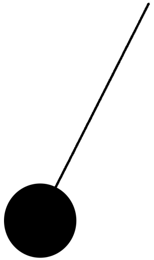
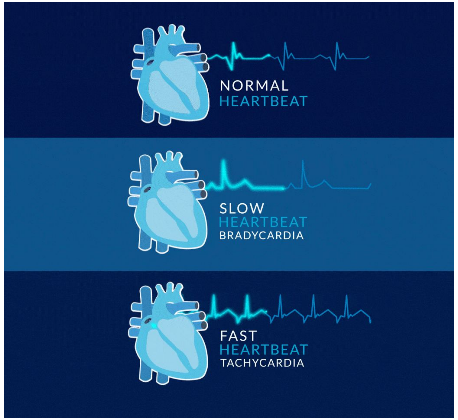
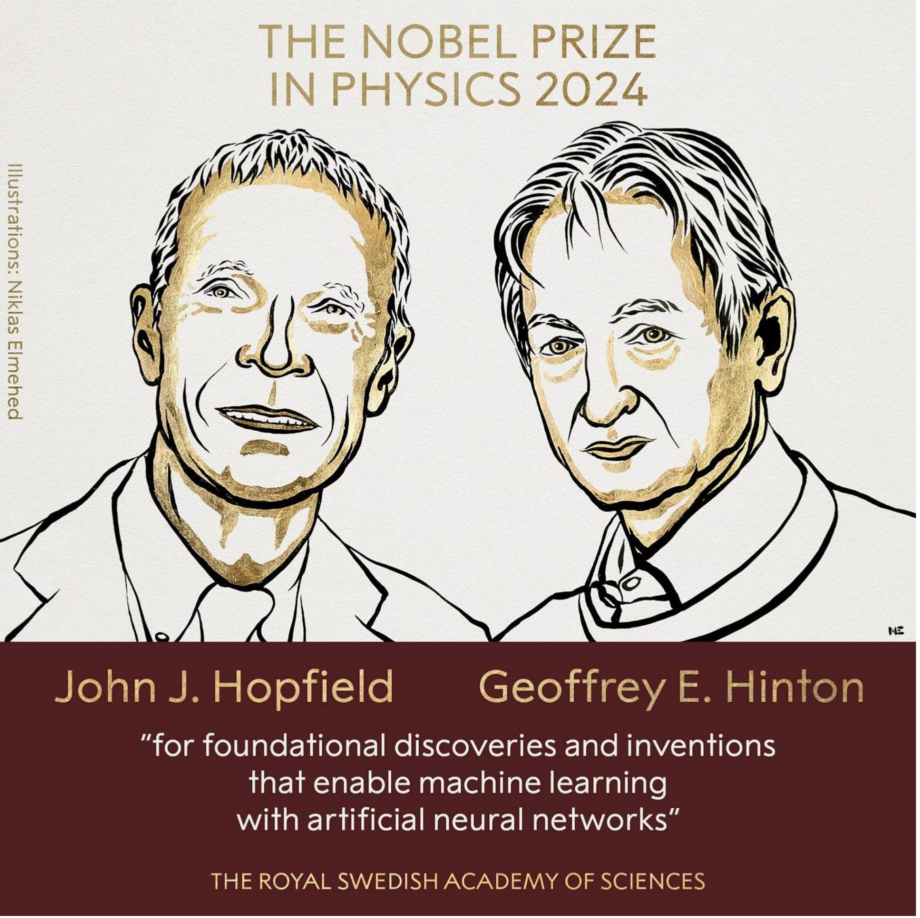
Physics Informed Neural Networks

PINN: An Overview

What Are Dynamical Systems and Why Do They Matter?

