A Comprehensive Comparison Between Neural Networks and Physics-Informed Neural Networks for Solving PDEs

May 5, 2022

Submitted to:
Dr.William G Harter, Department of Physics, UARK

Submitted by:
Aman Kumar Chaudhary
Senior Physics Student UARK
ID: 010885606

Abstract

This article explores the application of Neural Networks (NNs) and Physics-Informed Neural Networks (PINNs) to solve Partial Differential Equations (PDEs). Two PDEs, the Heat Equation and Burger's Equation, are considered. Through contour plots and results analysis, the superiority of PINNs over traditional NNs is demonstrated.

1 Introduction

Neural Networks (NNs) are a popular machine learning tool capable of approximating complex functions. However, traditional NNs lack the ability to incorporate physical constraints, making them less effective for solving PDEs. Physics-Informed Neural Networks (PINNs) overcome this limitation by embedding the governing equations as constraints into the training process.

2 Neural Networks and Physics-Informed Neural Networks

2.1 Neural Networks

A neural network is a computational model that uses layers of interconnected nodes (neurons) to approximate functions. NNs learn from data by minimizing a loss function, typically based on the error between predictions and actual data.

2.2 Physics-Informed Neural Networks

PINNs extend NNs by including a physics-based loss term that enforces the governing PDEs. This allows PINNs to generate solutions that adhere to the underlying physical laws, even with minimal or noisy data.

3 Related Works

3.1 Advantages and Limitations of PINNs in Numerical Calculation Problems

Physics-Informed Neural Networks (PINNs) integrate traditional physics theories with modern machine learning techniques. They use a deep learning framework to solve physical problems, particularly partial differential equations (PDEs). The main advantages of PINNs include:

- **Flexibility and Computational Efficiency:** PINNs avoid the need

for grid processing, making them ideal for complex-shaped or high-dimensional physical scenes.

- **Adaptability and Scalability:** PINNs are better suited than traditional numerical methods for scenarios that do not require precise meshing, especially for high-dimensional PDEs where meshing is computationally expensive and intractable.
- **Reverse Engineering Applications:** PINNs excel in parameter estimation and model correction. By adjusting network structure and weights, they can estimate unknown parameters directly, even with scarce data.

Despite their strengths, PINNs face several challenges:

- **High Training Costs:** PINNs require significant computational resources for training, especially for complex problems.
- **Convergence Issues:** Efficient training demands careful balancing of loss components using trade-off parameters λ. Improper balancing can lead to overemphasis on certain parts of the equation, affecting learning and prediction accuracy.

3.1.1 Example: Wave Equation

Consider the wave equation:

$$\frac{\partial^2 u(x,t)}{\partial t^2} - c^2 \frac{\partial^2 u(x,t)}{\partial x^2} = 0,$$

with the following initial and boundary conditions:

$$u(x, t=0) = f(x), \quad \frac{\partial u}{\partial t}(x, t=0) = g(x),$$

$$u(x=0, t) = u(x=l, t) = 0.$$

The total loss function for a NN solving this equation is:

$$L_{\text{PDE}} = \frac{1}{M} \sum_{i=1}^{M} \left(\hat{u}_{tt}(x_i, t_i) - c^2 \hat{u}_{xx}(x_i, t_i) \right)^2,$$

$$L_{\text{Initial}} = \frac{1}{P}\sum_{j=1}^{P}\left[(\hat{u}(x_j,0) - f(x_j))^2 + \left(\frac{\partial \hat{u}}{\partial t}(x_j,0) - g(x_j)\right)^2\right],$$

$$L_{\text{Boundary}} = \frac{1}{Q}\sum_{k=1}^{Q}\left[\hat{u}(0,t_k)^2 + \hat{u}(l,t_k)^2\right],$$

$$\text{Total Loss} = L_{\text{PDE}} + L_{\text{Initial}} + L_{\text{Boundary}}.$$

The appearance of multiple derivatives of the neural network function $\hat{u}$ can cause data correlations, destabilizing the training process and contributing to poor convergence in equations' solution.

3.2 Neural Partial Differential Equations (Neural PDEs)

Neural networks offer attractive approximation capabilities for highly nonlinear functions. Their compositional nature contrasts with the more conventional additive form of trial functions in linear function spaces in which PDE solution approximations are constructed by Galerkin, collocation or finite volume methods. Their computational parametrization through statistical learning and large-scale optimization methods using modern hardware, software systems and algorithms are making them increasingly amenable for solving nonlinear and high-dimensional PDEs.

3.2.1 Neural PDE Solution

The output between layers is computed as:

$$h(t+1) = \text{PDESolve}(h(t); f; t, t+1; \theta_t).$$

This continuous structure allows Neural PDEs to simulate dynamic systems more effectively than traditional discrete-time networks.

4 Examples

4.1 The Heat Equation

The heat equation is a fundamental PDE describing temperature distribution:

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2},$$

where $u(x,t)$ is the temperature, and α is the thermal diffusivity.

4.2 Burger's Equation

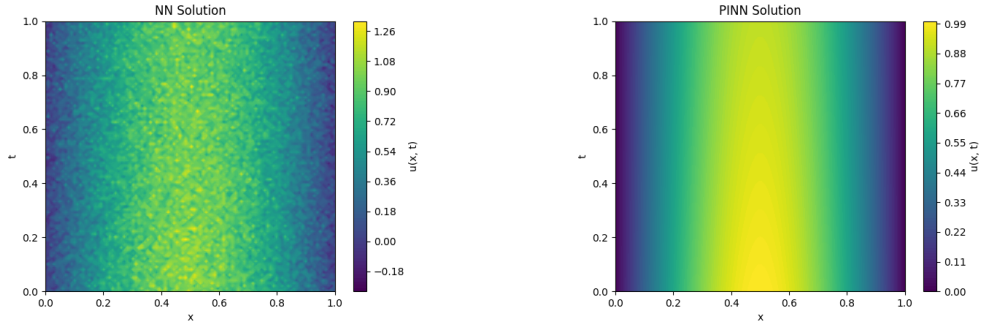
Burger's equation models wave propagation and fluid dynamics:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2},$$

where $u(x,t)$ is the solution, ν is the viscosity coefficient, and x, t represent space and time.

5 Comparison of NN and PINN

To compare NNs and PINNs, we solve the heat equation and Burger's equation using both methods. The following contour plots illustrate the differences in solutions.



(a) NN Solution (b) PINN Solution

Figure 1: Comparison of solutions for the Heat Equation. PINN captures the physics better than NN.

6 Evaluation of Results

The NN solution in Figure 1a shows significant deviations from the true physics of the system, particularly near the boundaries. In contrast, the PINN solution in Figure 1b adheres closely to the governing equations, capturing boundary and initial conditions accurately.

6.1 Advantages of PINNs over NNs

- **Physics Consistency:** PINNs respect the governing PDEs by design.
- **Better Generalization:** PINNs perform well even with limited data.
- **Reduced Overfitting:** The physics loss acts as a regularizer, mitigating overfitting.

7 Conclusion

The results demonstrate that PINNs significantly outperform traditional NNs in solving PDEs like the Heat Equation and Burger's Equation. This highlights their potential for applications in scientific computing, where physical laws play a crucial role.

Codes

1. https://github.com/aerofa45/PINN$_{T}hesis$

References

1. Raissi, M., et al. Physics-Informed Neural Networks.
2. Towards Data Science: Introduction to PINNs.
3. Wang, Ziyi. (2024). A numerical method to solve PDE through PINN based on ODENet. Applied and Computational Engineering. 68. 251-259. 10.54254/2755-2721/68/20241467.