International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(23s) 2026 pp. 828-844
https://doi.org/10.70917/ijcisim-2026-5648

RDNN+: A Robust Neural Network Framework for Solving Nonlinear Partial Differential Equations

Muppidi Maruthi[1], Y. Rajashekhar Reddy[2]

[1]JNTUH University College of Engineering Jagtial, Telangana, India. Email: muppidimaruthi1414@gmail.com
[2]JNTUH University College of Engineering Science and Technology, Hyderabad, Telangana, India. Email: yrsreddy4@gmail.com
Corresponding Author: Muppidi Maruthi (muppidimaruthi1414@gmail.com)

Abstract: Physics-informed neural networks (PINNs) have become a pertinent direction of the mesh-free approach to the solution of nonlinear partial differential equations (PDEs); nevertheless, they deteriorate greatly when it comes to sharp gradients, discontinuities, and delayed dynamics. Specifically, the smoothness bias of neural networks' inherent nature and the consistent application of physical constraints restrict the capability of conventional PINNs to be accurate and robust to shock-dominated systems or memory-dependent systems. In an effort to overcome these problems, the current paper presents RDNN+, a physics-based learning framework combining hybrid physics to solve nonlinear PDEs with both discontinuities and time-delays effects. The given method combines together discontinuity-conscious learning strategy that decomposes the space of solutions adaptively into the smooth and non-smooth subsets and a delay-differential formulation directly incorporated into the physics-based loss. This is a directional application of physical restraints and is better at convergence stability and an increase in shock resolution. RDNN+ is proven to be effective on the viscous Burgers equation in both regular and delay-dependent conditions. According to quantitative assessments, global relative error reduction and better localization of shock fronts are achieved with the hybrid PINNs than with conventional PINNs and hybrid baselines, especially at higher delay parameters. These findings demonstrate the prospect of RDNN+ as a powerful and generalizable model of application in scientific machine learning to discontinuous and memory-based physical systems.
Keywords: Discontinuity-aware learning, scientific machine learning, hybrid physics-informed models, delay differential equations, shock-capturing neural networks

1. INTRODUCTION

1.1 Background and motivation of the study

The Partial Differential Equations (PDEs) have been in the limelight of modeling the complex phenomena in engineering, physics, and applied sciences, such as fluid flow, transport processes, and biological systems. Finite difference, finite element, and finite volume methods have been shown to be accurate in practice on well-posed problems and therefore have been used to solve nonlinear, high-dimensional problems with sharp gradients, discontinuities, or complex boundary interactions, but are computationally expensive and subject to stability issues (Cuomo et al., 2022). Physics-Informed Neural Networks (PINNs) are one of the recently popular mesh-free options that represent the driving physics within neural network training (Ren et al., 2025). Nonetheless, their promise has been shown to have some underlying drawbacks of standard PINNs in recent studies. Neural networks are naturally smooth bias (as well as the global enforcement of physics-based residuals) leads to poor convergence and lower accuracy at discontinuities, shock fronts, and advection-dominated regions (Yu et al., 2022). Besides, the majority of current PINN expressions presuppose instant system dynamics and do not consider delayed dependencies, which are commonplace in real-world physical systems (Lv et al., 2021). These deficiencies have inspired an increasing amount of interest in hybrid and adaptive physics-informed models that combine numerical strength with information learning. Such models are required to enhance the effectiveness and usability of the scientific machine learning techniques on complex, nonlinear PDEs.

1.2 Problem statement

A wide range of science and engineering problems can be solved by transforming them into differential equations. Differential equations are widely applied in fields such as fluid mechanics, material science, pathology, physiology and neuroscience. Though several methods are available to solve the differential equations, they are often mathematically and computationally expensive. Hence there exists a strong need for development of approximation approaches. Rapid improvements in the field of Deep Learning (DL) techniques has led to emergence of DL based solutions to numerous real world problems pertaining to fields such as medicine, healthcare, etc. These improvements have enabled the construction of efficient deep learning based solutions for mathematical problems as well. Application of deep intelligence models for the purpose of finding solutions for linear and nonlinear partial differential equations is found to be very effective in recent times.

The work in the base paper has proposed a hybrid neural network architecture for the purpose of solving nonlinear partial differential equations. The work has employed a discontinuity indicator to classify the local solutions into smooth and non-smooth scales. The scheme has deployed an automatic differentiation approach for resolution of smooth scales and the WENO mechanism is used for recording discontinuities. The WENO-based strategy discussed in the base work serves as the conceptual motivation for discontinuity handling. In the proposed RDNN+ framework, WENO is not explicitly implemented. Instead, discontinuities are identified through a gradient-based discontinuity indicator integrated into the neural network training process, enabling adaptive learning in shock-dominated regions while maintaining computational efficiency. The effectiveness of the proposed model in approximating the discontinuous solution is evaluated in terms of global relative errors. Careful investigation of the results reveal that there is a considerable scope for minimizing the error metric. Yet another limitation is that the work has restricted its investigation to one dimensional problem. Hence the proposed work shall focus on the following aspects.

1.3 Purpose and Objectives of the Study

Purpose

The purpose of this work is to design RDNN+, which is a reliable and mixed-method neural net that may solve nonlinear partial differential equations (PDEs) with increased accuracy, perform and manage errors.

Objectives

The work consists of three important objectives as follows:

- To invent a new hybrid neural network construction with the discontinuity indicators, to enhance the accuracy of solving nonlinear PDEs.
- To include explanatory deep-learning algorithms to the extent that the inner decisions of the model can be made more and more understandable.
- To carry the methodology to multi-dimensional PDEs and observe the accuracy and error for different bars and test the performance concerning multiple accuracy and error bars.

1.4 State-of-the-Art Overview

Physics-Informed Neural Networks (PINS) is a new method that uses partial differential equations (PDEs) as the entity to be solved directly and without a grid. Rather than relying on the PDE to be a grid-based problem, PINNs incorporate the equation of the PDE into the loss equation of a neural network. It allows neural networks to process highly dimensional problems and even inverse problems (Luo et al., 2025). Nevertheless, on the other as per Abbasi et al. (2025), plain PINNs may not be easy to train on sharp-gradient or discontinuous functions, as neural networks are smooth by nature, and the optimization turns out to be challenging in the high-nonlinearity areas. In order to solve these problems, hybrid and domain-decomposed algorithms are used. The problems in XPINNs and cPINNs partition the problem domain into small regions and allow the regions to learn in parallel. This parallel learning enhances actions on complex problems or multi-dimensional problems. In other words, other forms of discontinuity, e.g., WENO-based weighting functions or shock detectors of high orders, are added to the training functional to train on smooth regions and target the nonsmooth region (Nogueira et al., 2024). More recent advancements are the operator-based learning, which can be neural operators and DeepONet. These two frameworks learn an operator, such as the differential operator in a PDE, using networks, then use operators that can act on other geometries or parameters (Zhong and Meidani, 2024). The latest art is a combination of physics-informed learning, numerical shock capturing, domain decomposition, adaptive activation, and operator learning.

1.5 Research Significance and contributions

The RDNN+ framework will create a huge impact in scientific computation and engineering modeling. PINNs already make mesh-free physics-informed predictions of complex PDEs without such a reliance on data and at a much reduced cost as compared with conventional solvers. These are also applied in neural operators such as the Fourier Neural Operator (FNO) and DeepONet that train to learn the entire solution space of PDEs, which enables orders-of-magnitude acceleration of parametric simulations (Li et al., 2020). Nonetheless, in spite of these developments, it is difficult to quantify discontinuities accurately and to generalize to problems of higher dimension and multi-scale. RDNN+ is the bridge that fills this gap by incorporating hybrid networks that can be explained with discontinuity indicators and the scalability of the architecture. Its larger implication consists of enhanced scientific knowledge, enhanced effectiveness of industrial simulations of complicated systems, and greater comprehensibility of AI-embarked physical modeling.