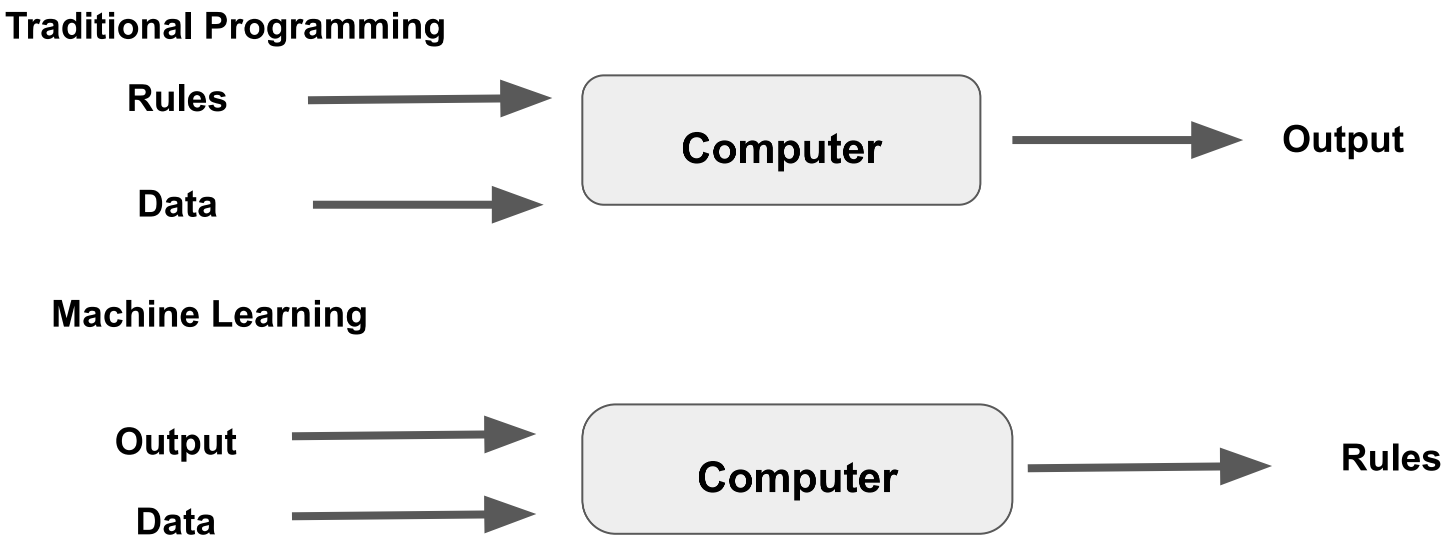




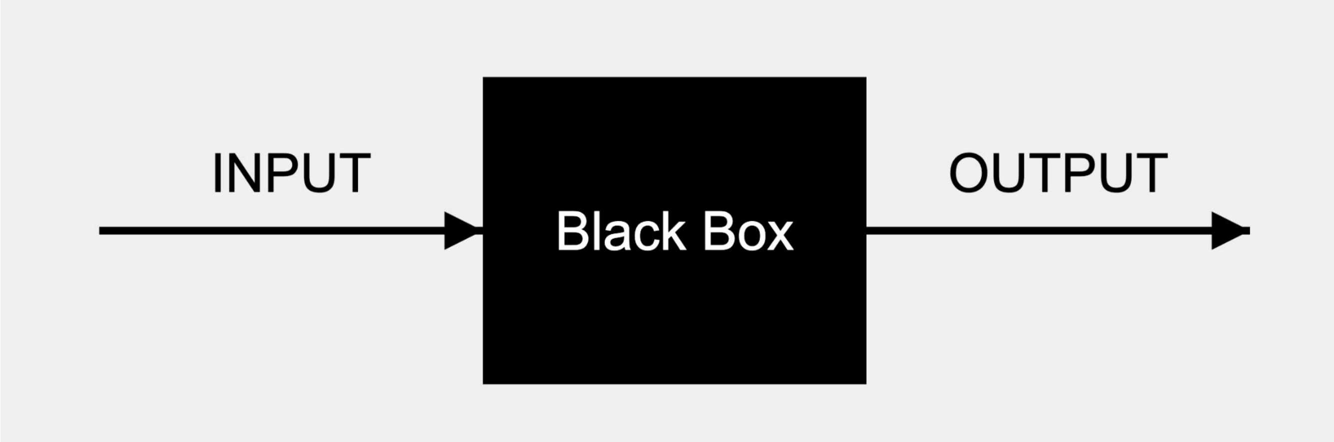

THE NOBEL PRIZE
IN PHYSICS 2024
Illustrations: Niklas Elmehed
John J. Hopfield
Geoffrey E. Hinton
"for foundational discoveries and inventions
that enable machine learning
with artificial neural networks"
THE ROYAL SWEDISH ACADEMY OF SCIENCES


THE NOBEL PRIZE
IN CHEMISTRY 2024
Illustrations: Niklas Elmehed
David
Baker
Demis
Hassabis
John M.
Jumper
"for computational
protein design"
"for protein structure prediction"
THE ROYAL SWEDISH ACADEMY OF SCIENCES

