-Arnold-inspired architectures. These innovations underscore scalability by reducing computational costs. Despite these advancements, few studies have also explored the alternatives to replace automatic differentiation (AD) with the Finite Difference Method [86, 87] and Radial Basis Function-Finite Difference method [88] to avoid the use of high-order derivatives; Spectral methods based on orthogonal polynomials [89] and Fourier basis [90] to eliminate the need for computing derivatives, resulting in faster training with low memory allocation. This review also unveils diverse applications of evolving PINN in real-world problems from medical imaging to renewable energy to the food industry, to give readers valuable insights into its potential capabili-

ties. Finally, possible future directions have also been explored, which could contribute to faster training, better interpretability, and more accurate predictions in solving increasingly complex problems and beyond.

Conflict of Interest

The author declares that they have no conflict of interest.

Data availability statement

Not Applicable

Funding statement

No Funding.

Acknowledgements

The author Ilyas Khan extends the appreciation to the Deanship of Postgraduate Studies and Scientific Research at Majmaah University for funding this research work through the project number (PGR-2025-2266).

References

[1] Bivas Bhaumik, Soumen De, and Satyasaran Changdar. Deep learning based solution of nonlinear partial differential equations arising in the process of arterial blood flow. *Mathematics and Computers in Simulation*, 217:21–36, 2024.

[2] Isaac E Lagaris, Aristidis Likas, and Dimitrios I Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, 1998.

[3] Shagun Panghal and Manoj Kumar. Approximate analytic solution of burger huxley equation using feed-forward artificial neural network. *Neural Processing Letters*, 53:2147–2163, 2021.

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

[5] Yifan Wang and Linlin Zhong. Nas-pinn: neural architecture search-guided physics-informed neural network for solving pdes. *Journal of Computational Physics*, 496:112603, 2024.

[6] Ziya Uddin, Sai Ganga, Rishi Asthana, and Wubshet Ibrahim. Wavelets based physics informed neural networks to solve non-linear differential equations. *Scientific Reports*, 13(1):2882, 2023.

[7] Saviz Mowlavi and Saleh Nabi. Optimal control of pdes using physics-informed neural networks. *Journal of Computational Physics*, 473:111731, 2023.

[8] Majid Rasht-Behesht, Christian Huber, Khemraj Shukla, and George Em Karniadakis. Physics-informed neural networks (pinns) for wave propagation and full waveform inversions. *Journal of Geophysical Research: Solid Earth*, 127(5):e2021JB023120, 2022.

[9] Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481):1026–1030, 2020.

[10] Rudolf LM van Herten, Amedeo Chiribiri, Marcel Breeuwer, Mitko Veta, and Cian M Scannell. Physics-informed neural networks for myocardial perfusion mri quantification. *Medical Image Analysis*, 78:102399, 2022.

[11] Carlos Ruiz Herrera, Thomas Grandits, Gernot Plank, Paris Perdikaris, Francisco Sahli Costabal, and Simone Pezzuto. Physics-informed neural networks to learn cardiac fiber orientation from multiple electroanatomical maps. *Engineering with Computers*, 38(5):3957–3973, 2022.

[12] Kuan-Chung Lin, Cheng-Hung Hu, and Kuo-Chou Wang. Innovative deep energy method for piezoelectricity problems. *Applied Mathematical Modelling*, 126:405–419, 2024.

[13] Zhicheng Zhu, Jia Hao, Jin Huang, and Biao Huang. Bc-pinn: an adaptive physics informed neural network based on biased multiobjective coevolutionary algorithm. *Neural Computing and Applications*, 35(28):21093–21113, 2023.

[14] Ana Kaplarević-Mališić, Branka Andrijević, Filip Bojović, Sran Nikolić, Lazar Krstić, Boban Stojanović, and Miloš Ivanović. Identifying optimal architectures of physics-informed neural networks by evolutionary strategy. *Applied Soft Computing*, 146:110646, 2023.