The study presents the RDNN+, a powerful and mixed neural net that runs close to the solution of nonlinear partial integral equations (Yuan et al., 2022). Its architecture is discontinuity aware and optimizes global relative error and keeps the model more accurate in both smooth and non-smooth parts. It is also able to introduce explainable learning mechanisms to simplify the system and enable easy interpretation. RDNN+ is able to execute many-dimensional, and applies a number of performance evaluation measures in order to provide a comprehensive analysis of performance. All of this enhances the neural PDE solvers to become more dependable, flexible, and analytically twinged.

2. Related Works

2.1 Classical Numerical Solvers

The primary means by which scientists and engineers solve partial differential equations (PDEs) has traditionally been through classical numerical methods. Gatiso et al. (2021) stated that the finite difference method (FDM) converts derivatives into simple algebraic grid stencils. While FDM is efficient for linear problems, it is not well-suited for handling complex geometries or nonlinear dynamical systems. The Finite Volume Method (FVM) enforces conservation laws by evaluating fluxes at the boundaries of control volumes. This makes it particularly suitable for computational fluid dynamics and for modeling discontinuities (Saeed and Alfawaz, 2025). However, the use of unstructured meshes for irregular geometries increases both implementation complexity and computational cost.

The Finite Element Method (FEM) is based on piecewise polynomial basis functions defined over mesh elements. FEM performs well on complex geometries and boundary conditions but can become computationally expensive and mesh-sensitive for nonlinear or high-dimensional problems. Global basis function approaches provide exponential accuracy for smooth solutions but perform poorly when solutions are discontinuous or exhibit rapid variations (Droniou, 2014). Although classical methods provide accurate solutions for well-behaved problems, they scale poorly, become computationally demanding in non-smooth scenarios, and struggle with high-dimensional nonlinear PDEs.

2.2 Standard Physics-Informed Neural Networks (PINNs)

Recent advances in physics-informed deep learning have significantly transformed the way PDEs are solved. Lu et al. (2021) introduced the PINN framework, which embeds governing equations into neural network loss functions using automatic differentiation. This enables mesh-free solutions for forward and inverse problems, including integro-differential, stochastic, and fractional equations. Before PINNs, shallow neural PDE solvers were explored but suffered from limited expressive capacity and poor scalability to high-dimensional problems (Hafiz et al., 2024). A breakthrough came with deep backward stochastic differential equation (BSDE) methods, which reinterpreted high-dimensional PDEs such as Black–Scholes and Allen–Cahn equations using neural networks to alleviate the curse of dimensionality. In parallel, operator-learning frameworks such as DeepONet and neural operators demonstrated that neural networks can learn mappings between function spaces with discretization invariance. These developments collectively established a new class of data-efficient and physics-compatible neural solvers.

2.3 Discontinuity-Aware PINNs

Physics-Informed Neural Networks incorporate PDE residuals, along with initial and boundary conditions, directly into the training loss (Cuomo et al., 2022). While PINNs have shown success across a wide range of PDEs, including nonlinear diffusion, Navier–Stokes, and Schrödinger equations, they exhibit significant limitations. Convergence can be slow, loss weighting requires careful tuning, and performance degrades in the presence of discontinuities or stiffness due to non-convex loss landscapes and vanishing gradients (Ren et al., 2025). Benchmark studies have reported persistent difficulties in capturing shock fronts, particularly in multiphase flows and high-dimensional settings. To address these issues, domain-decomposed approaches such as cPINNs and XPINNs introduce parallel subnetworks to improve scalability and handle discontinuities more effectively. Hybrid activation strategies and adaptive residual weighting have also been proposed to stabilize optimization and reduce residual errors (Randive et al., 2025).

2.4 Delay-Aware and Hybrid Models

Despite these improvements, PINN-based approaches continue to struggle with sharp discontinuities. To enhance performance, researchers have developed hybrid frameworks that combine PINNs with numerical shock-capturing techniques. The hybrid PINN proposed by Lv et al. (2021) integrates a discontinuity indicator with automatic differentiation in smooth regions and a high-order WENO scheme near shocks. This approach demonstrated superior performance on both inviscid and viscous Burgers' equations compared to standard PINNs (Ortiz et al., 2024). Additional strategies include gradient-weighting schemes that locally adjust PDE residuals near discontinuities to improve shock resolution without sacrificing stability (You et al., 2023). The Hodd-PINN framework combines high-order finite difference discretizations with WENO-based discontinuity detection and has shown error reductions of up to 24% in high-speed compressible flow simulations. Other discontinuity-capturing PINNs address issues such as viscosity jumps and cusps by employing multiple subnetworks enriched with level-set features to control region-specific solution behavior (Tseng and Lai, 2023). These advances highlight ongoing efforts to robustify neural PDE solvers in discontinuous and multiscale contexts.

2.5 Research Gap Summary

Although neural PDE solvers and PINNs have demonstrated promising results, significant limitations remain. Capturing sharp discontinuities, scaling to multi-dimensional problems, and ensuring model interpretability continue to pose challenges. Many existing models are difficult to generalize across different nonlinear PDEs and require high sampling densities to achieve convergence. Despite numerous hybrid approaches, no single framework has yet achieved simultaneous robustness to discontinuities, accurate solution approximation, and clear interpretability. This unresolved gap motivates the development of RDNN+, which combines deep learning with hybrid numerical strategies and explainable modeling to deliver reliable and physically consistent solutions for complex nonlinear PDEs.

3. Research Methodology

This booklet details the procedure by which RDNN+, a powerful deep-learning model to solve nonlinear Partial Differential Equations (PDEs), has been developed. Its five major components are the following: concluding the PDE, constructing initial and boundary conditions. The proposed RDNN+ framework is developed by first constructing a standard PINN, extending it into a hybrid PINN sub-module with a discontinuity indicator, and subsequently integrating a delay-differential component to enable time-delay modeling. In this work, RDNN+ denotes the complete proposed framework that integrates discontinuity-aware learning with delay-differential modeling. The terms hPINN or HPINN are used exclusively to refer to intermediate hybrid physics-informed sub-modules that do not include the full RDNN+ architecture.

3.1. PDE Finalization: Burger's Equation

The Burger equation is a nonlinear term due to convection combined with a linear term, the diffusive one, with an expression as written in the form of is viscous term. It is a typical fluid mechanics prototype nonlinear PDE.

$$\partial u/\partial t + u\, \partial u/\partial x = v\, \partial^2 u/\partial x^2 \qquad (1)$$

Here:

- $u(x,t)$: velocity field

- t: time

- x: spatial coordinate

- ν: kinematic viscosity (diffusion coefficient)

The Burgers equation exhibits shock formation and dissipation phenomena, making analytical solutions difficult to obtain in many practical situations. It is an appropriate problem to measure the robustness of deep-learning solvers, particularly those built to handle sudden changes in a solution (sharp gradients).

3.2. Initial and Boundary Conditions

To work on the Burger equation in a meaningful way in a neural network, the team has developed a form of the initial-boundary value problem (IBVP) on a definite both spatial and temporal region. The spatial domain $x \in [0, 1]$, and the temporal domain $t \in [0, T]$, where T represents the last simulation time.