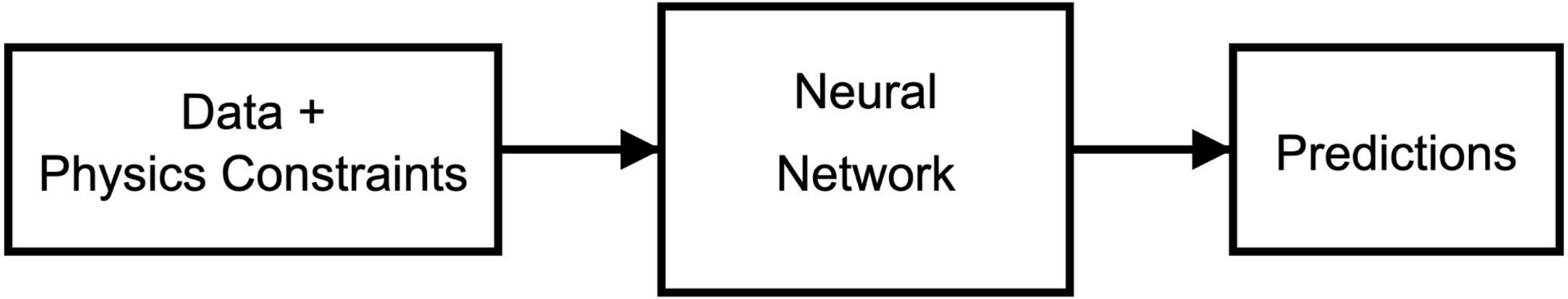
MACHINE LEARNING



Why Do We Need Better Models?

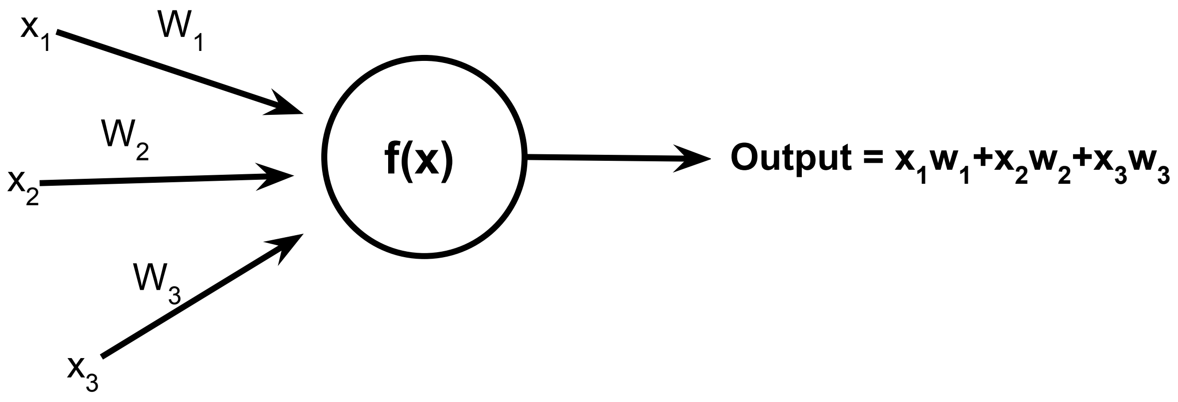


Introducing Physics-Informed Neural Networks (PINNs)



