Transferable Physics-Informed Representations via Closed-Form Head Adaptation

Jian Cheng Wong*, Isaac Yin Chung Lai†, Pao-Hsiung Chiu*, Chin Chun Ooi*, Abhishek Gupta‡, Yew-Soon Ong*§
*Institute of High Performance Computing (IHPC), Agency for Science, Technology and Research (A*STAR), Singapore
†National University of Singapore (NUS), Singapore
‡Indian Institute of Technology Goa (IIT Goa), India
§Nanyang Technological University (NTU), Singapore

***Abstract*—Physics-informed neural networks (PINNs) have garnered significant interest for their potential in solving partial differential equations (PDEs) that govern a wide range of physical phenomena. By incorporating physical laws into the learning process, PINN models have demonstrated the ability to learn physical outcomes reasonably well. However, current PINN approaches struggle to predict or solve new PDEs effectively when there is a lack of training examples, indicating they do not generalize well to unseen problem instances. In this paper, we present a transferable learning approach for PINNs premised on a fast Pseudoinverse PINN framework (Pi-PINN). Pi-PINN learns a transferable physics-informed representation in a shared embedding space and enables rapid solving of both known and unknown PDE instances via closed-form head adaptation using a least-squares-optimal pseudoinverse under PDE constraints. We further investigate the synergies between data-driven multi-task learning loss and physics-informed loss, providing insights into the design of more performant PINNs. We demonstrate the effectiveness of Pi-PINN on various PDE problems, including Poisson's equation, Helmholtz equation, and Burgers' equation, achieving fast and accurate physics-informed solutions without requiring any data for unseen instances. Pi-PINN can produce predictions 100–1000 times faster than a typical PINN, while producing predictions with 10–100 times lower relative error than a typical data-driven model even with only two training samples. Overall, our findings highlight the potential of transferable representations with closed-form head adaptation to enhance the efficiency and generalization of PINNs across PDE families and scientific and engineering applications.**

***Index Terms*—physics-informed neural networks, transfer learning, multi-task learning, generalization**

I. Introduction

Physics-informed neural networks (PINNs) have attracted significant attention as a neural approach for solving partial differential equations (PDEs) that govern a wide range of physical phenomena across science and engineering [1], [2]. PINNs incorporate governing laws directly into the learning objective by treating PDE residuals, and associated boundary conditions (BCs) and initial conditions (ICs), as training constraints, resulting in a "physics-informed" loss [3]–[6].

Despite their promise, two practical limitations have repeatedly emerged. First, PINNs often train slowly and can exhibit poor optimization behavior compared to purely data-driven networks, largely due to the stiffness and complex loss landscapes introduced by physics-informed constraints [7]–[9]. Second, even after expensive training, standard PINNs generalize poorly to new PDE instances (e.g., new coefficients, source terms, BCs/ICs, or parameter regimes), limiting their usefulness when extrapolation or rapid redeployment is needed [10]. As a result, PINNs are still commonly used in a single-instance setting where the physics-informed loss is enforced directly for the target PDE, while extending a trained model to new instances typically requires substantial re-training and re-tuning.

In many real-world settings, however, one does not start from scratch: there may exist a small collection of related PDE instances with solutions (or partial labels) that can be leveraged to learn representations that transfer. This motivates a transfer-learning perspective for PINNs, with the objective to build PINN models that are more reusable and adaptable across PDE families and parameter regimes.

Hence, in this paper, we present Pi-PINN, a fast Pseudoinverse PINN framework that learns *transferable physics-informed representations* and enables *closed-form head adaptation* via a least-squares-optimal pseudoinverse under PDE constraints. The core idea is to decouple learning into (i) a shared embedding that captures transferable structure across related PDE instances and (ii) a task-specific output head that can be adapted efficiently through a closed-form linear solve, avoiding costly gradient-based re-optimization for each new instance. In addition, we study how combining data-driven multi-task learning losses with physics-informed residuals can produce more performant and transferable PINN models.

