

Neural Oscillators for Generalization of Physics-Informed Machine Learning

Taniya Kapoor^{1*}, Abhishek Chandra^{2*}, Daniel M. Tartakovsky³,
Hongrui Wang¹, Alfredo Nunez¹, Rolf Dollevoet¹

¹Department of Engineering Structures, Delft University of Technology, The Netherlands

²Department of Electrical Engineering, Eindhoven University of Technology, The Netherlands

³Department of Energy Science and Engineering, Stanford University, USA
t.kapoor@tudelft.nl

Abstract

A primary challenge of physics-informed machine learning (PIML) is its generalization beyond the training domain, especially when dealing with complex physical problems represented by partial differential equations (PDEs). This paper aims to enhance the generalization capabilities of PIML, facilitating practical, real-world applications where accurate predictions in unexplored regions are crucial. We leverage the inherent causality and temporal sequential characteristics of PDE solutions to fuse PIML models with recurrent neural architectures based on systems of ordinary differential equations, referred to as neural oscillators. Through effectively capturing long-time dependencies and mitigating the exploding and vanishing gradient problem, neural oscillators foster improved generalization in PIML tasks. Extensive experimentation involving time-dependent nonlinear PDEs and biharmonic beam equations demonstrates the efficacy of the proposed approach. Incorporating neural oscillators outperforms existing state-of-the-art methods on benchmark problems across various metrics. Consequently, the proposed method improves the generalization capabilities of PIML, providing accurate solutions for extrapolation and prediction beyond the training data.

Introduction

In machine learning and artificial intelligence, *generalization* refers to the ability of a model to perform on previously unseen data beyond its training domain. This entails prediction of outcomes for a sample \mathbf{x} that lies outside the convex hull of the training set $X = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, where N is the number of training samples (Balestrieri, Pesenti, and LeCun 2021). Current deep-learning models exhibit robust generalization on tasks like image (Su et al. 2023), and speech recognition (Chen et al. 2023), among others (Zhou et al. 2022). In physical sciences, state-of-the-art deep-learning models, also known as data-driven approaches, learn patterns and correlations from training data but lack intrinsic comprehension of the underlying governing laws of the problem (Liu et al. 2019; Alber et al. 2019). Despite their effective approximation of complex functions and relationships, these data-driven methods face challenges in generalizing to scenarios significantly different from the training

distribution, resulting in a physical-agnostic methodology (Gu et al. 2022).

Limitations of data-driven methods, characterized by their inability to adhere to physical laws and their agnosticism towards underlying physics, underscore the need for deep learning models capable of effectively capturing fundamental physical phenomena, such as their structure and symmetry (Lee, Trask, and Stinis 2021). Adopting such learning approaches promises to enhance the generalization capabilities of the model significantly. Consequently, a growing interest has been in embedding physics principles into machine learning to develop physics-aware models such as physics-informed neural networks (PINNs) (Raissi, Perdikaris, and Karniadakis 2019). PINNs consider mathematical models of the underlying physical process, represented as partial differential equations (PDEs), and integrate them into the loss function during training.

Despite their popularity, experimental evidence suggests that PINNs might fail to generalize. Minimizing the PDE residual in PINN does not straightforwardly control the generalization error (Mishra and Molinaro 2023, 2022). Although PINNs and their subsequent enhancement aim to incorporate soft or hard physical constraints for robustness, they often struggle to achieve strong generalization (Kim et al. 2021; Daw et al. 2022a; Fesser, Qiu, and D’Amico-Wong 2023). Hence, simply embedding physical equations into the loss function need not necessarily guarantee genuine physics awareness or robustness beyond the training domain. Ideally, a physics-informed model must reproduce known physics in the training domain and exhibit predictive capabilities for new scenarios while respecting conservation laws and effectively handling variations and uncertainties in real-world applications. Attaining this level of physics awareness remains a crucial challenge in developing dependable and powerful physics-informed machine learning methods (Fuks and Tchelepi 2020; Shin, Darbon, and Karniadakis 2020).

One way to enhance the extrapolation power of PINNs is to dynamically manipulate the gradients of the loss terms, building upon a gradient-based optimizer (Kim et al. 2021). This method shares similarities with gradient-based techniques employed in domain generalization tasks (Wang et al. 2022). However, one drawback of such methods is the need for training until a specific user-defined tolerance in the

*These authors contributed equally.

Copyright © 2024, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

loss is achieved, resulting in convergence issues and increased computational costs. We adopt a different strategy to tackle the generalization challenge by leveraging the inherent *causality* present in PDE solutions (Wang, Sankaran, and Perdikaris 2022). Leveraging causality enables us to enhance generalizability by learning the underlying dynamics that preserve the structure and symmetry of the underlying problem.

A recurrent neural network (RNN) might be capable of learning the dynamics owing to its remarkable success in various sequential tasks. Gated architectures, like long short-term memory (LSTM) (Hochreiter and Schmidhuber 1997) and gated recurrent unit (GRU) (Cho et al. 2014), have been mooted to address the exploding and vanishing gradient problem (EVGP) in vanilla RNNs (Pascanu, Mikolov, and Bengio 2013). However, EVGP can remain a concern as presented by Li et al. RNNs with orthogonality constraints on recurrent weight matrices are used to tackle EVGP (Henaff, Szlam, and LeCun 2016; Arjovsky, Shah, and Bengio 2016; Wisdom et al. 2016; Kerg et al. 2019). While this strategy alleviates EVGP, it may reduce expressivity and hinder performance in practical tasks (Kerg et al. 2019). We posit that *neural oscillators* (Lanthaler, Rusch, and Mishra 2023) offers a practical means to achieve high expressibility and mitigate EVGP. Neural oscillators use ordinary differential equations (ODEs) to update the hidden states of the recurrent unit, enabling efficient dynamic learning.

This paper introduces a new approach to address the generalization challenge. It employs a physics-informed neural architecture that learns the underlying dynamics in the training domain, followed by a neural oscillator to exploit the causality and learn temporal dependencies between the solutions at subsequent time levels. This extension of a physics-informed architecture helps increase the accuracy of a generalization task since neural oscillators carry a hidden state that retains information from previous time steps, enabling the model to capture and leverage temporal dependencies in the data.