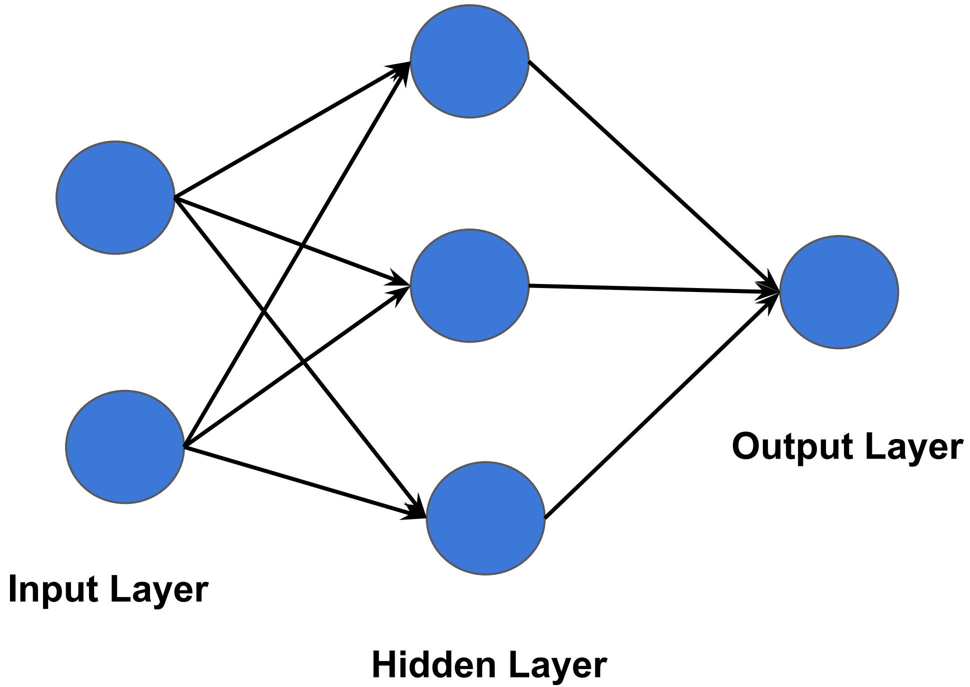
Types

- Supervised Learning
- Unsupervised Learning
- Semi-Supervised Learning

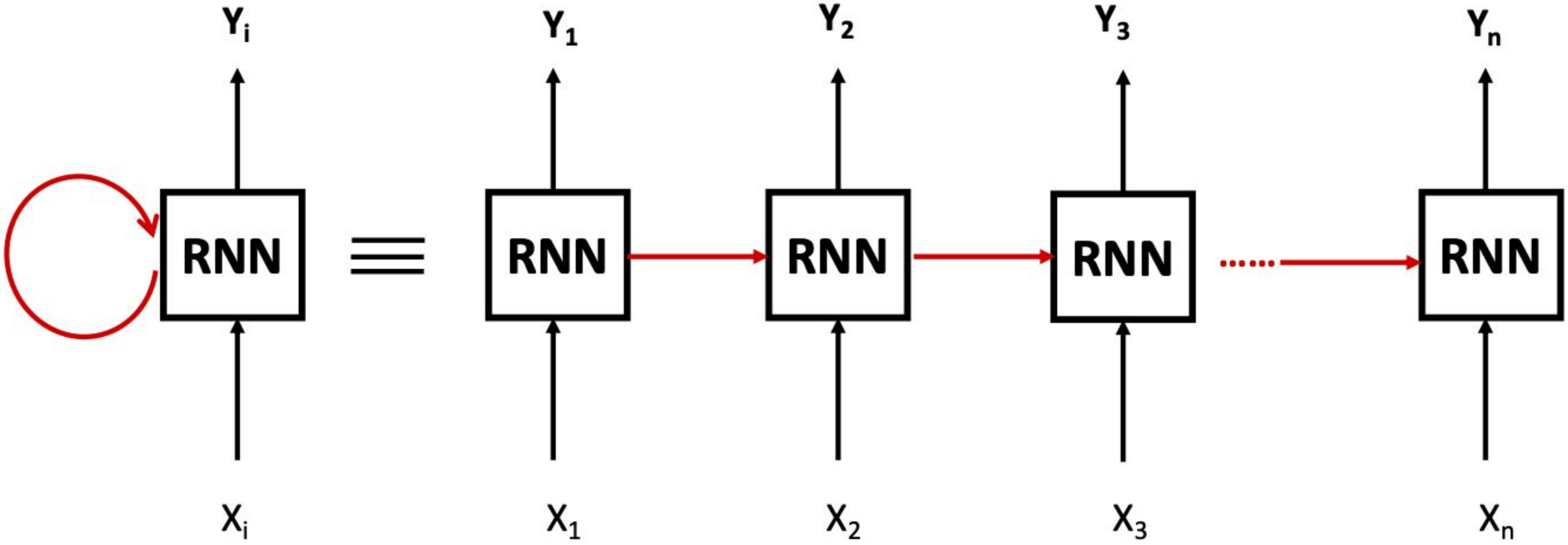


x1
W1
W2
x2
f(x)
W3
x3
Output = x1w1+x2w2+x3w3

FNN - FeedForward Neural Network



RNN- Recurrent Neural Network



Output

$$y_i^{(1)} = \sigma(\sum_{j=0}^{n} w_{ij}^{(1)} x_j + b_i^{(1)})$$

$$Y^{(1)} = W^{(1)} X + B^{(1)}$$

$$Y^{(2)} = W^{(2)} (W^{(1)} X + B^{(1)}) + B^{(2)}$$

$$FNN(\theta, x) = f^{(n)} \circ f^{(n-1)} ... f^{(3)} \circ f^{(2)} \circ f^{(1)}(x)$$

$$RNN(\theta, h, x) = f \circ f ... f \circ f \circ f(h, x)$$

LOSS →

$$L1(\theta) = \frac{1}{N} \sum_{x \in \mathbb{X}} |y - y^{'}|$$

$$L2(\theta) = \frac{1}{N} \sum_{x \in \mathbb{X}} (y - y^{'})^2$$

$$w^{new} = w^{old} - \eta \frac{\partial L}{\partial w^{old}}$$

ACTIVATION FUNCTIONS →

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad ReLU(x) = max(0, x)$$

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad softmax(x_i) = \frac{e^{(x_i)}}{\sum_{j=1}^{n} e^{x_j}}$$

Romanian Reports in Physics **74**, 113 (2022)

PRIMER ON SOLVING DIFFERENTIAL EQUATIONS USING MACHINE LEARNING TECHNIQUES

TAMIL ARASAN BAKTHAVATCHALAM[1,2,a], SELVAKUMAR MURUGAN[2,b], SURIYADEEPAN RAMAMOORTHY[2,c], MALAIKANNAN SANKARASUBBU [2,d], RADHA RAMASWAMY[3,e], VIJAYALAKSHMI SETHURAMAN[1,f], BORIS A. MALOMED [4,g]