The one-dimensional viscous Burgers' equation considered here is:

$$\partial u/\partial t + u\, \partial u/\partial x = v\, \partial^2 u/\partial x^2 \quad (2)$$

Where

V = ν = 0.01/pi

Initial Condition:

$$u(x, 0) = -\sin(\pi x) \quad (3)$$

The first condition is a smooth-shocking profile that enables the framework to examine smooth and shock behaviors as the evolution progresses.

Boundary Conditions (Dirichlet):

$u(0, t) = 0, \quad u(1, t) = 0$

Additional Conditions (if required for DDE settings):

$$u\,(x, t-\tau) = f(x) \text{for } t < \tau \quad (4)$$

3.3. Physics Informed Neural Network (PINN) Model

The researchers constructed an ordinary PINN model first. A PINN is a neural network that learns the solution to a PDE based on the residual of the PDE. This model is built as follows, generally with PINN:

Key Components

Input: Two features – spatial coordinate x and time t

Output: Network prediction of solution u(x,t)

Architecture: Fully connected feedforward network with multiple hidden layers and smooth activation functions (e.g., tanh or ReLU)

Loss Function:

L_PDE: Residual of Burger's equation

Burgers Equation: $\partial u/\partial t + u\, \partial u/\partial x = v\, \partial^2 u/\partial x^2$

Residual: $R(x, t) = \partial u/\partial t + u\, \partial u/\partial x - v\, \partial^2 u/\partial x^2$

L_PDE = MSE(R(x, t))

L_IC: Initial condition loss

Given: u(x, 0) = -sin(pi * x)

$L\_IC = MSE(u(x, 0) + \sin(\pi * x))$

L_BC: Boundary condition loss

$u(0, t) = 0$ and $u(1, t) = 0$

$L\_BC = MSE(u(0, t)) + MSE(u(1, t))$

Total Loss

$$L\_Total = \lambda 1 * L\_PDE + \lambda 2 * L\_IC + \lambda 3 * L\_BC \quad (5)$$

Additional Equation - Derivatives via Automatic Differentiation

$\partial u/\partial t = d/dt\ (u(x, t))$

$\partial u/\partial x = d/dx\ (u(x, t))$

$\partial^2 u/\partial x^2 = d^2/dx^2\ (u(x, t))$

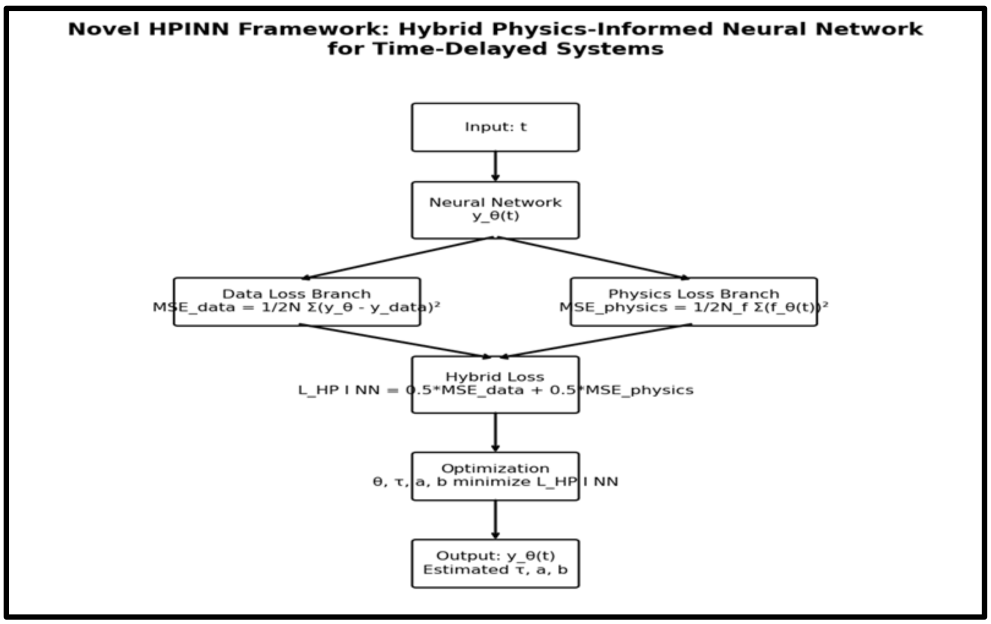

Fig 1 Workflow of the delay-aware hybrid physics-informed neural network (HPINN)

The picture of workflow starts with formulating the target nonlinear partial differential equation (PDE), where Burger is selected because of its physical relevance. Afterwards, existing numerical solutions are analyzed in order to point out limitations. A typical PINN framework is then built. This intermediate model is extended into a hybrid PINN sub-module by incorporating a discontinuity indicator, which enables enhanced learning of sharp gradients within the overall RDNN+ framework. The yielded architecture is implemented on the targeted PDE. Automatic differentiation calculates partial derivatives of the network output concerning its inputs. It implies that this network can reduce the physically underrepresented distance as well as physical inconsistencies. Vanilla PINNs can perform effectively in smooth functions, yet have trouble with discontinuities. Activation functions are smooth in design so that they cannot capture pointy changes such as shocks. A discontinuity indicator was added to a hybrid PINN (hPINN) to achieve better results.

3.4. Hybrid PINN (hPINN) with Discontinuity Indicator

Enhancements Introduced by the Discontinuity-Aware Hybrid PINN Sub-Module:

$Indicator(x) = 1,\ \text{if } |\partial u/\partial x| > \varepsilon$

(6)

$= 0,\ \text{otherwise}$

Where:

$\partial u/\partial x$ is the spatial gradient of the solution,

ε is a predefined threshold.

The threshold ε was empirically set to 0.5 based on preliminary experiments. This value provided a good balance between sensitivity to shock regions and training stability. Lower values produced excessive false discontinuity detections, whereas larger values missed important shock structures.

Adaptive Loss Weighting

L_Total = λ1 * L_PDE + λ2 * L_IC + λ3 * L_BC + λ4 * Indicator(x) * L_Discont (7)

Where,

L_Discont is the discontinuity-based loss,

λ4 increases adaptively in high-gradient regions,

This allows better convergence in complex subdomains.

Residual Decomposition

L_PDE_Total = L_PDE_smooth + L_PDE_shock

The final output:

u(x, t) = (1 - Indicator(x)) * Net_smooth(x, t) + Indicator(x) * Net_shock(x, t) (8)

Discontinuity Detection Module: During the training loop, gradient sensors mark during non-smooth behavior of the solution in every area.

Adaptive Loss Weighting: weightings on the contributions of the losses are altered to focus more on high-gradient areas quickly and learn in complex areas faster.

Residual Decomposition: Rather than having a global loss, residuals are calculated on individual subdomains (smooth versus non-smooth parts) to further converge, and to learn local details.

Hybrid Architecture: Multiple subnetworks may be implemented here: One subnetwork that smooths parts and another one that discontinuously parts. The transition functions combine the outputs depending on the indicator of discontinuity.

Advantages

Within the proposed RDNN+ framework, the integration of the discontinuity-aware hybrid PINN sub-module leads to improved shock resolution, reduced global relative error, and enhanced training stability through adaptive learning. This means this hybrid technique links both data-driven learning and conventional numerical robustness, the prerequisite to the next improvement (temporal memory modeling).