We consider two different neural oscillators: *coupled oscillator recurrent neural network* (CoRNN) (Rusch and Mishra 2020) and *long expressive memory* (LEM) (Rusch et al. 2021). Both methods use a coupled system of ODEs to update the hidden states. We ascertain the relative performance of these two oscillators on three benchmark nonlinear problems: viscous Burgers equation, Allen–Cahn equation, and Schrödinger equation. Additionally, we evaluate the performance of our method in generalizing a solution for the Euler–Bernoulli beam equation. To showcase the performance of the proposed framework for higher-dimensional PDEs, we performed an experiment on 2D Kovasznay flow as presented in supplementary material SM5C provided at (Kapoor et al. 2023b).

The remainder of the manuscript is structured as follows. The “Related Work” section provides an overview of pertinent literature and recent studies related to the current work. In the “Method” section, our approach for enhancing the generalization of physics-informed machine learning through integration with a neural oscillator is explained in detail. Our method is validated through a series of numer-

ical experiments in the “Numerical Experiments” section. Finally, key findings and implications of this study are collated in the “Conclusions” section.

Related Work

PIML Our research aims to advance physics-informed models, a subset of machine learning techniques that address physical problems formulated as PDEs. PIML encompasses a range of methodologies, including physics-informed (Karniadakis et al. 2021), physics-based (Cuomo et al. 2022), physics-guided (Daw et al. 2022b), and theory-guided (Cuomo et al. 2022) approaches. The review papers (Karniadakis et al. 2021; Cuomo et al. 2022) provide a comprehensive overview of progress in PIML. Recently, PIML has demonstrated considerable utility in scientific and engineering disciplines, encompassing fluid dynamics (Raissi, Yazdani, and Karniadakis 2020) and materials science (Zhang et al. 2022), among others. Our primary focus is to improve PIML variants that integrate governing equations into the loss function during training to foster generalization, which involves advancing PINNs and their variations, such as causal PINNs (Wang, Sankaran, and Perdikaris 2022), and self-adaptive PINNs (McClenny and Braga-Neto 2023).

Domain generalization Domain generalization focuses on training models to effectively handle unseen domains with diverse data distributions, even when trained on data from related but distinct domains (Zhou et al. 2022; Wang et al. 2022). In contrast, domain adaptation involves transferring knowledge from a labeled source domain to an unlabeled or partially labeled target domain, assuming access to some labeled data in the target domain (Shen et al. 2018). Our research shares the core principles with these fields but differs in that we learn *exclusively* from a single training set without using multiple domains, as in domain generalization, or having access to any target domain data, as in domain adaptation. Moreover, we do not employ any transfer learning techniques. Our task is to train *solely* on the training set and *directly deploy* the trained model on the test region.

Generalization in PIML Despite limited research on the generalization of physics-informed models, some studies have specifically focused on generalizing PINNs. One noteworthy approach is the dynamic pulling method (DPM) (Kim et al. 2021), which utilizes a gradient-based technique to extend the solution of nonlinear benchmark problems beyond the trained convex hull X , focusing on generalizing solutions in the temporal domain. Other investigations have centered on generalizing the parameter space for parametric PDEs, employing techniques like curriculum learning, sequence-to-sequence learning (Krishnapriyan et al. 2021) and incremental learning (Dekhovitch et al. 2023). However, these approaches involve training and testing *within the convex hull* of the parameter space, which differs from the focus and approach to our work.

Neural oscillators Oscillator networks are ubiquitous in natural and engineering systems, exemplified by pendulums (classical mechanics) and heartbeats (biology). A growing trend involves building RNN architectures based on ODEs

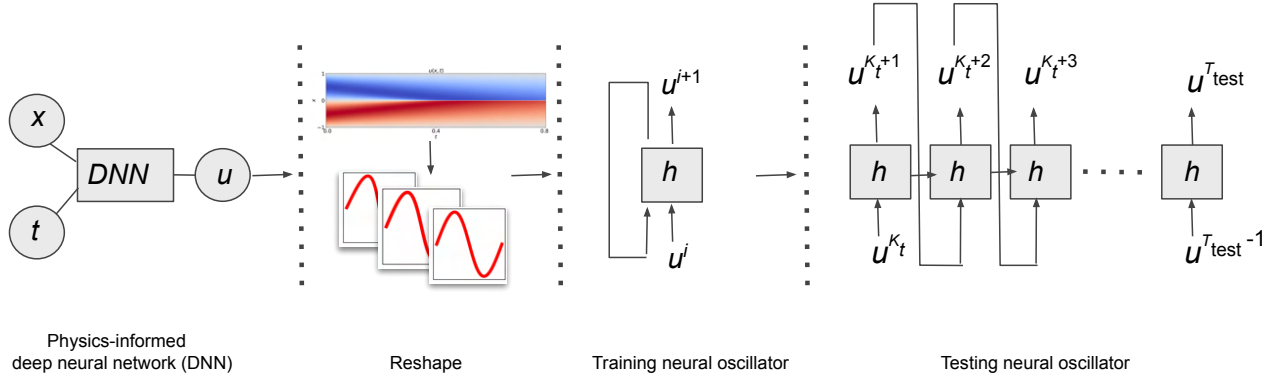


Figure 1: The proposed framework in which a physics-informed architecture (e.g., PINN or its variants) learns a solution in the convex hull X_1 . After reshaping, these solutions are represented sequentially and processed by one of the neural oscillators. The neural oscillator is finally tested in the convex testing hull X_2 , where the output of the last prediction step is the input for the next prediction step. Here, $1 \leq i \leq k_t$, $i \in \mathbb{Z}$, and $h = [y, z]$. The dotted lines separate different stages of training and testing the framework.

and dynamical systems (Chen et al. 2018; Rubanova, Chen, and Duvenaud 2019; Chang et al. 2019; Rusch and Mishra 2021). Recent research has abstracted the fundamental nature of functional brain circuits as networks of oscillators, constructing RNNs using simpler mechanistic systems represented by ODEs while disregarding complex biological neural function details. Driven by the *long-term memory* of these oscillators and inspired by the *universal approximation property* (Lanthaler, Rusch, and Mishra 2023), our goal is to integrate them with physics-informed models to enhance generalization.