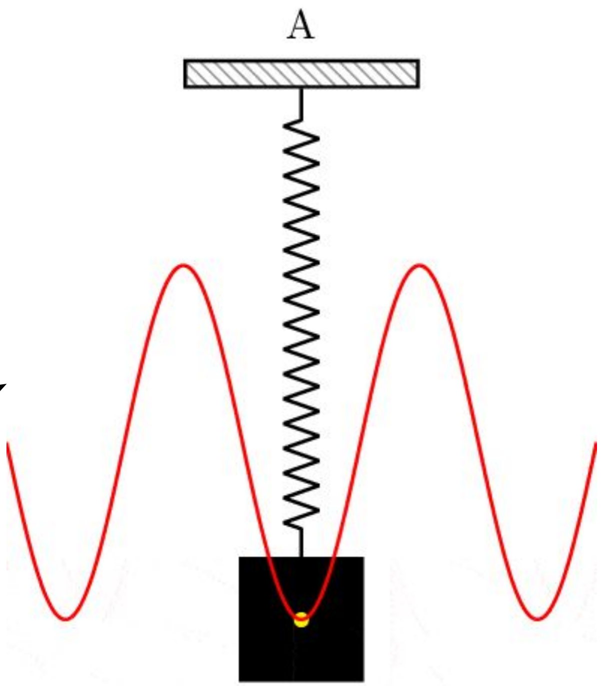
Simple Harmonic Oscillator

$$m\frac{d^2x}{dt^2} = -kx,$$

$$\frac{d^2x}{dt^2} + \omega^2 x = 0,$$

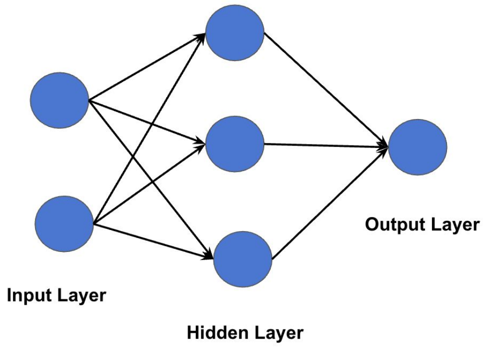
$$x(t) = A \sin \omega t + B \cos \omega t$$