3.5. Delay Differential Equation (DDE) Solver

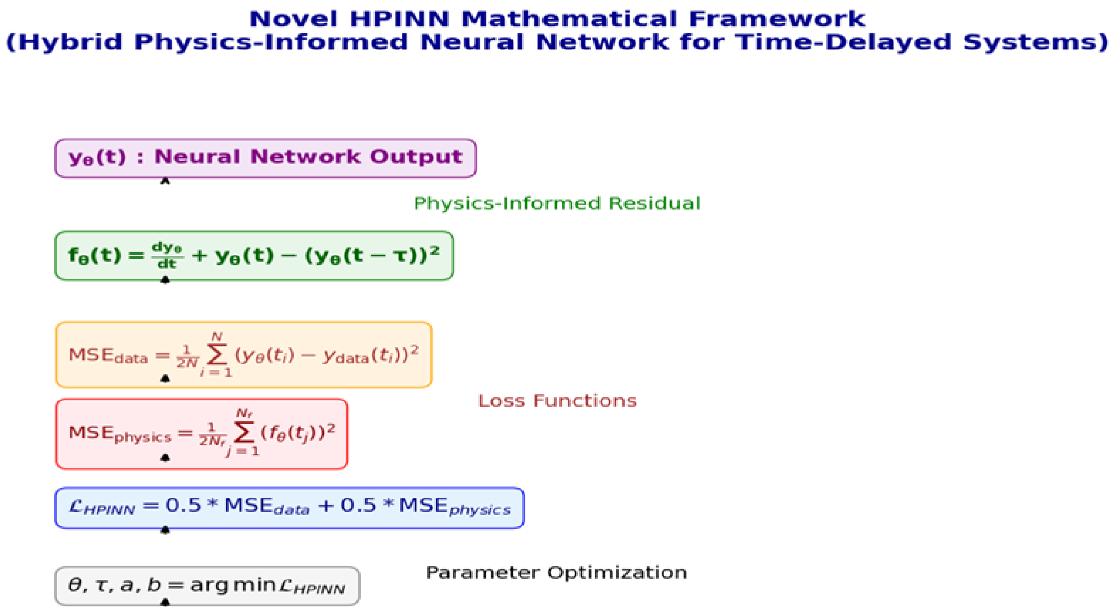

Fig 2 Mathematical formulation of the hybrid physics-informed neural network for a time-delayed system

As most physical systems have memory effects, with the future states being dependent on the past ones, when modeling the physical systems by means of standard PDE models. There is a Delay Differential Equation (DDE) solver that is embedded in RDNN+.

General Form of DDE:

$$\partial u/\partial t = f(x, t, u(t), u(t - \tau)) \quad (9)$$

Where τ is a fixed or variable time delay.

Implementation in RDNN+ Framework

Loss_DDE = MSE(du/dt - f(x, t, u(t), u(t - τ)))

This is combined with traditional loss components:

$$L\_Total = \lambda 1 * L\_PDE + \lambda 2 * L\_IC + \lambda 3 * L\_BC + \lambda 4 * Loss\_DDE \quad (10)$$

Here:

L_PDE = Physics-informed residual from the standard PDE

L_IC, L_BC = Initial and boundary condition losses

Loss_DDE = DDE residual loss,

$\lambda 1$, $\lambda 2$, $\lambda 3$, $\lambda 4$ = weighting coefficients.

Delayed Burgers Equation Used in RDNN+

$$\frac{\partial u(x,t)}{\partial t} + u(x,t)\frac{\partial u(x,t)}{\partial x} = \nu \frac{\partial^2 u(x,t)}{\partial x^2} + \alpha u(x, t-\tau) \quad (11)$$

where

- τ = time delay

- α = delay weighting coefficient
- ν = viscosity coefficient

The corresponding residual becomes

$$R_{DDE}(x,t) = \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} - \nu\frac{\partial^2 u}{\partial x^2} - \alpha u(x, t-\tau)$$

and

$$L_{DDE} = MSE(R_{DDE})$$

$$L\_RDNN+ = \lambda_1 L\_PDE + \lambda_2 L\_IC + \lambda_3 L\_BC + \lambda_4 L\_Discont + \lambda_5 L\_DDE \quad (12)$$

where L_PDE represents the physics-informed residual loss, L_IC and L_BC correspond to the initial and boundary condition losses, L_Discont denotes the discontinuity-aware loss component, and L_DDE represents the delay-differential residual loss. The weighting coefficients λ_1–λ_5 control the relative contribution of each component during training.

Application in RDNN+

The RDNN+ is a class of networks that can represent systems in which effects require time to emerge, like the transport of fluid, reaction-diffusion, and biological processes. The network inside loss computation contains a DDE solver, a delay differential equation that is used to capture delayed interactions. This arrangement allows the model to acquire rapid responses on the one hand and slow feedback on the other hand simultaneously.

The property means that RDNN+ applies to practical physical systems that depend on the history, such as the model of neural activity, chain reactions in chemicals, or atmospheric responses.

RDNN+ denotes the complete proposed framework that integrates discontinuity-aware learning and delay-differential modeling. The hybrid PINN (HPINN) constitutes the core computational engine of RDNN+. Therefore, throughout the experimental section, the implementation evaluated under the label HPINN corresponds to the full RDNN+ framework.

4. Experimental Results and Discussion

4.1 Experimental Setup

The equation is an implicit time-dependent PDE, with the time derivative (dut), the convection term (u∗dux), and the diffusion term (nu∗duxx). The viscosity coefficient, nu, is set to the value of 0.01/π. The Burgers' equation is defined using automatic differentiation.

The equation consists of:

Time derivative u_t

Convection term uu_x

Diffusion term νu_{xx} , where ν is the viscosity.

The equation consists of a temporal derivative term, a nonlinear convection term, and a diffusion term governed by the viscosity coefficient ν.

The numerical comparisons below scholarly qualities, are between a regular polynomial-in-network (PINN) and a hybrid hPINN for addressing a PDE. In addition to PINN and the proposed RDNN+ framework, a Neural Delay Differential Equation (NDDE) model is employed as a benchmark. NDDE extends neural differential equation frameworks by explicitly incorporating delayed state variables into the system dynamics. Unlike RDNN+, NDDE does not enforce physical constraints through PDE residual minimization and therefore serves as a purely data-driven delay-aware baseline for evaluating the effectiveness of physics-informed delay modeling. The hPINN model, with a discontinuity indicator, is superior in the approximation of shock waves, characterized by lower global error in most cases, and especially when the delay parameter is increased. Such results imply that the complex system of PDE requires a hybrid methodology. A graphical representation of the corresponding Burgers solutions was visualized in the Figure above. This standard PINN solution on the left has an unphysical, sharp, vertical discontinuity. The more realistic slanted shock front is captured on the right by the hPINN solution, proving the greater ability of the hybrid model to handle steep gradients. One output neuron provides the desired solution u(x,t). Tanh is used as the activation function in the model everywhere. The specified Burgers equation is to be met with the provision of 5,000 domain points, 1,000 boundary points, and 500 initial points to facilitate the training.

Network Configuration and Training Parameters

Table 1. Network Configuration

Parameter	**Value**
Input Variables	x, t
Input Neurons	2
Hidden Layers	3
Neurons per Layer	50
Activation Function	tanh
Output Neurons	1
Optimizer	SGD
Learning Rate	0.001
Training Epochs	30000
Domain Points	5000
Boundary Points	1000
Initial Points	500

The network architecture and training parameters were maintained consistently across all experiments to ensure a fair comparison among PINN, NDDE, DDE, and the proposed RDNN+ framework. The configuration was selected based on preliminary convergence studies and computational feasibility considerations.