Method

The proposed framework comprises a feedforward neural network informed by physics (such as PINN, causal PINN, self-adaptive PINN, or any other physics-guided architecture), followed by a neural oscillator. For example, we combine PINN with the coupled oscillatory recurrent neural network (CoRNN) or the long expressive memory (LEM) model. The output of the PINN serves as input to the oscillator. The PINN learns a solution within a convex training hull $X_1 = D \times T$, where $D \in \mathbb{R}^d$ is the d -dimensional spatial domain and $T \in \mathbb{R}$ is the temporal domain of the PDE. In our experiments, $d = 1$.

The neural oscillator processes the PINN’s output as sequential data and predicts solutions within a different convex testing hull X_2 . The hulls are distinct, X_1 and X_2 , and $X_2 \not\subseteq X_1$. For example, $X_2 = D \times T'$, where D is the same spatial domain but $T' \in \mathbb{R}$ is the extrapolated temporal domain with $\inf(T') \geq \sup(T)$, which implies that testing is performed on time $t' \in T' \geq t \in T$.

The PINN maps the input space X_1 onto the solution space \mathcal{U} , such that a solution of the PDE $u \in \mathcal{U}$. This mapping enables learning the evolution of u from a given initial condition. The abstract formulation of an operator \mathcal{N} incor-

porating the PDE and initial and boundary conditions is

$$\mathcal{N}(u) = f, \quad (1)$$

where f is the source term. The loss function of an abstract PINN is formed by minimising the residuals of (1) along with the available data on boundaries and at the initial time.

Following the PINN training on X_1 , its testing is conducted on k_t uniform time steps in T and k_x uniform locations in D making a total of $k_t \cdot k_x$ testing points within X_1 . The solution obtained from the PINN is reshaped to be further fed into the neural oscillator (Fig. 1).

Conventional feed-forward neural networks lack explicit mechanisms to learn dependencies among outputs, presenting a fundamental *challenge in handling temporal relationships*. To mitigate this challenge, recurrent neural architectures preserve a hidden state to retain information from previous time steps, thereby improving sequence learning. We employ neural oscillators to treat the PINN’s outputs as a sequence. The motivation arises from feed-forward neural networks, where all outputs are independent, whereas sequence learning requires capturing temporal dependencies. Neural oscillators capture these dependencies through feedback loops and hidden states, enabling information propagation and temporal dependency capture.

While training an oscillator, its hidden states are updated using the current input and the previous hidden states, akin to vanilla RNNs. The fundamental distinction between vanilla or gated RNNs and neural oscillators lies in the hidden state update methodology. In neural oscillators, these updates are based on systems of ODEs, in contrast to algebraic equations used in typical RNNs. When employing CoRNN, the hidden states are updated through the second-order ODE

$$\mathbf{y}'' = \sigma(\mathbf{W}\mathbf{y} + \mathbf{W}\mathbf{y}' + \mathbf{V}\mathbf{u} + \mathbf{b}) - \gamma\mathbf{y} - \epsilon\mathbf{y}'. \quad (2)$$

Here, $\mathbf{y} = \mathbf{y}(t) \in \mathbb{R}^m$ is the hidden state of the RNN with weight matrices $\mathbf{W}, \mathbf{W} \in \mathbb{R}^{m \times m}$ and $\mathbf{V} \in \mathbb{R}^{m \times k_x}$; t corresponds to the time levels at which the PINN’s testing has

been performed; $\mathbf{u} = \mathbf{u}(t) \in \mathbb{R}^{k_x}$ is the PINN solution; $\mathbf{b} \in \mathbb{R}^m$ is the bias vector; and $\gamma, \epsilon > 0$ are the oscillatory parameters. We set the activation function $\sigma : \mathbb{R} \mapsto \mathbb{R}$ to $\sigma(u) = \tanh(u)$. Introducing $\mathbf{z} = \mathbf{y}'(t) \in \mathbb{R}^m$, we rewrite (2) as the first-order system

$$\mathbf{y}' = \mathbf{z}, \quad \mathbf{z}' = \sigma(\mathbf{W}_y \mathbf{y} + \mathbf{W}_z \mathbf{z} + \mathbf{V}_u \mathbf{u} + \mathbf{b}) - \gamma \mathbf{y} - \epsilon \mathbf{z}. \quad (3)$$

We use an explicit scheme with a time step $0 < \Delta t < 1$ to discretize these ODEs,

$$\begin{aligned} \mathbf{y}_n &= \mathbf{y}_{n-1} + \Delta t \mathbf{z}_n, \\ \mathbf{z}_n &= \mathbf{z}_{n-1} + \Delta t \sigma(\mathbf{W}_y \mathbf{y}_{n-1} + \mathbf{W}_z \mathbf{z}_{n-1} + \mathbf{V}_u \mathbf{u}_n + \mathbf{b}) \\ &\quad - \Delta t \gamma \mathbf{y}_{n-1} - \Delta t \epsilon \mathbf{z}_{n-1}. \end{aligned} \quad (4)$$

Similarly, LEM updates the hidden states by solving the ODEs

$$\begin{aligned} \mathbf{y}' &= \hat{\sigma}(\mathbf{W}_2 \mathbf{y} + \mathbf{V}_2 \mathbf{u} + \mathbf{b}_2) \odot [\sigma(\mathbf{W}_y \mathbf{z} + \mathbf{V}_y \mathbf{u} + \mathbf{b}_y) - \mathbf{y}] \\ \mathbf{z}' &= \hat{\sigma}(\mathbf{W}_1 \mathbf{y} + \mathbf{V}_1 \mathbf{u} + \mathbf{b}_1) \odot [\sigma(\mathbf{W}_z \mathbf{y} + \mathbf{V}_z \mathbf{u} + \mathbf{b}_z) - \mathbf{z}] \end{aligned} \quad (5)$$