The key contribution of our work are: (1) we introduce Pi-PINN, a pseudoinverse-based physics-informed learning framework that supports closed-form, least-squares-optimal head adaptation under PDE constraints, substantially reducing the computational cost of adapting PINNs to new PDE instances (see schematic in Figure 1). (2) We propose a representation-learning formulation for PINNs that learns transferable deep embeddings from related PDE instances, improving generalization across PDE families and parameter regimes. (3) We analyze the synergy between data-driven

*Corresponding author: wongj@a-star.edu.sg

multi-task learning objectives and physics-informed residual losses, providing practical insights for training PINNs that are both accurate and reusable. (4) Empirically, across various PDE problems including Poisson, Helmholtz, and Burgers equations, Pi-PINN achieves 100–1000× faster prediction/adaptation than conventional PINNs, while obtaining 10–100× lower relative error than typical data-driven models in sparse-data regimes (e.g., 2–4 training samples).

II. Pseudoinverse PINN

Without loss of generality, we consider the use of PINNs to model the outcomes u in any spatial-temporal domain $x \in \Omega, t \in (0, T]$ by satisfying the following governing equations:

$$\text{PDE:} \quad \mathcal{N}[u(x,t)] = h(x,t), \qquad x \in \Omega, t \in (0,T] \tag{1a}$$

$$\text{BC:} \quad \mathcal{B}[u(x,t)] = g(x,t), \qquad x \in \partial\Omega, t \in (0,T] \tag{1b}$$

$$\text{IC:} \quad u(x, t=0) = u_0(x), \qquad x \in \Omega \tag{1c}$$

where $\mathcal{N}[u(x,t)]$ is a general differential operator which can include linear and nonlinear combinations of temporal and spatial derivatives, and $h(x,t)$ is an arbitrary source term. Eq. 1b and Eq. 1c specify the BC and IC of the physical system, where $\mathcal{B}[u(x,t)]$ is also a differential operator.

To simplify the notation, we consider that the variation across different PDE instances (including variations in BC and IC) can be parameterized by a set of task parameters ϑ. Every new PDE instance leads to different physical outcomes, u_ϑ, spanning the entire spatial-temporal domain $x \in \Omega, t \in (0, T]$.

A PINN model with sufficient capacity (number of nodes n_{node}, hidden layers L) can effectively learn a good approximation to the target output, $u_{\text{PINN}}(x, t; \boldsymbol{w})$, by minimizing the PINN loss function w.r.t. its network parameters $\boldsymbol{w} = (\boldsymbol{W}_1, \mathbf{b}_1, ..., \boldsymbol{W}_{L-1}, \mathbf{b}_{L-1}, \mathbf{w}_L)$.

A. Pseudoinverse Physics-Informed $[Pi]^2$ Computation.

There has been some prior work on transfer learning in the PINN context [11]–[13]. These typically involve the transfer of network weights from a similar, previously-solved physics scenario as a better initialization for PINN training on a new task, i.e., minimizing the physics-informed loss by penalizing any violation of the PDE, BC/IC constraints made by the PINN model outputs on the target PDE. Generally, only the final few layers of the networks are fine-tuned in transfer learning, facilitating the exchange of valuable features acquired when training the PINN model for the previous task.

In the extreme, we can transfer the entire learned representation up to the final nonlinear hidden layer, i.e., $\mathbf{x}_L$, and only fine-tune the weight parameters in the linear output layer $\mathbf{w}_L$. In the absence of target labels for a new PDE instance, this can be done by means of physics-informed fine-tuning that purely minimizes violations of the PDE and BC/IC constraints by the PINN output $u_{\text{PINN}} = \mathbf{w}_L^T \mathbf{x}_L$ for the target PDE instance.