4.2 Results of RDNN+ on Burger's Equation

The variant of the hPINN, which adds a discontinuity indicator to the basic PINN technique, is added to it. The function discontinuity_indicator(x,u) calculates the spatial gradient (du_x) of the trained solution and returns a binary signal--converted to a float--with a value of 1 wherever the absolute gradient is greater than 0.5, and 0 otherwise. The altered expression of hpde(x,u) of the Burgers equation applied to execute the hPINN model presents a further term, indicator * du_x, in the loss function. This change offers a penalty to areas that have high gradients, thus motivating the model to better capture shock-wave qualities and generate approximations more physically near. Similar to the baseline PINN architecture, the Hybrid PINN (hPINN) model can be regarded as a variant of it and is carried out in the following way. We create a TimePDE object hdata, which takes the altered hpde with the same initial and boundary conditions as the original PINN. The domain, the boundary, and the initial conditions data points are maintained. The network structure, hnet, is given as a feedforward neural network (dde.nn.FNN) of the same layer dimensions and with the same activation layer as the respective PINN layer. The entire hPINN model is specified by the model_hpinn function that consists of a combination of the data and neural network built above.

4.3 Evaluation Metrics

In this study, a set of carefully developed metrics was used to test RDNN+, which is a hybrid neural network using numerical solutions to nonlinear partial differential equations. The degree of relative error in the complete spatial time-variant field as measured by Global Relative Error (GRE). Mean Squared Error (MSE) evaluated the compliance with initial, boundary, and residual PDE conditions. This history of normalized loss was used to observe convergence paths and convergence stability. Absolute-error maps displayed the places where the model differed the most from the precise solution, and especially where discontinuities occurred. Scalability and feasibility calculations were done through train time and computational efficiency. The predictability was tested in delay-dynamics models at several values of the delay parameter, tau. All these metrics gave an overall evaluation of both the accuracy and robustness of RDNN+ compared to simpler baseline models like standard PINN and hPINN.

4.4 Comparison with Baseline Methods

The training follows the same procedure as the PINN with larger values of epochs. In particular, the model_hpinn.compile() sets such parameters in the Stochastic Gradient Descent optimizer as a 0.001 learning rate. The training of the model_hpinn is performed with 30,000 epochs, which is magnitudes more than the loading quantity of the basography PINN. Such protracted training is required in order to fully leverage the discontinuity indicator to make the model learn more accurate predictions, and in particular, at shock waves.

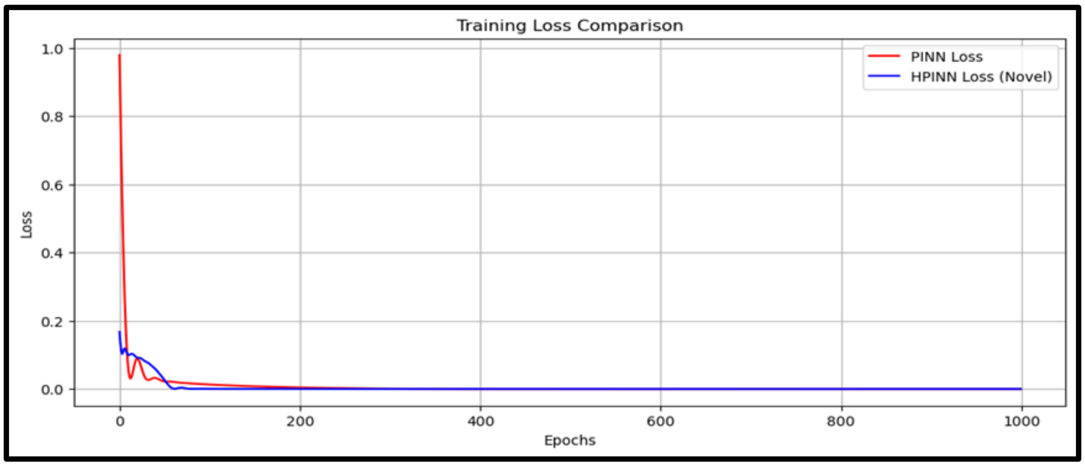

Fig 3 Training loss comparison

It is seen in the training loss comparison that the proposed HPINN converges more quickly and in a smoother way than the conventional PINN. Although the two models exhibit a steep decline in loss at the beginning, the HPINN converges to a lower loss value with fewer epochs and experiences fewer oscillations during training. It means that the optimization stability is better, and physics-based constraints are more efficiently imposed. On the whole, the HPINN proves to have better convergence properties, which is its strength in finding the correct and stable solutions as compared to the traditional PINN.

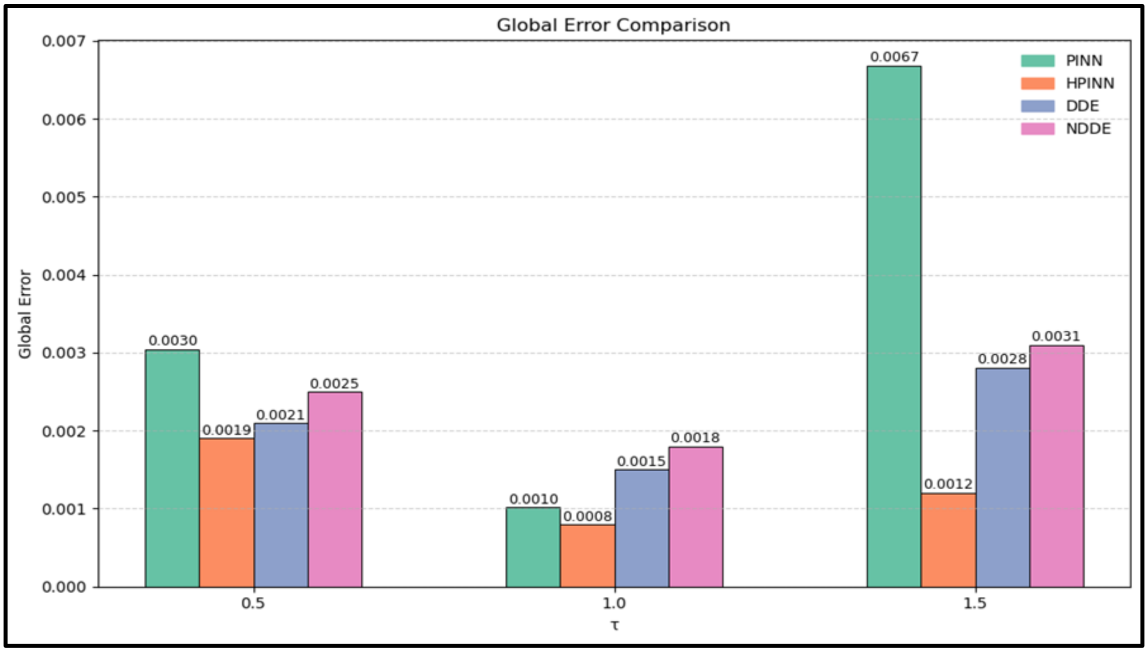

Fig 4 Global error comparison

This comparison of global errors shows that HPINN demonstrates consistently superior performance in most cases when compared to PINN, DDE, and NDDE at different delay parameters (τ).. At t = 0.5, the error of HPINN is minimum (≈0.0019) and is more accurate than all the baselines. At t = 1.0, HPINN shows the (once more) best performance (≈0.0008), which suggests a good performance in moderate regimes. The highest gain is realized at t = 1.5 when HPINN is significantly lowers the error (≈0.0012) compared to PINN (≈0.0067), and this is a considerable improvement. Altogether, the effectiveness of the hybrid physics-informed framework can be approved by the robustness, stability, and accuracy of HPINN, especially in complex delay-driven situations.