In addition to previously defined quantities, $\mathbf{W}_{1,2}, \mathbf{W}_{y,z} \in \mathbb{R}^{m \times m}$ and $\mathbf{V}_{1,2}, \mathbf{V}_{y,z} \in \mathbb{R}^{m \times k_x}$ are the weight matrices; $\mathbf{b}_{1,2}$ and $\mathbf{b}_{y,z} \in \mathbb{R}^m$ are the bias vectors; $\hat{\sigma}$ is the sigmoid activation function; and \odot refers to the componentwise product of vectors. A discretization of (5) similar to CoRNN yields

$$\begin{aligned} \Delta \mathbf{t}_n &= \Delta t \hat{\sigma}(\mathbf{W}_1 \mathbf{y}_{n-1} + \mathbf{V}_1 \mathbf{u}_n + \mathbf{b}_1) \\ \overline{\Delta \mathbf{t}_n} &= \Delta t \hat{\sigma}(\mathbf{W}_2 \mathbf{y}_{n-1} + \mathbf{V}_2 \mathbf{u}_n + \mathbf{b}_2) \\ \mathbf{z}_n &= (1 - \Delta \mathbf{t}_n) \odot \mathbf{z}_{n-1} \\ &\quad + \Delta \mathbf{t}_n \odot \sigma(\mathbf{W}_z \mathbf{y}_{n-1} + \mathbf{V}_z \mathbf{u}_n + \mathbf{b}_z) \\ \mathbf{y}_n &= (1 - \overline{\Delta \mathbf{t}_n}) \odot \mathbf{y}_{n-1} \\ &\quad + \overline{\Delta \mathbf{t}_n} \odot \sigma(\mathbf{W}_y \mathbf{z}_n + \mathbf{V}_y \mathbf{u}_n + \mathbf{b}_y). \end{aligned} \quad (6)$$

Both CoRNN and LEM are augmented with a linear output state $\omega_n \in \mathbb{R}^{k_x}$ with $\omega_n = \mathbf{Q} \mathbf{y}_n$ and $\mathbf{Q} \in \mathbb{R}^{k_x \times m}$.

We train the PINN and the neural oscillator separately to leverage the *resolution-invariance* property of physics-informed learning during training. While neural oscillators require evenly spaced data, a PINN can be trained discretization-invariantly, allowing flexibility in handling multi-resolution data, such as using different sampling techniques (Daw et al. 2022a). The PINN is trained until a predefined epoch or until its validation error stabilizes in consecutive epochs and is then employed in inference to generate training data for the oscillator. Subsequently, the oscillator learns a mapping between the PINN outputs from one-time level to the next, forming a sequential relationship.

Numerical Experiments

We validate the proposed framework on three time-dependent nonlinear PDEs and a fourth-order biharmonic beam equation. The software and hardware environments used to perform the experiments are as follows: UBUNTU 20.04.6 LTS, PYTHON 3.9.7, NUMPY 1.20.3, SCIPY 1.7.1, MATPLOTLIB 3.4.3, TENSORFLOW-GPU 2.9.1, PYTORCH 1.12.1, CUDA 11.7, and NVIDIA Driver 515.105.01, i7 CPU, and NVIDIA GEFORCE RTX 3080.

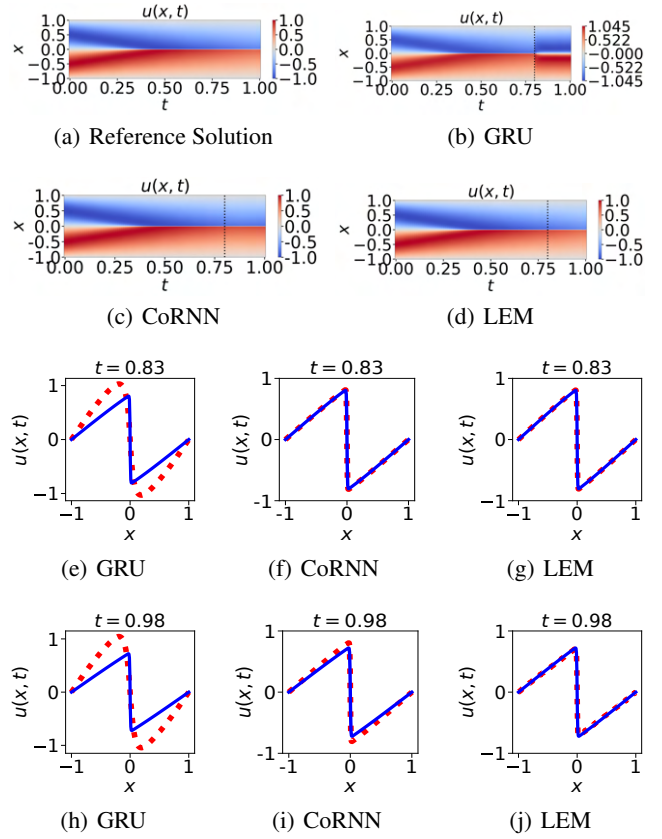


Figure 2: Top two rows: the complete reference solution and predictions for viscous Burgers equation. The black vertical line delineates the region before which the PINN has been trained. The region after the black vertical line represents the generalization domain. The meaning of the vertical line remains the same in the following figures. Bottom: the solution snapshots at $t = \{0.83, 0.98\}$ obtained in the generalization region, where blue represents the reference solution, and red refers to the recurrent method. The colors are used consistently for the following figures.

PDEs The four equations—viscous Burgers equation, Allen-Cahn (AC) equation, nonlinear Schrödinger equation (NLS) and Euler-Bernoulli beam equation—along with their boundary and initial conditions are provided in the supplementary material **SM§B**. For training/testing, we divide the entire time domain into two segments: $T := [0, T_{\text{train}}]$ and $T' := (T_{\text{train}}, T_{\text{test}}]$, where $T_{\text{test}} > T_{\text{train}} > 0$. Our task is to predict the PDE solution in the convex testing hull $X_2 = D \times T'$ after the model has been trained on the convex training hull $X_1 = D \times T$. For all the problems, $T_{\text{train}} = 0.8T_{\text{test}}$, dividing the training and test sets in the ratio 4 : 1, following the work of DPM (Kim et al. 2021) to maintain uniformity. The domain for each PDE, i.e., D, T and T' , is defined in **SM§B**.