[15] Vikas Dwivedi, Nishant Parashar, and Balaji Srinivasan. Distributed physics informed neural network for data-efficient solution to partial differential equations. *arXiv preprint arXiv:1907.08967*, 2019.

[16] E. Haghighat, M. Raissi, A. Moure, H. Gomez, and R. Juanes. A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics. *Computer Methods in Applied Mechanics and Engineering*, 379:113741, 2021.

[17] E. O. Oluwasakin and A. Q. M. Khaliq. Optimizing physics-informed neural network in dynamic system simulation and learning of parameters. *Algorithms*, 16(12):547, 2023.

[18] Ladislav Zjavka. Construction and adjustment of differential polynomial neural network. 2(3):40–50, 2011.

[19] Maziar Raissi and George Em Karniadakis. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141, 2018.

[20] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10561*, 2017.

[21] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Multistep neural networks for data-driven discovery of nonlinear dynamical systems. *arXiv preprint arXiv:1801.01236*, 2018.

[22] Maziar Raissi. Deep hidden physics models: Deep learning of nonlinear partial differential equations. *Journal of Machine Learning Research*, 19(25):1–24, 2018.

[23] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[24] Seho Son, Jinho Jeong, Dayeon Jeong, Kyung Ho Sun, and Ki-Yong Oh. Physics-informed neural network: Principles and applications. In *IntechOpen*. 2024.

[25] Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021.

[26] Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W Mahoney. Characterizing possible failure modes in physics-informed neural networks. In *Advances in Neural Information Processing Systems*, pages 26548–26560, 2021.

[27] Sifan Wang, Shyam Sankaran, and Paris Perdikaris. Respecting causality is all you need for training physics-informed neural networks. *arXiv preprint arXiv:2203.07404*, 2022.

[28] Ehsan Haghighat and Ruben Juanes. Sciann: A keras/tensorflow wrapper for scientific computations and physics-informed deep learning using artificial neural networks. *Computer Methods in Applied Mechanics and Engineering*, 373:113552, 2021.

[29] Stefano Markidis. The old and the new: Can physics-informed deep-learning replace traditional linear solvers? *Frontiers in Big Data*, 4:669097, 2021.

[30] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems*, volume 24, 2011.

[31] Paul Escapil-Inchauspé and Gonzalo A Ruz. Hyper-parameter tuning of physics-informed neural networks: Application to helmholtz problems. *Neurocomputing*, 561:126826, 2023.

[32] Xiaoli Chen, Jinqiao Duan, and George Em Karniadakis. Learning and meta-learning of stochastic advection–diffusion–reaction systems from sparse measurements. *European Journal of Applied Mathematics*, 32(3):397–420, 2021.

[33] Prakhar Sharma, Llion Evans, Michelle Tindall, and Perumal Nithiarasu. Hyperparameter selection for physics-informed neural networks (pinns)–application to discontinuous heat conduction problems. *Numerical Heat Transfer, Part B: Fundamentals*, pages 1–15, 2023.

[34] Elliott Skomski, Jan Drgona, and Aaron Tuor. Physics-informed neural state space models via learning and evolution. *arXiv preprint arXiv:2011.13497*, 2020.

[35] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8697–8710, 2018.

[36] Yicheng Wang, Xiaotian Han, Chia-Yuan Chang, Daochen Zha, Ulisses Braga-Neto, and Xia Hu. Auto-pinn: understanding and optimizing physics-informed neural ar-

chitecture. *arXiv preprint arXiv:2205.13748*, 2022.

[37] Petra Vidnerová and Roman Neruda. Evolution strategies for deep neural network models design. In *ITAT*, 2017.

[38] Liangliang Yan, You Zhou, Huan Liu, and Lingqi Liu. An improved method for physics-informed neural networks that accelerates convergence. *IEEE Access*, 2024.

[39] Zhiyuan Zhao, Xueying Ding, and B Aditya Prakash. Pinnsformer: A transformer-based framework for physics-informed neural networks. *arXiv preprint arXiv:2307.11833*, 2023.