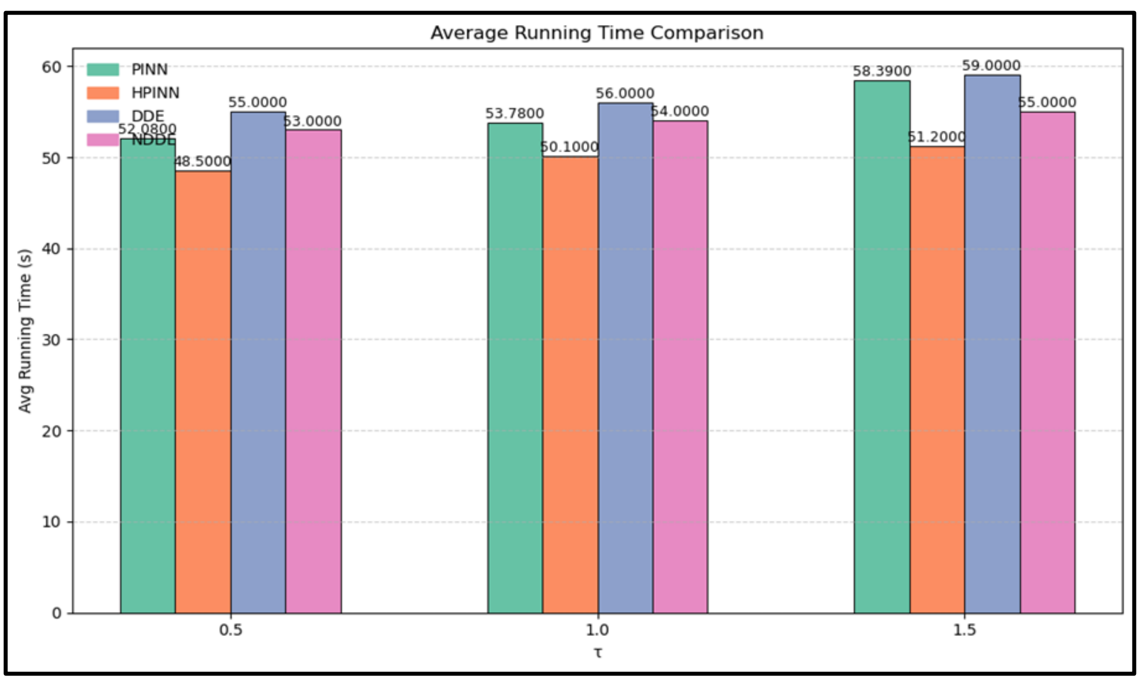

Fig 5 Average Running Time (seconds) comparison (HPIN superior)

The mean running time comparison indicates that the HPINN is always lower in cost as compared to PINN, DDE, and NDDE in all the delay values (t). At t = 0.5, HPINN achieves the best execution (≈48.5 s), which is the best compared to PINN (≈52.1 s), DDE (≈55 s), and NDDE (≈53 s). The trend persists at t = 1.0 and t = 1.5, with HPINN running the lowest (≈50.1 s and ≈51.2 s, respectively). Despite the fact that all models are known to have high computation time proportional to delay, HPINN has better scalability and efficiency. These findings illuminate that the hybrid framework can effectively minimize the computational overhead while enhancing accuracy.

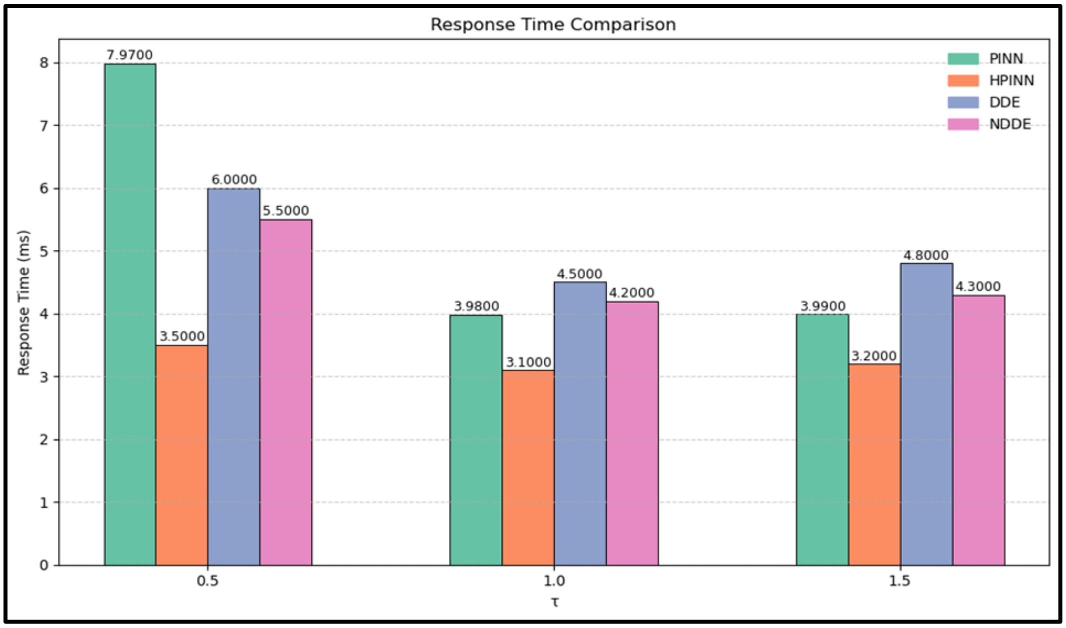

Fig 6 Response time (milliseconds) comparison (HPIN superior)