Baselines Our objective is to make predictions beyond X_1 , i.e., on X_2 , and to assess how well the trained mod-

PDE	L2-norm (\downarrow)			Explained variance score (\uparrow)			Max error (\downarrow)			Mean absolute error (\downarrow)		
	DPM	CoRNN	LEM	DPM	CoRNN	LEM	DPM	CoRNN	LEM	DPM	CoRNN	LEM
Vis. Burgers	0.083	0.0044	0.0001	0.621	0.9955	0.9998	1.534	0.1035	0.0246	0.277	0.0222	0.0035
Allen-Cahn	0.182	0.0051	0.0049	0.967	0.9954	0.9956	0.836	0.3201	0.1376	0.094	0.0356	0.0348
Schrödinger	0.141	0.0426	0.0034	-3.257	0.9250	0.9944	3.829	0.6596	0.0948	0.868	0.9250	0.0281

Table 1: The generalization accuracy in terms of the relative errors in the L2-norm, the explained variance error, the max error, and the mean absolute error for nonlinear benchmark PDEs. Higher (or lower) values are preferred, corresponding to \uparrow (or \downarrow).

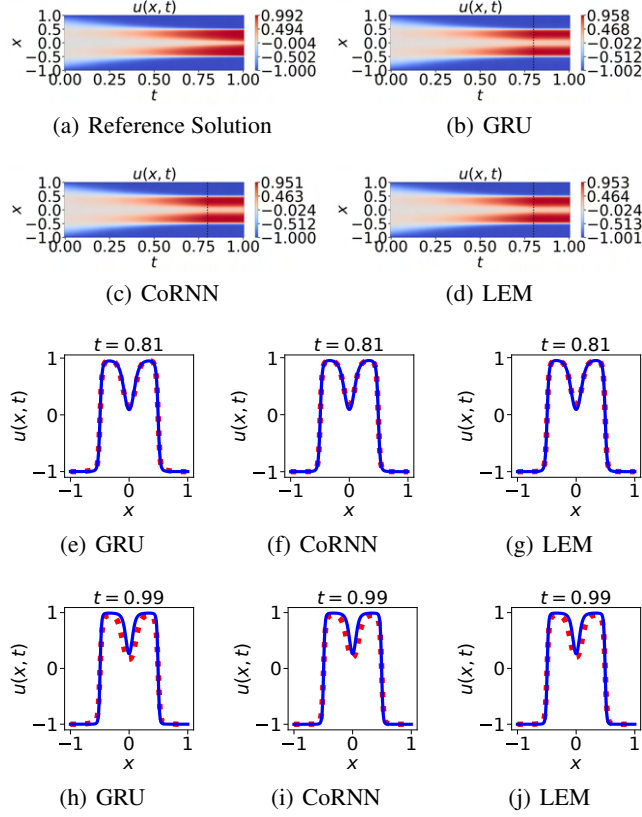


Figure 3: Top two rows: the complete reference solution and predictions for the Allen-Cahn equation. Bottom: the solution snapshots at $t = \{0.81, 0.99\}$ obtained in the generalization region.

els generalize. We compare the performance of PINNs with CoRNN or LEM on this task. We also compare our approach to the state-of-the-art DPM (Kim et al. 2021). A comparative analysis is also carried out when traditional recurrent networks, RNN, LSTM, and GRU, are augmented with the physics-informed model instead of the oscillatory networks. This analysis provides insight into how well the oscillatory methods perform relative to traditional recurrent networks and gradient techniques when confronted with generalization tasks.

Hyperparameters To predict a solution to Burgers equation in X_1 using PINNs, 1600 training points are used, comprising 1000 residual points and 600 points for bound-

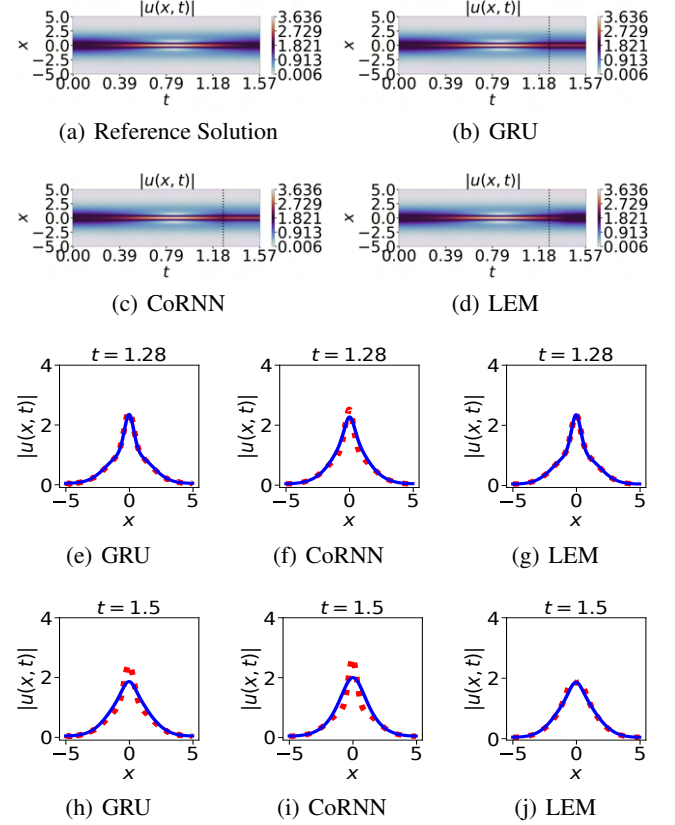


Figure 4: Top two rows: the complete reference solution and predictions for the Schrödinger equation. Bottom: the solution snapshots at $t = \{1.28, 1.5\}$ obtained in the generalization region.

ary and initial time. The feedforward neural network has two inputs, space $x \in D$ and time $t \in T$. Four hidden layers, each containing 20 neurons, and hyperbolic tangent (tanh) activation function are used to predict the approximation of the solution $u \in \mathcal{U}$. Optimization is performed using the LBFGS algorithm for 3500 epochs. For the Euler-Bernoulli beam equation, 16000 training points are distributed as 10000 residual points and 6000 points designated for both initial and boundaries. The hyperparameters are kept the same as in the viscous Burgers equation. Allen-Cahn and Schrödinger equations are simulated using the software DeepXDE (Lu et al. 2021) with the default hyperparameters described therein.