[40] Feng Zhang, Long Nghiem, and Zhangxin Chen. A novel approach to solve hyperbolic buckley-leverett equation by using a transformer based physics informed neural network. *Geoenergy Science and Engineering*, 236:212711, 2024.

[41] Ruben Rodriguez-Torrado, Pablo Ruiz, Luis Cueto-Felgueroso, Michael Cerny Green, Tyler Friesen, Sebastien Matringe, and Julian Togelius. Physics-informed attention-based neural network for hyperbolic partial differential equations: application to the buckley–leverett problem. *Scientific Reports*, 12(1):7557, 2022.

[42] Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljačić, Thomas Y Hou, and Max Tegmark. Kan: Kolmogorov-arnold networks. *arXiv preprint arXiv:2404.19756*, 2024.

[43] Yizheng Wang, Jia Sun, Jinshuai Bai, Cosmin Anitescu, Mohammad Sadegh Eshaghi, Xiaoying Zhuang, Timon Rabczuk, and Yinghua Liu. Kolmogorov arnold informed neural network: A physics-informed deep learning framework for solving pdes based on kolmogorov arnold networks. *arXiv preprint arXiv:2406.11045*, 2024.

[44] Spyros Rigas, Michalis Papachristou, Theofilos Papadopoulos, Fotios Anagnostopoulos, and Georgios Alexandridis. Adaptive training of grid-dependent physics-informed kolmogorov-arnold networks. *IEEE Access*, 2024.

[45] Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.

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

[47] Sifan Wang, Hanwen Wang, and Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed deeponets. *Science Advances*, 7(40):eabi8605, 2021.

[48] Kuangdai Leng, Mallikarjun Shankar, and Jeyan Thiyagalingam. Zero coordinate shift: Whetted automatic differentiation for physics-informed operator learning. *Journal of Computational Physics*, 505:112904, 2024.

[49] Junwoo Cho, Seungtae Nam, Hyunmo Yang, Seok-Bae Yun, Youngjoon Hong, and Eunbyung Park. Separable physics-informed neural networks. In *Advances in Neural Information Processing Systems*, volume 36, 2024.

[50] Luis Mandl, Somdatta Goswami, Lena Lambers, and Tim Ricken. Separable deeponet: Breaking the curse of dimensionality in physics-informed machine learning. *arXiv preprint arXiv:2407.15887*, 2024.

[51] Ameya D Jagtap, Ehsan Kharazmi, and George Em Karniadakis. Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. *Computer Methods in Applied Mechanics and Engineering*, 365:113028, 2020.

[52] Ameya D Jagtap and George Em Karniadakis. Extended physics-informed neural networks (xpinns): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, 28(5), 2020.

[53] Zheyuan Hu, Ameya D Jagtap, George Em Karniadakis, and Kenji Kawaguchi. Augmented physics-informed neural networks (apinns): A gating network-based soft domain decomposition methodology. *Engineering Applications of Artificial Intelligence*, 126:107183, 2023.

[54] Revanth Mattey and Susanta Ghosh. A novel sequential method to train physics informed neural networks for allen cahn and cahn hilliard equations. *Computer Methods in Applied Mechanics and Engineering*, 390:114474, 2022.

[55] Dinglei Zhang, Ying Li, and Shihui Ying. Trans-net: A transferable pretrained neural networks based on temporal domain decomposition for solving partial differential equations. *Computer Physics Communications*, 299:109130, 2024.

[56] Ehsan Kharazmi, Zhongqiang Zhang, and George Em Karniadakis. hp-vpinns: Variational physics-informed neural networks with domain decomposition. *Computer Methods in Applied Mechanics and Engineering*, 374:113547, 2021.

[57] Ehsan Kharazmi, Zhongqiang Zhang, and George Em Karniadakis. Variational physics-informed neural networks for solving partial differential equations. *arXiv preprint arXiv:1912.00873*, 2019.

[58] Chuang Liu and HengAn Wu. cv-pinn: Efficient learning of variational physics-informed neural network with domain decomposition. *Extreme Mechanics Letters*, 63:102051, 2023.