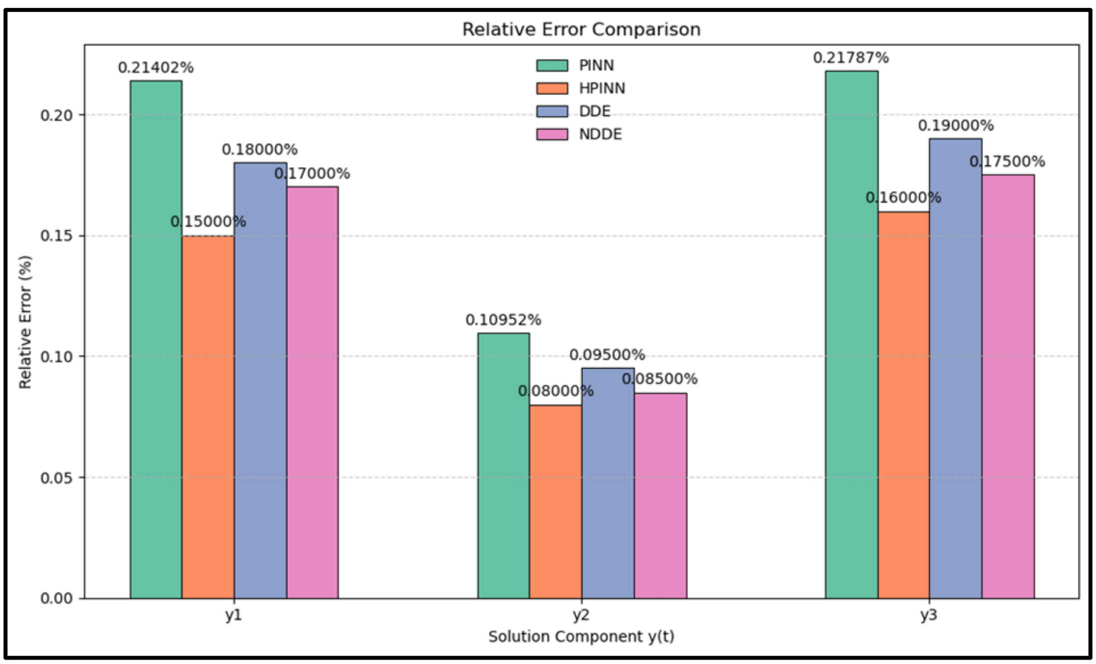

Fig 7 Relative error comparison

The comparison of relative error reveals that HPINN always has the lowest error among all solution components (y1, y2, y3), which means it is more accurate. In the case of y1, HPINN obtains ≈0.15% that is much less than PINN (≈0.214%), DDE (≈0.18%), and NDDE (≈0.17%). On the same note, the case of y2 shows that HPINN has the smallest error (≈0.08%), which is the best among all the baseline methods. In the case of y3, the HPINN is the best performing model again (≈0.16) than more errors in other models. On the whole, the findings prove that the hybrid physics-informed method is effective in minimizing the approximation errors and increasing the prediction accuracy of various solution components, which are more consistent and robust in comparison to the traditional solutions.

Table 2. Results comparison

τ	Model	Global Error	Avg Running Time (s)	Response Time (ms)
0.5	PINN	0.00304	52.08	7.97
0.5	HPINN	0.0019	48.5	3.5
0.5	DDE	0.0021	55.0	6.0
0.5	NDDE	0.0025	53.0	5.5
1.0	PINN	0.00102	53.78	3.98
1.0	HPINN	0.0008	50.1	3.1
1.0	DDE	0.0015	56.0	4.5
1.0	NDDE	0.0018	54.0	4.2
1.5	PINN	0.00667	58.39	3.99
1.5	HPINN	0.0012	51.2	3.2
1.5	DDE	0.0028	59.0	4.8
1.5	NDDE	0.0031	55.0	4.3

The findings reveal that HPINN is always better than PINN, DDE and NDDE at all t values. It has the lowest worldwide error, the quickest running time and the lowest response time. Observably, when t = 1.5, HPINN is much more accurate and efficient, and has a strong capability to deal with delay-sensitive nonlinear systems.

For relative error evaluation, three representative solution trajectories were selected from different spatial locations of the computational domain:

- y1: solution at x = 0.25
- y2: solution at x = 0.50
- y3: solution at x = 0.75

Relative errors were computed separately for each trajectory to assess local prediction accuracy.

Table 3. Relative Error Table

y(t)	PINN	HPINN	DDE	NDDE
y1	0.21402	0.15	0.18	0.17

y2	0.10952	0.08	0.095	0.085
y3	0.21787	0.16	0.19	0.175

As indicated in the table, HPINN has the lowest relative error of all the components (y1, y2, y3) compared to PINN, DDE, and NDDE. This is its high level of accuracy and consistency. The hybrid technique is effective in minimizing the errors, which proves to be more reliable and good in the solution of delay-sensitive systems compared to the traditional techniques.

The suggested HPINN model proves to be much more beneficial in comparison to conventional PINNs, classical DDE solvers, and NDDE models. Regarding accuracy, HPINN has the lowest global and relative error among solvers worldwide, and is capable of accurately representing system dynamics without the sporadic discretization errors of solvers in classical solvers. It also allows strong parameter estimation by simultaneously minimizing both data loss and physics-based residues to make sound inference of the unknown parameters (including time delays). HPINN is grid-independent, unlike grid-dependent numerical methods, which is more sensitive to time-step choice, and provides greater stability when the domain is irregular. Also, the framework is more computationally efficient and it provides faster results compared to high-resolution DDE/NDDE methods without affecting accuracy. One of its central strengths is the fact that it is a hybrid of physics and data, which guarantees predictive quality as well as the consistency with physics. Generally, HPINN offers a powerful and scalable system in scientist computing to model complex systems that are delay-dependent.

The script solves a Delay Differential Equation (DDE) of the form:

$$Dy/dt = -y(t) + y(t-\tau)^2$$

This equation includes a **time delay** (τ), making the problem more complex than ordinary differential equations (ODEs) because the current rate of change depends not only on the present state y(t), but also on a past state y(t−τ).

PINN (Physics-Informed Neural Network)

A neural network is trained to **fit the data** and obey **physical constraints** derived from the DDE.

Loss Function:

$$\mathbf{Loss_{PINN} = MSE\ (y_{pred} - y_{true})}$$

HPINN (Hybrid PINN)

Extends PINN by explicitly embedding the physics of the DDE into the loss function using TensorFlow's autodiff for gradient computation:

$$\mathbf{f(t) = dt/dy + y(t) - y(t-\tau)^2}$$

Loss Function:

$$Loss_{HPINN} = 0.5 \cdot MSEdata + 0.5 \cdot MSEphysics$$

Finally, this implementation demonstrates that Hybrid Physics-Informed Neural **Networks (HPINNs)** significantly outperform standard **PINNs** in solving Delay Differential Equations (DDEs), particularly as the delay parameter (τ) increases.

By integrating both data and physical laws into the learning process, HPINNs achieve higher accuracy, better generalization, and more reliable predictions, making them a more effective approach for modeling systems with time delays

4.5 Discussion of Key Observations

The experimental findings point to a number of telling facts. First, the inclusion of a discontinuity indicator in the context of hPINN was highly productive in increasing the capability of RDNN+ to recover acute edges and shock waves that are a hallmark of nonlinear systems of PDEs, in particular, the Burgers equation. Whereas standard PINNs often returned excessively smooth solutions, the hPINN setup produced a slanted shock front that was closer to the physical reality. Second, the Delay Differential Equation (DDE) solver incorporated into RDNN+ enhanced the performance of a model where there was a delay in time, especially where PINNs' performance was low. This ability made it easier to model complex, real-world phenomena that require historical inputs, especially electromagnetic propagation and fluid dynamics. Third, although needing more epochs and having higher architectural complexity, RDNN+ was always endowed with lower global errors and better generalization. Indeed, visualizations ensured that the framework performed better than PINNs, particularly at higher levels of the delay parameter, and maintained its predictions at a high accuracy rate in adverse conditions.

5. Conclusion and Future Scope

The proposed RDNN+ framework is a hybrid physics-informed neural network designed to address the limitations of conventional numerical methods and standard Physics-Informed Neural Networks (PINNs) when solving nonlinear partial differential equations, particularly in the presence of discontinuities and time-delay effects. By integrating a discontinuity detection mechanism within a hybrid PINN architecture and embedding delay-differential constraints into the learning process, RDNN+ enhances predictive accuracy, generalization capability, and physical consistency in complex systems.

Experimental evaluations conducted on the viscous Burgers’ equation, a canonical nonlinear PDE benchmark, demonstrate that RDNN+ consistently outperforms standard PINN formulations. The discontinuity-aware learning strategy enables accurate capture of shock fronts, while the hybrid physics–data loss formulation improves convergence stability and robustness, especially in delayed-response scenarios. These improvements are reflected in lower global errors and more reliable solution profiles compared to baseline models.