The input and output size of the recurrent networks is

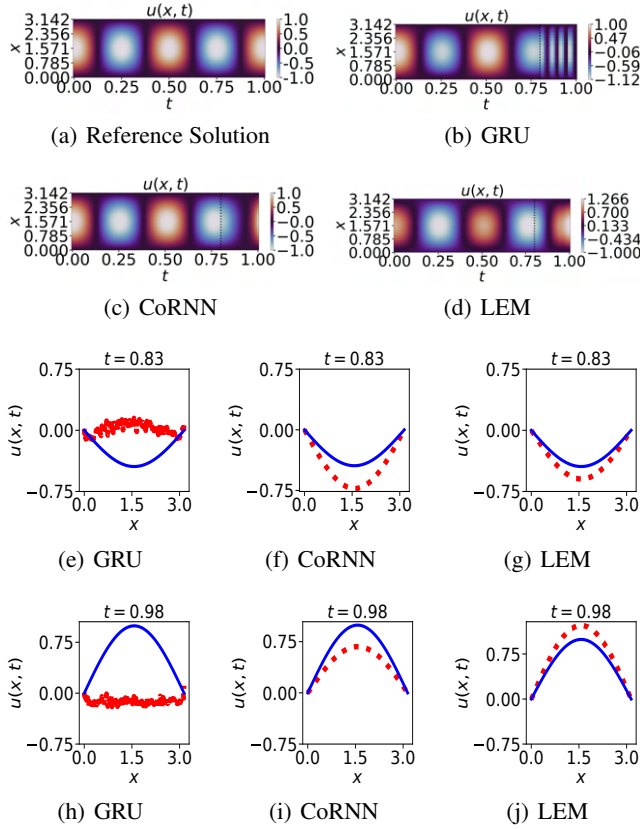


Figure 5: Top two rows: the complete reference solution and predictions for the Euler-Bernoulli beam equation. Bottom: the solution snapshots at $t = \{0.83, 0.98\}$ obtained in the generalization region.

taken to be k_x , with a single hidden layer of size 32. The sequence length is chosen to be k_t . The exact values of k_x and k_t are defined in the “Train and test criteria” subsection for each equation. ADAM optimizer is used to train the recurrent networks. The learning rates for LEM, CoRNN, GRU, LSTM, and RNN are 0.001, 0.001, 0.01, 0.01, and 0.01, respectively, across all equations. For Schrödinger equation, a learning rate of 0.01 is used to train the LEM. In the case of CoRNN, two additional hyperparameters, γ and ϵ , are set to 1.0 and 0.01, respectively. The number of epochs executed for Burgers and Allen-Cahn equations is 20,000, while for Schrödinger equation, it is 30,000. Lastly, 200,000 epochs are performed for the Euler-Bernoulli beam equation.

Evaluation metrics For the first three experiments, the errors are reported relative to the numerical solutions of the corresponding PDEs. The reference for the Euler-Bernoulli beam equation is an analytical solution described in SM§B. As the criteria for assessment, we employ standard evaluation metrics: the relative errors in the L2-norm, the explained variance score, the maximum error, and the mean absolute error, defined in SM§D. Each of these metrics provides distinct insights into the performance. Furthermore, we present visual snapshots of both the reference and approximate so-

lutions at specific time instances. Additional snapshots and contour results are provided in SM§C.

Train and test criteria The trained PINN is tested on $k_t \cdot k_x$ points in X_1 . For the Burgers equation and the Euler-Bernoulli beam equations, we set $k_x = 256$ and $k_t = 80$. For the Allen-Cahn equation, $k_x = 201$ and $k_t = 80$. For the Schrödinger equation, $k_x = 256$ and $k_t = 160$.

The PINN output provides input to train the neural oscillators, adhering to the specified hyperparameter configuration. After training the neural oscillator on X_1 , testing is extended to X_2 . This testing sequence commences at $\inf(T')$ as the initial input. The ensuing output is then utilized as the input for the subsequent sequence (Fig. 1). Such testing is crucial since, in practical scenarios, knowledge about the solution u in X_2 is absent. Thus, the solely available information for generalization is derived from the predicted solution within X_2 . This testing process is iterated until reaching $\sup(T')$. The domains X_1 , X_2 and T' for all the equations are provided in SM§B.

Experimental Results

Tables 1 and 2 collate the overall performance metrics for the oscillator-based methods (LEM, CoRNN) in comparison with DPM, RNN, LSTM and GRU. The results show that LEM exhibits significantly superior performance across all the benchmark problems.

Viscous Burgers equation Figure 2 provides a visual comparison between the reference solution (Fig. 2(a)) and its counterparts generated with GRU, CoRNN and LEM (Figs. 2(b)–2(d), respectively). GRU struggles to accurately capture the solution of Burgers equation, leading to the loss in prediction accuracy as time t increases. Our methods based on CoRNN and LEM exhibit notably improved predictive accuracy, even when t approaches the end of the time domain. Figures 2(e)–2(j) provide further insights into the solution at time instances $t = 0.83, 0.98$. They reveal that LEM outperforms the alternative methods across the entire space-time domain. The performance of CoRNN is comparable to that of LEM, producing reasonably accurate predictions. These findings underscore the significance of neural oscillators in precise generalization. Additional experiments on *sensitivity analysis* of oscillator parameter (Δt) along with an *ablation study* on CoRNN parameters ϵ and γ is presented in SM§C. Additionally, the generalization in parametric space (Kapoor et al. 2023a) is also presented in SM§C.

Allen-Cahn equation In Figure 3, the reference solution of the Allen-Cahn equation is compared to its counterparts generated with GRU, CoRNN and LEM. Our oscillator-based methods (CoRNN and LEM) yield the most precise approximations in the generalization domain (Figs. 3(a)–3(d)). The LEM-based solution exhibits a nearly symmetric behavior with respect to $x = 0$, demonstrating its ability to preserve the symmetry and structure of the solution. At $t = 0.81$, all three methods display a similar level of accuracy (Figs. 3(e)–3(g)). However, as time advances, e.g., at $t = 0.99$, the performance of LEM surpasses that of

PDE	L2-norm (\downarrow)				Explained variance score (\uparrow)				Max error (\downarrow)				Mean absolute error (\downarrow)			
	RNN	LSTM	GRU	LEM	RNN	LSTM	GRU	LEM	RNN	LSTM	GRU	LEM	RNN	LSTM	GRU	LEM
Vis. Burgers	0.4154	0.4635	0.3768	0.0001	0.584	0.536	0.623	0.999	0.594	1.040	0.544	0.024	0.266	0.285	0.253	0.003
Allen–Cahn	0.0058	0.0570	0.0093	0.0049	0.995	0.946	0.991	0.995	0.145	0.399	0.176	0.137	0.040	0.194	0.050	0.034
Schrödinger	0.3170	0.5022	0.0218	0.0034	0.440	0.272	0.961	0.994	1.695	0.153	0.434	0.094	0.260	0.326	0.075	0.028
Euler–Bernoulli	4.6509	2.1198	2.9176	0.0593	-0.844	-0.258	-0.566	0.940	1.997	1.465	2.044	0.267	0.797	0.500	0.604	0.091

Table 2: The generalization accuracy in terms of the relative errors in the L2-norm, the explained variance error, the max error, and the mean absolute error for various PDEs. Higher (or lower) values are preferred, corresponding to \uparrow (or \downarrow).

the other techniques throughout the extrapolation domain (Figs. 3(h)–3(j)).