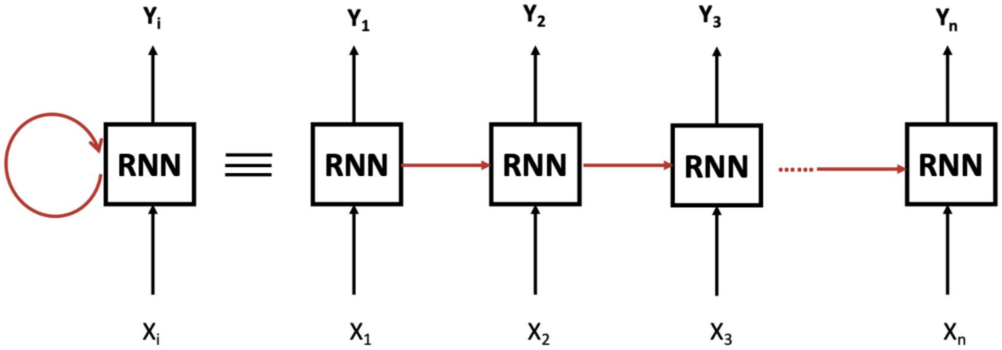
$$x(t) = \cos(\omega t)$$



Neural Network

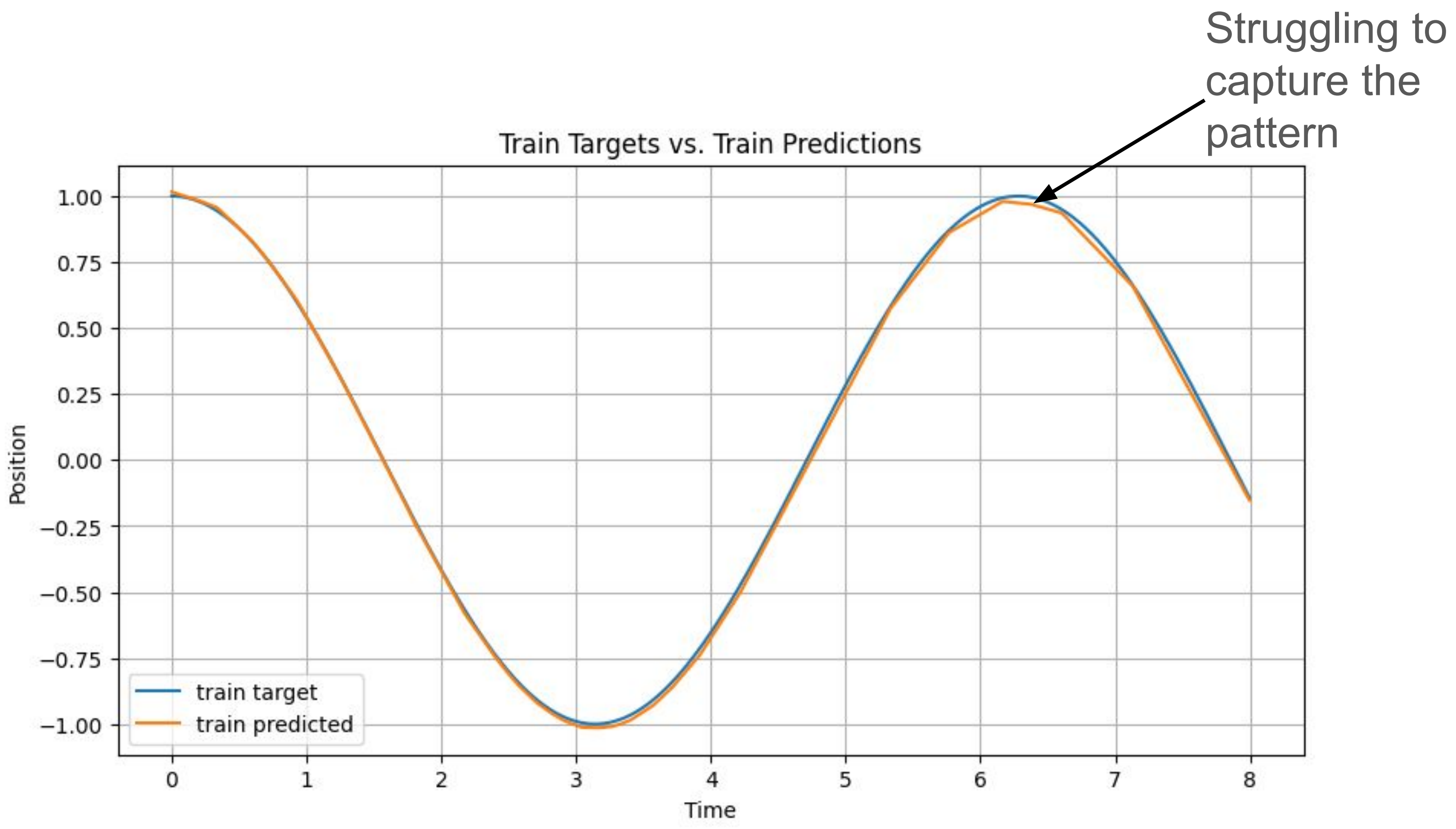
FNN

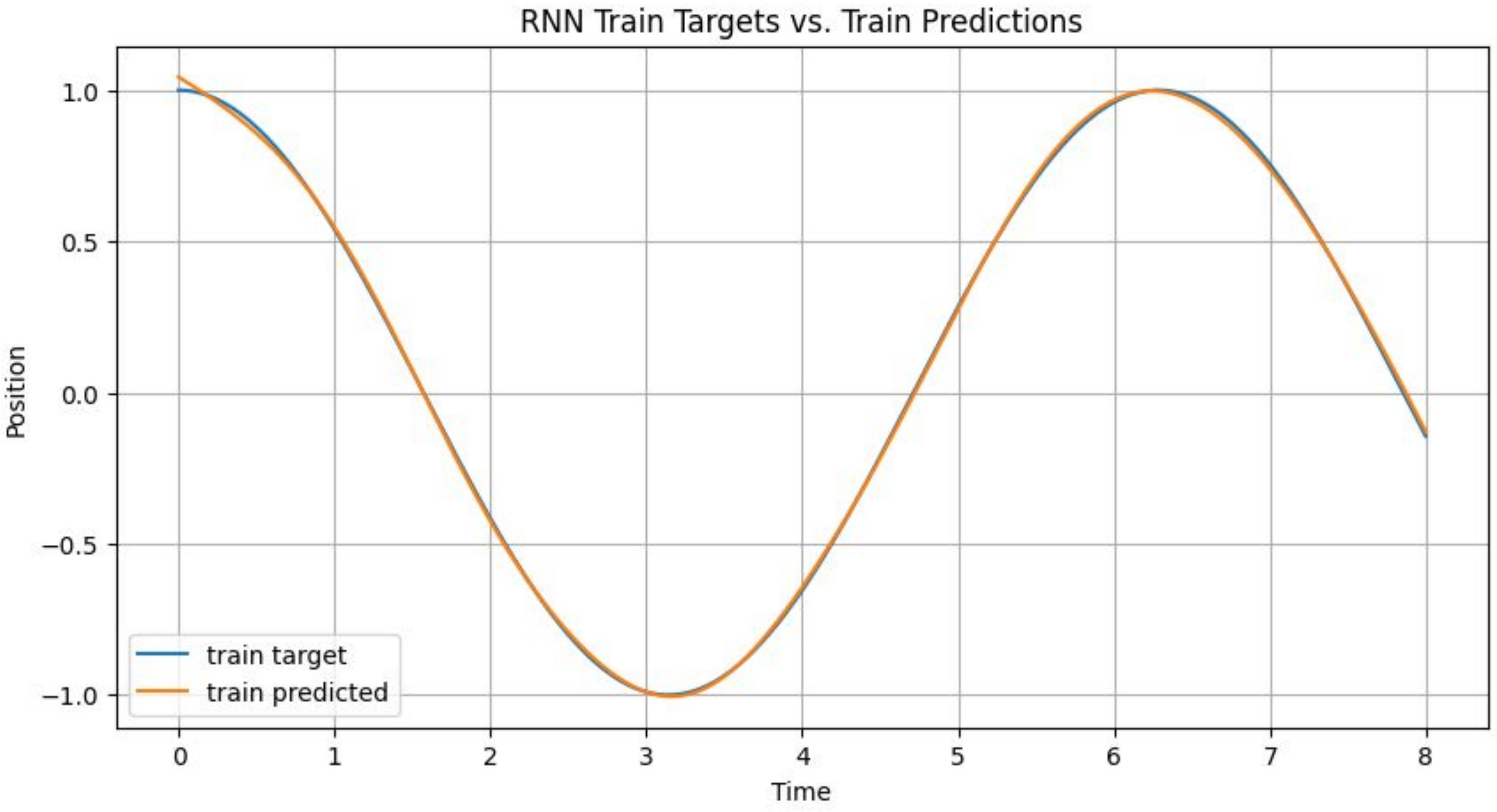


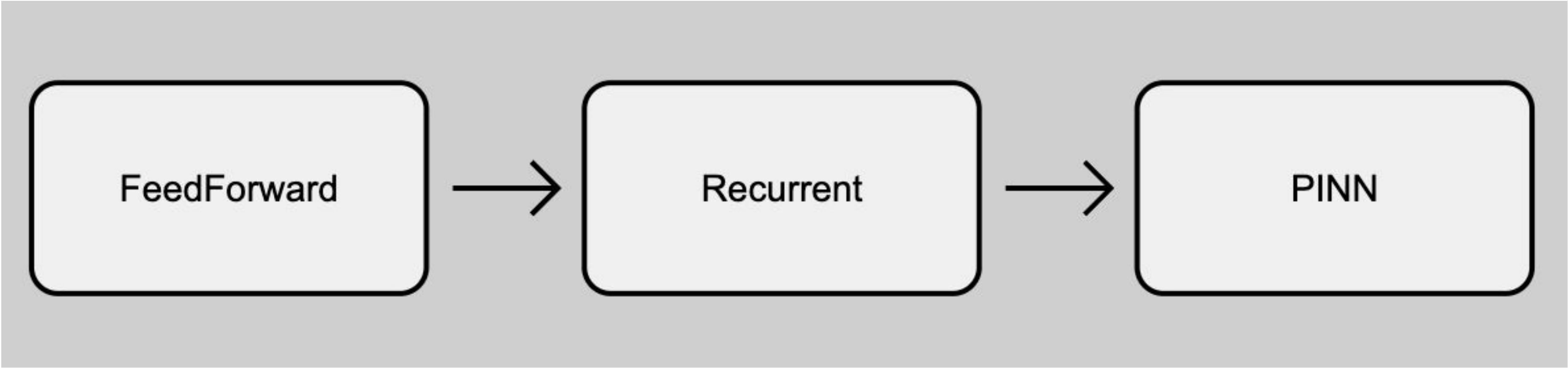


Loss:

$$\mathcal{L}_{\text{data}} = \frac{1}{N} \sum_{i=1}^{N} \left(x_{\text{NN}}(t_i) - x(t_i)\right)^2$$


Struggling to capture the pattern
Train Targets vs. Train Predictions
Position
Time
train target
train predicted
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
1
2
3
4
5
6
7
8


RNN Train Targets vs. Train Predictions
Position
Time
1.0
0.5
0.0
−0.5
−1.0
0
1
2
3
4
5
6
7
8
train target
train predicted