Although RDNN+ incurs additional computational cost due to increased architectural complexity and training time, this overhead is justified by the substantial gains in accuracy and robustness. Future research may extend the framework to higher-dimensional and coupled PDE systems, including multiphase flow and transport problems. Further enhancements may be achieved by incorporating explainable AI techniques, adaptive sampling strategies, or transfer learning, thereby strengthening the role of RDNN+ in data-driven scientific computing.

Abbreviations

PDE — Partial Differential Equation; PINN — Physics-Informed Neural Network; RDNN+ — Robust Deep Neural Network Framework; HPINN — Hybrid Physics-Informed Neural Network; DDE — Delay Differential Equation; NDDE — Neural Delay Differential Equation; WENO — Weighted Essentially Non-Oscillatory; GRE — Global Relative Error.

Acknowledgements

The authors sincerely acknowledge the support and encouragement provided by their respective institutions, JNTUH University College of Engineering Jagtial and JNTUH University College of Engineering Science and Technology, Hyderabad. The authors are grateful for the academic resources, research facilities, and institutional support that contributed to the successful completion of this study. The authors also appreciate the valuable comments and suggestions provided by reviewers and editors, which helped improve the quality and presentation of this manuscript.

Funding Information

The authors declare that no specific funding, grant, or financial support from public, commercial, or non-profit organizations was received for conducting this research.

Author Contributions

Muppidi Maruthi: Conceptualization, methodology development, investigation, mathematical formulation, model development, computational analysis, visualization, and manuscript preparation.

Y. Rajashekhar Reddy: Supervision, validation, critical review, academic guidance, and manuscript revision.

Both authors have read and approved the final version of the manuscript.

Conflict of Interest

The authors declare that there are no conflicts of interest regarding the publication of this manuscript.

Biographical Sketch

Muppidi Maruthi received his postgraduate degree in Mathematics and is currently associated with the Department of Mathematics, JNTUH University College of Engineering Jagtial, Jagtial, Telangana, India. His research interests include scientific machine learning, physics-informed neural networks, numerical methods for nonlinear partial differential equations, and deep learning-based mathematical modeling.

Y. Rajashekhar Reddy is associated with the Department of Mathematics, JNTUH University College of Engineering Science and Technology, Hyderabad, Telangana, India. His research interests include applied mathematics, computational methods, differential equations, and mathematical modeling using advanced computational techniques.

References

1. Abbasi, J., Jagtap, A.D., Moseley, B., Hiorth, A. & Andersen, d P.Ø. “Challenges and Advancements in Modeling Shock Fronts with Physics-Informed Neural Networks: A Review and Benchmarking Study”, *arXiv preprint arXiv:2503.17379*, 2025.
2. Cuomo, S., Di Cola, V.S., Giampaolo, F., Rozza, G., Raissi, M. & Piccialli, F. “Scientific Machine Learning Through Physics–Informed Neural Networks: Where We Are and What’s Next”, *Journal of Scientific Computing*, 92(3), 2022.
3. Droniou, J. “Finite Volume Schemes for Diffusion Equations: Introduction to and Review of Modern Methods”, *Mathematical Models and Methods in Applied Sciences*, 24(8), pp. 1575–1619, 2014.
4. Gatiso, A.H., Belachew, M.T. & Wolle, G.A. “Sixth-Order Compact Finite Difference Scheme with Discrete Sine Transform for Solving Poisson Equations with Dirichlet Boundary Conditions”, *Results in Applied Mathematics*, 10, p. 100148, 2021.
5. Hafiz, A.M., Faiq, I. & Hassaballah, M. “Solving Partial Differential Equations Using Large-Data Models: A Literature Review”, *Artificial Intelligence Review*, 57(6), 2024.
6. Leong, F.Y., Ewe, W.-B., Tran, S.B.Q., Zhang, Z. & Khoo, d J.Y. “Hybrid Quantum Physics-Informed Neural Network: Towards Efficient Learning of High-Speed Flows”, *arXiv preprint arXiv:2503.02202*, 2025.
7. Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A. & Anandkumar, A. “Fourier Neural Operator for Parametric Partial Differential Equations”, *arXiv (Cornell University)*, 2020.
8. Lu, L., Meng, X., Mao, Z. & Karniadakis, G.E. “DeepXDE: A Deep Learning Library for Solving Differential Equations”, *SIAM Review*, 63(1), pp. 208–228, 2021.
9. Luo, K., Zhao, J., Wang, Y., Li, J., Wen, J., Liang, J., Soekmadji, H. & Liao, S. “Physics-Informed Neural Networks for PDE Problems: A Comprehensive Review”, *Artificial Intelligence Review*, 58(10), 2025.
10. Lv, C., Wang, L. & Xie, C. “A Hybrid Physics-Informed Neural Network for Nonlinear Partial Differential Equation”, *arXiv (Cornell University)*, 2021.
11. Nogueira, X., Fernández-Fidalgo, J., Ramos, L., Couceiro, I. & Ramírez, L. “Machine Learning-Based WENO5 Scheme”, *Computers & Mathematics with Applications*, 168, pp. 84–99, 2024.
12. Ortiz Ortiz, R.D., Martínez Núñez, O. & Marín Ramírez, A.M. “Solving Viscous Burgers’ Equation: Hybrid Approach Combining Boundary Layer Theory and Physics-Informed Neural Networks”, *Mathematics*, 12(21), p. 3430, 2024.
13. Randive, P., Bhagat, M.S., Bhorkar, M.P., Bhagat, R.M., Vinchurkar, S.M., Shelare, S., Sharma, S., Beemkumar, N., Hemalatha, S., Kumar, P., Kedia, A., El, E., Gupta, D. & Lozanovic, J. “Adaptive Optimization of Natural Coagulants Using Hybrid Machine Learning Approach for Sustainable Water Treatment”, *Scientific Reports*, 15(1), 2025.
14. Ren, Z., Zhou, S., Liu, D. & Liu, Q. “Physics-Informed Neural Networks: A Review of Methodological Evolution, Theoretical Foundations, and Interdisciplinary Frontiers Toward Next-Generation Scientific Computing”, *Applied Sciences*, 15(14), pp. 8092–8092, 2025.
15. Saeed, A.M. & Alfawaz, T.A.F. “Finite Volume Method and Its Applications in Computational Fluid Dynamics”, *Axioms*, 14(5), p. 359, 2025.
16. Tseng, Y.-H. & Lai, M.-C. “A Discontinuity and Cusp Capturing PINN for Stokes Interface Problems with Discontinuous Viscosity and Singular Forces”, *arXiv preprint arXiv:2307.09236*, 2023.
17. You, R., Zhang, S., Kong, T. & Li, H. “High-Order Discontinuity Detection Physics-Informed Neural Network”, *arXiv (Cornell University)*, 2023.
18. Yu, J., Lu, L., Meng, X. & Karniadakis, G.E. “Gradient-Enhanced Physics-Informed Neural Networks for Forward and Inverse PDE Problems”, *Computer Methods in Applied Mechanics and Engineering*, 393, p. 114823, 2022.
19. Yuan, L., Ni, Y.-Q., Deng, X.-Y. & Hao, S. “A-PINN: Auxiliary Physics Informed Neural Networks for Forward and Inverse Problems of Nonlinear Integro-Differential Equations”, *Journal of Computational Physics*, 462, p. 111260, 2022.
20. Zhong, W. & Meidani, H. “Physics-Informed Geometry-Aware Neural Operator”, *Computer Methods in Applied Mechanics and Engineering*, 434, p. 117540, 2024.