Schrödinger equation Figure 4 illustrates a comparison between the reference solution of Schrödinger equation and its counterparts generated with GRU, CoRNN and LEM. Rather than plotting the real and imaginary parts of this solution, Figs. 4(a)–4(d) exhibit its magnitude, $|u(x, t)|$; the solutions are visually indistinguishable. The three approximations are accurate at time $t = 1.28$ (Figs. 4(e)–4(g)), but the GRU- and CoRNN-based solutions at $t = 1.5$ have errors around $x = 0$ whereas the LEM-based solution retains its accuracy within that region (Figs. 4(f)–4(j)).

Euler-Bernoulli beam equation In Figure 5, we compare the analytical solution of the Euler-Bernoulli beam equation to approximate solutions obtained with GRU, CoRNN and LEM. The intricacy of this linear equation stems from the presence of fourth-order derivatives (Kapoor et al. 2023c; Cao, Goswami, and Karniadakis 2023), rendering it a compelling challenge for the proposed methodology. The visual comparison afforded by Figs. 5(a)–5(d) demonstrates the superiority of the LEM-based solution and the inferiority of the GRU-based one. At $t = 0.83$, all three approximations are qualitatively correct, with various degrees of accuracy (Figs. 5(e)–5(h)). At $t = 0.98$, the GRU-based solution is not only inaccurate but is also qualitatively incorrect, while the oscillator-based approximators correctly predict the system’s behavior (Figs. 5(h)–5(i)).

Conclusion

We introduced a method that combines neural oscillators with physics-informed neural networks to enhance performance in unexplored regions. This novel approach enables the model to learn the long-time dynamics of solutions to the governing partial differential equations. We demonstrated the effectiveness of our method on three benchmark nonlinear PDEs: viscous Burgers, Allen–Cahn, and Schrödinger equations, as well as the biharmonic Euler-Bernoulli beam equation. Our results showcase the improved generalization performance of the PIML augmented with neural oscillators, which outperforms state-of-the-art methods in various metrics. The codes to reproduce the presented results are provided at https://github.com/taniyakapoor/AAAI24_Generalization_PIML.

References

- Alber, M.; Buganza Tepole, A.; Cannon, W. R.; De, S.; Dura-Bernal, S.; Garikipati, K.; Karniadakis, G.; Lytton, W. W.; Perdikaris, P.; Petzold, L.; et al. 2019. Integrating machine learning and multiscale modeling—perspectives, challenges, and opportunities in the biological, biomedical, and behavioral sciences. *npj digital medicine*, 2(1): 115.
- Arjovsky, M.; Shah, A.; and Bengio, Y. 2016. Unitary evolution recurrent neural networks. In *International conference on machine learning*, 1120–1128. PMLR.
- Balestrieri, R.; Pesenti, J.; and LeCun, Y. 2021. Learning in high dimension always amounts to extrapolation. *arXiv preprint arXiv:2110.09485*.
- Cao, Q.; Goswami, S.; and Karniadakis, G. E. 2023. LNO: Laplace neural operator for solving differential equations. *arXiv preprint arXiv:2303.10528*.
- Chang, B.; Chen, M.; Haber, E.; and Chi, E. H. 2019. AntisymmetricRNN: A dynamical system view on recurrent neural networks. *arXiv preprint arXiv:1902.09689*.
- Chen, C.; Hu, Y.; Zhang, Q.; Zou, H.; Zhu, B.; and Chng, E. S. 2023. Leveraging modality-specific representations for audio-visual speech recognition via reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 12607–12615.
- Chen, R. T.; Rubanova, Y.; Bettencourt, J.; and Duvenaud, D. K. 2018. Neural ordinary differential equations. *Advances in neural information processing systems*, 31.
- Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.
- Cuomo, S.; Di Cola, V. S.; Giampaolo, F.; Rozza, G.; Raissi, M.; and Piccialli, F. 2022. Scientific machine learning through physics-informed neural networks: Where we are and what’s next. *Journal of Scientific Computing*, 92(3): 88.
- Daw, A.; Bu, J.; Wang, S.; Perdikaris, P.; and Karpatne, A. 2022a. Rethinking the importance of sampling in physics-informed neural networks. *arXiv preprint arXiv:2207.02338*.
- Daw, A.; Karpatne, A.; Watkins, W. D.; Read, J. S.; and Kumar, V. 2022b. Physics-guided neural networks (pgnn): An application in lake temperature modeling. In *Knowledge Guided Machine Learning*, 353–372. Chapman and Hall/CRC.