[59] Carlos Uriarte, Manuela Bastidas, David Pardo, Jamie M Taylor, and Sergio Rojas. Optimizing variational physics-informed neural networks using least squares. *arXiv preprint arXiv:2407.20417*, 2024.

[60] Ben Moseley, Andrew Markham, and Tarje Nissen-Meyer. Finite basis physics-informed neural networks (fbpinns): a scalable domain decomposition approach for solving differential equations. *Advances in Computational Mathematics*, 49(4):62, 2023.

[61] Victorita Dolean, Alexander Heinlein, Siddhartha Mishra, and Ben Moseley. Multilevel domain decomposition-based architectures for physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 429:117116, 2024.

[62] Samuel Anderson, Victorita Dolean, Ben Moseley, and Jennifer Pestana. Elm-fbpinn: efficient finite-basis physics-informed neural networks. *arXiv preprint arXiv:2409.01949*, 2024.

[63] Zachary Burns and Zhaowei Liu. Untrained, physics-informed neural networks for structured illumination microscopy. *Optics Express*, 31(5):8714–8724, 2023.

[64] Felix F Zimmermann, Christoph Kolbitsch, Patrick Schuenke, and Andreas Kofler.

Pinqi: an end-to-end physics-informed approach to learned quantitative mri reconstruction. *IEEE Transactions on Computational Imaging*, 2024.

[65] Chayan Banerjee, Kien Nguyen, Olivier Salvado, Truyen Tran, and Clinton Fookes. Pinns for medical image analysis: A survey. *arXiv preprint arXiv:2408.01026*, 2024.

[66] Olzhas Mukhmetov, Yong Zhao, Aigerim Mashekova, Vasilios Zarikas, Eddie Yin Kwee Ng, and Nurduman Aidossov. Physics-informed neural network for fast prediction of temperature distributions in cancerous breasts as a potential efficient portable ai-based diagnostic tool. *Computer Methods and Programs in Biomedicine*, 242:107834, 2023.

[67] Isaac Perez-Raya, Carlos Gutierrez, and Satish Kandlikar. A transformative approach for breast cancer detection using physics-informed neural network and surface temperature data. *ASME Journal of Heat and Mass Transfer*, 146(10), 2024.

[68] Philipp Moser, Wolfgang Fenz, Stefan Thumfart, Isabell Ganitzer, and Michael Giretzlehner. Modeling of 3d blood flows with physics-informed neural networks: comparison of network architectures. *Fluids*, 8(2):46, 2023.

[69] Ibai Ramirez, Joel Pino, David Pardo, Mikel Sanz, Luis del Rio, Alvaro Ortiz, Kateryna Morozovska, and Jose I Aizpurua. Residual-based attention physics-informed neural networks for efficient spatio-temporal lifetime assessment of transformers operated in renewable power plants. *arXiv preprint arXiv:2405.06443*, 2024.

[70] Runze Tian, Peng Kou, Yuanhang Zhang, Mingyang Mei, Zhihao Zhang, and Deliang Liang. Residual-connected physics-informed neural network for anti-noise wind field reconstruction. *Applied Energy*, 357:122439, 2024.

[71] Zhonghai Lu, Chao Guo, Mingrui Liu, and Rui Shi. Remaining useful lifetime estimation for discrete power electronic devices using physics-informed neural network. *Scientific Reports*, 13(1):10167, 2023.

[72] Neda Namaki, MR Eslahchi, and Rezvan Salehi. The use of physics-informed neural network approach to image restoration via nonlinear pde tools. *Computers & Mathematics with Applications*, 152:355–363, 2023.

[73] Chanaka P Batuwatta-Gamage, Charith Rathnayaka, Helambage CP Karunasena, Hyogu Jeong, Azharul Karim, and Yuan Tong Gu. A novel physics-informed neural networks approach (pinn-mt) to solve mass transfer in plant cells during drying. *Biosystems Engineering*, 230:219–241, 2023.

[74] Shaojun Ren, Shiliang Wu, and Qihang Weng. Physics-informed machine learning methods for biomass gasification modeling by considering monotonic relationships. *Bioresource Technology*, 369:128472, 2023.