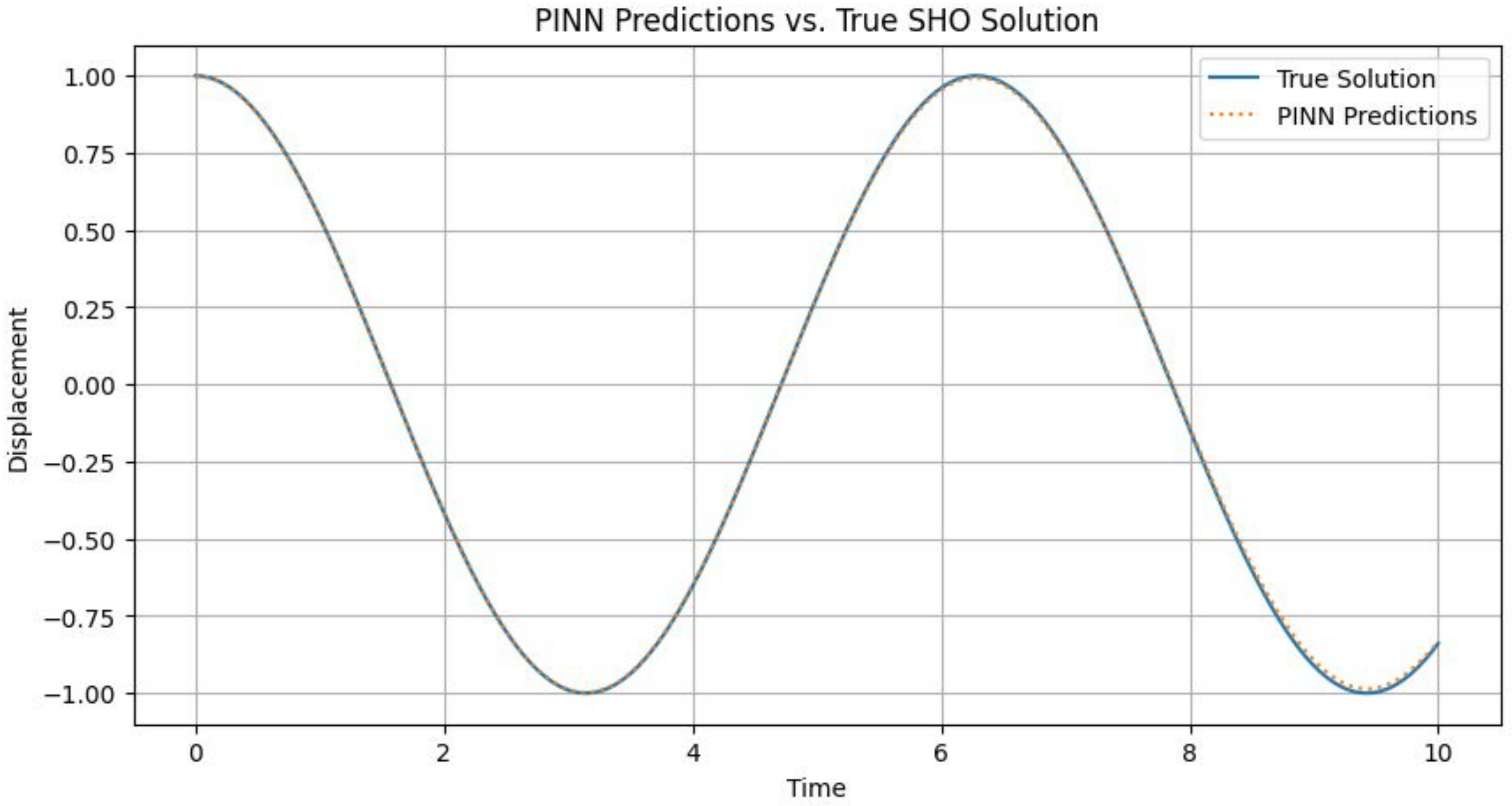

FeedForward
Recurrent
PINN

PINN Loss Function

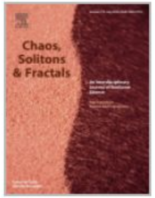
Data Loss + Physics Loss

$$\mathcal{L}_{\text{data}} = \frac{1}{N}\sum_{i=1}^{N}\left(x_{\text{PINN}}(t_i) - x(t_i)\right)^2$$

$$\mathcal{L}_{\text{physics}} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{d^2}{dt^2}x_{\text{PINN}}(t_i) + \omega^2 x_{\text{PINN}}(t_i)\right)^2$$


PINN Predictions vs. True SHO Solution
True Solution
PINN Predictions
Displacement
Time
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10

Chaos, Solitons & Fractals
Volume 172, July 2023, 113509



Data-driven multi-valley dark solitons of multi-component Manakov Model using Physics-Informed Neural Networks

Pros of PINN	Cons of PINN
Requires less data due to physics-based constraints.	Slightly higher computational cost due to solving differential equations.
Handles noisy and sparse data effectively.	Requires domain knowledge to encode the governing equations properly.
Provides high interpretability by adhering to physical laws.	May struggle if the governing equations are unknown or highly complex.
Reduces overfitting by enforcing physical constraints.	Implementation can be more challenging compared to traditional neural networks.
Capable of extrapolating beyond training data.	Training may take longer due to physics-informed loss computation.
Ideal for modeling dynamic systems governed by differential equations.	Sensitive to the quality of the equations used.
Ensures stable predictions by constraining solutions to physically valid ones.	Balancing between data loss and physics loss requires careful tuning.