- Dekhovitch, A.; Sluiter, M. H.; Tax, D. M.; and Bessa, M. A. 2023. iPINNs: Incremental learning for Physics-informed neural networks. *arXiv preprint arXiv:2304.04854*.
- Fesser, L.; Qiu, R.; and D’Amico-Wong, L. 2023. Understanding and Mitigating Extrapolation Failures in Physics-Informed Neural Networks. *arXiv preprint arXiv:2306.09478*.
- Fuks, O.; and Tchalepi, H. A. 2020. Limitations of physics informed machine learning for nonlinear two-phase transport in porous media. *Journal of Machine Learning for Modeling and Computing*, 1(1).
- Gu, J.; Gao, Z.; Feng, C.; Zhu, H.; Chen, R.; Boning, D.; and Pan, D. 2022. NeurOLight: A Physics-Agnostic Neural Operator Enabling Parametric Photonic Device Simulation. *Advances in Neural Information Processing Systems*, 35: 14623–14636.
- Henaff, M.; Szlam, A.; and LeCun, Y. 2016. Recurrent orthogonal networks and long-memory tasks. In *International Conference on Machine Learning*, 2034–2042. PMLR.
- Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation*, 9(8): 1735–1780.
- Kapoor, T.; Chandra, A.; Tartakovsky, D.; Wang, H.; Núñez, A.; and Dollevoet, R. 2023a. Neural oscillators for generalizing parametric PDEs. In *The Symbiosis of Deep Learning and Differential Equations III*.
- Kapoor, T.; Chandra, A.; Tartakovsky, D. M.; Wang, H.; Nunez, A.; and Dollevoet, R. 2023b. Neural oscillators for generalization of physics-informed machine learning. *arXiv preprint arXiv:2308.08989*.
- Kapoor, T.; Wang, H.; Nunez, A.; and Dollevoet, R. 2023c. Physics-informed neural networks for solving forward and inverse problems in complex beam systems. *arXiv preprint arXiv:2303.01055*.
- Karniadakis, G. E.; Kevrekidis, I. G.; Lu, L.; Perdikaris, P.; Wang, S.; and Yang, L. 2021. Physics-informed machine learning. *Nature Reviews Physics*, 3(6): 422–440.
- Kerg, G.; Goyette, K.; Puelma Touzel, M.; Gidel, G.; Vorontsov, E.; Bengio, Y.; and Lajoie, G. 2019. Non-normal recurrent neural network (nnrnn): learning long time dependencies while improving expressivity with transient dynamics. *Advances in neural information processing systems*, 32.
- Kim, J.; Lee, K.; Lee, D.; Jhin, S. Y.; and Park, N. 2021. DPM: A novel training method for physics-informed neural networks in extrapolation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 8146–8154.
- Krishnapriyan, A.; Gholami, A.; Zhe, S.; Kirby, R.; and Mahoney, M. W. 2021. Characterizing possible failure modes in physics-informed neural networks. *Advances in Neural Information Processing Systems*, 34: 26548–26560.
- Lanthaler, S.; Rusch, T. K.; and Mishra, S. 2023. Neural Oscillators are Universal. *arXiv preprint arXiv:2305.08753*.
- Lee, K.; Trask, N.; and Stinis, P. 2021. Machine learning structure preserving brackets for forecasting irreversible processes. *Advances in Neural Information Processing Systems*, 34: 5696–5707.
- Li, S.; Li, W.; Cook, C.; Zhu, C.; and Gao, Y. 2018. Independently recurrent neural network (indrnn): Building a longer and deeper rnn. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5457–5466.
- Liu, H.; Zhang, T.; Anoop Krishnan, N.; Smedskjaer, M. M.; Ryan, J. V.; Gin, S.; and Bauchy, M. 2019. Predicting the dissolution kinetics of silicate glasses by topology-informed machine learning. *Npj Materials Degradation*, 3(1): 32.
- Lu, L.; Meng, X.; Mao, Z.; and Karniadakis, G. E. 2021. DeepXDE: A deep learning library for solving differential equations. *SIAM review*, 63(1): 208–228.
- McClenny, L. D.; and Braga-Neto, U. M. 2023. Self-adaptive physics-informed neural networks. *Journal of Computational Physics*, 474: 111722.
- Mishra, S.; and Molinaro, R. 2022. Estimates on the generalization error of physics-informed neural networks for approximating a class of inverse problems for PDEs. *IMA Journal of Numerical Analysis*, 42(2): 981–1022.
- Mishra, S.; and Molinaro, R. 2023. Estimates on the generalization error of physics-informed neural networks for approximating PDEs. *IMA Journal of Numerical Analysis*, 43(1): 1–43.
- Pascanu, R.; Mikolov, T.; and Bengio, Y. 2013. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, 1310–1318. Pmlr.
- Raissi, M.; Perdikaris, P.; and Karniadakis, G. E. 2019. 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.
- Raissi, M.; Yazdani, A.; and Karniadakis, G. E. 2020. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481): 1026–1030.
- Rubanov, Y.; Chen, R. T.; and Duvenaud, D. K. 2019. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32.
- Rusch, T. K.; and Mishra, S. 2020. Coupled Oscillatory Recurrent Neural Network (coRNN): An accurate and (gradient) stable architecture for learning long time dependencies. *arXiv preprint arXiv:2010.00951*.
- Rusch, T. K.; and Mishra, S. 2021. Unicornn: A recurrent model for learning very long time dependencies. In *International Conference on Machine Learning*, 9168–9178. PMLR.
- Rusch, T. K.; Mishra, S.; Erichson, N. B.; and Mahoney, M. W. 2021. Long expressive memory for sequence modeling. *arXiv preprint arXiv:2110.04744*.
- Shen, J.; Qu, Y.; Zhang, W.; and Yu, Y. 2018. Wasserstein distance guided representation learning for domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.
- Shin, Y.; Darbon, J.; and Karniadakis, G. E. 2020. On the convergence and generalization of physics informed neural networks. *arXiv e-prints*, arXiv:2004.

- Su, Z.; Yao, K.; Yang, X.; Huang, K.; Wang, Q.; and Sun, J. 2023. Rethinking data augmentation for single-source domain generalization in medical image segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 2366–2374.
- Wang, J.; Lan, C.; Liu, C.; Ouyang, Y.; Qin, T.; Lu, W.; Chen, Y.; Zeng, W.; and Yu, P. 2022. Generalizing to unseen domains: A survey on domain generalization. *IEEE Transactions on Knowledge and Data Engineering*.
- Wang, S.; Sankaran, S.; and Perdikaris, P. 2022. Respecting causality is all you need for training physics-informed neural networks. *arXiv preprint arXiv:2203.07404*.
- Wisdom, S.; Powers, T.; Hershey, J.; Le Roux, J.; and Atlas, L. 2016. Full-capacity unitary recurrent neural networks. *Advances in neural information processing systems*, 29.
- Zhang, E.; Dao, M.; Karniadakis, G. E.; and Suresh, S. 2022. Analyses of internal structures and defects in materials using physics-informed neural networks. *Science advances*, 8(7): eabk0644.
- Zhou, K.; Liu, Z.; Qiao, Y.; Xiang, T.; and Loy, C. C. 2022. Domain generalization: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.