[75] Chi Zhang and Abdollah Shafieezadeh. Nested physics-informed neural network for analysis of transient flows in natural gas pipelines. *Engineering Applications of Artificial Intelligence*, 122:106073, 2023.

[76] Microsoft. Microsoft copilot. Available from: https://copilot.microsoft.com/, 2023.

[77] Jassem Abbasi and Pål Østebø Andersen. Physical activation functions (pafs): An approach for more efficient induction of physics into physics-informed neural networks (pinns). *Biosystems Engineering*, 608:128352, 2024.

[78] Salah A Faroughi, Ramin Soltanmohammadi, Pingki Datta, Seyed Kourosh Mahjour,

and Shirko Faroughi. Physics-informed neural networks with periodic activation functions for solute transport in heterogeneous porous media. *Mathematics*, 12(1):63, 2023.

[79] Paweł Maczuga and Maciej Paszyński. Influence of activation functions on the convergence of physics-informed neural networks for 1d wave equation. In *International Conference on Computational Science*, pages 74–88. Springer, 2023.

[80] Raghav Gnanasambandam, Bo Shen, Jihoon Chung, Xubo Yue, et al. Self-scalable tanh (stan): Faster convergence and better generalization in physics-informed neural networks. *arXiv preprint arXiv:2204.12589*, 2022.

[81] Ameya D Jagtap, Kenji Kawaguchi, and George Em Karniadakis. Adaptive activation functions accelerate convergence in deep and physics-informed neural networks. *Journal of Computational Physics*, 404:109136, 2020.

[82] HyunJae Lee, Gihyeon Lee, Junhwan Kim, Sungjun Cho, Dohyun Kim, and Donggeun Yoo. Improving multi-fidelity optimization with a recurring learning rate for hyperparameter tuning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2309–2318, 2023.

[83] Shintaro Takenaga, Yoshihiko Ozaki, and Masaki Onishi. Dynamic fidelity selection for hyperparameter optimization. In *Proceedings of the Companion Conference on Genetic and Evolutionary Computation*, pages 2304–2307, 2023.

[84] T. Hai, A. Basem, A. Alizadeh, et al. Optimizing ternary hybrid nanofluids using neural networks, gene expression programming, and multi-objective particle swarm optimization: a computational intelligence strategy. *Scientific Reports*, 15:1986, 2025.

[85] K. Sudarmozhi, V. Sreelatha Devi, H. F. Öztop, H. Ahmad, A. Abd-Elmonem, W. Jamshed, and N. A. A. Siddig. Ai-neural network modelling of williamson blood flow in porous medium soret-dufour effects with tetra hybrid nanoparticles. *International Communications in Heat and Mass Transfer*, 171:110107, 2026.

[86] Pu Ren, Chengping Rao, Yang Liu, Jian-Xun Wang, and Hao Sun. Phycrnet: Physics-informed convolutional-recurrent network for solving spatiotemporal pdes. *Computer Methods in Applied Mechanics and Engineering*, 389:114399, 2022.

[87] Chengping Rao, Pu Ren, Qi Wang, Oral Buyukozturk, Hao Sun, and Yang Liu. Encoding physics to learn reaction–diffusion processes. *Nature Machine Intelligence*, 5(7):765–779, 2023.

[88] Ramansh Sharma and Varun Shankar. Accelerated training of physics-informed neural networks (pinns) using meshless discretizations. In *Advances in Neural Information Processing Systems*, volume 35, pages 1034–1046, 2022.

[89] Mingtao Xia, Lucas Böttcher, and Tom Chou. Spectrally adapted physics-informed neural networks for solving unbounded domain problems. *Machine Learning: Science and Technology*, 4(2):025024, 2023.

[90] Tianchi Yu, Yiming Qi, Ivan Oseledets, and Shiyi Chen. Spectral informed neural network: An efficient and low-memory pinn. *arXiv preprint arXiv:2408.16414*, 2024.