For linear PDEs, the process of fine-tuning the output weight parameters $\mathbf{w}_L$ reduces to a linear least-squares computation. This can be observed by examining the PDE, BC/IC constraints of Eq. 1, where the terms $\mathcal{N}[u_{\text{PINN}}(x,t;\boldsymbol{w})] - h(x,t)$, $u_{\text{PINN}}(x, t=0; \boldsymbol{w}) - u_0(x)$, and $\mathcal{B}[u_{\text{PINN}}(x,t;\boldsymbol{w})] - g(x,t)$, are all linear in terms of the output weights $\mathbf{w}_L$, as long as $\mathcal{N}[\cdot]$ and $\mathcal{B}[\cdot]$ comprise only linear differential operators (as-is true for linear PDE systems) [14], [15]. These constraints form a system of linear equations and can be efficiently solved utilizing the Moore-Penrose pseudoinverse [16], giving a least-squares-optimal solution to the unknown $\boldsymbol{w}_L$.

Given a target PDE and a set of PDE $(x^{(p)}, t^{(p)})$, BC $(x^{(q)}, t^{(q)})$, and IC $(x^{(r)}, t^{(r)})$ collocation points, the pseudoinverse computation solves the following weighted system of equations:

$$\begin{bmatrix} \lambda_{PDE}\ \mathcal{N}[\mathbf{x}_L^T(x^{(p)}, t^{(p)})] \\ \vdots \\ \lambda_{BC}\ \mathcal{B}[\mathbf{x}_L^T(x^{(q)}, t^{(q)})] \\ \vdots \\ \lambda_{IC}\ \mathbf{x}_L^T(x^{(r)}, t^{(r)}) \\ \vdots \end{bmatrix} \mathbf{w}_L = \begin{bmatrix} \lambda_{PDE}\ h(x^{(p)}, t^{(p)}) \\ \vdots \\ \lambda_{BC}\ g(x^{(q)}, t^{(q)}) \\ \vdots \\ \lambda_{IC}\ u_0(x^{(r)}) \\ \vdots \end{bmatrix}$$

$$\mathbf{X}\ \mathbf{w}_L = \mathbf{y} \tag{2}$$

where $\mathbf{w}_L$ are the weight parameters in the output layer to be solved, $\mathbf{x}_L$ is the transferable deep embeddings to be learned, and λ_{PDE}, λ_{BC}, and λ_{IC} are the weighting hyper-parameters. It has been previously reported in PINN literature that the different physics loss terms, such as PDE and BC/IC constraints, can have vastly different magnitudes, necessitating proper reweighting during training [17]. Similarly, during the pseudoinverse physics-informed computation, the relative weighting of PDE and BC/IC constraints can be adjusted.

Moreover, we can reformulate Eq. 2 into

$$[\lambda_{PI}\mathbf{I} + \mathbf{X}^T\mathbf{X}]\ \mathbf{w}_L = \mathbf{X}^T\mathbf{y} \tag{3}$$

with the regularization hyper-parameter $\lambda_{PI} \geq 0$. This regularization can significantly improve the conditioning of the problem, thereby improving the quality of the obtained solution. Additionally, Eq. 3 reduces the matrix size for the inversion, hence it is more efficient to solve.

Importantly, the pseudoinverse computation allows for rapid fine-tuning (in a single, fast computation) of weight parameters $\boldsymbol{w}_L$ in the final layer of the PINN model for a broad range of linear PDE problems, eliminating the need for gradient-based parameter updates over potentially long and complex convergence trajectories as have been previously reported [8]. It's also important to note that the pseudoinverse PINN approach can be seamlessly extended to handle nonlinear PDEs by using an iterative update process for the linearized version of the nonlinear constraints [18] (as demonstrated in the **Experiments** on the set of nonlinear Burgers' equation).

B. Pseudoinverse PINN Adapted From Data-Driven Neural Network: MLP+$[Pi]^2$

Given a set of training data $u_{\vartheta_1}, u_{\vartheta_2}, ..., u_{\vartheta_K}$ from K PDE instances $\vartheta_1, \vartheta_2, ..., \vartheta_K$, a natural first approach is to train a purely data-driven neural network, e.g., a multilayer

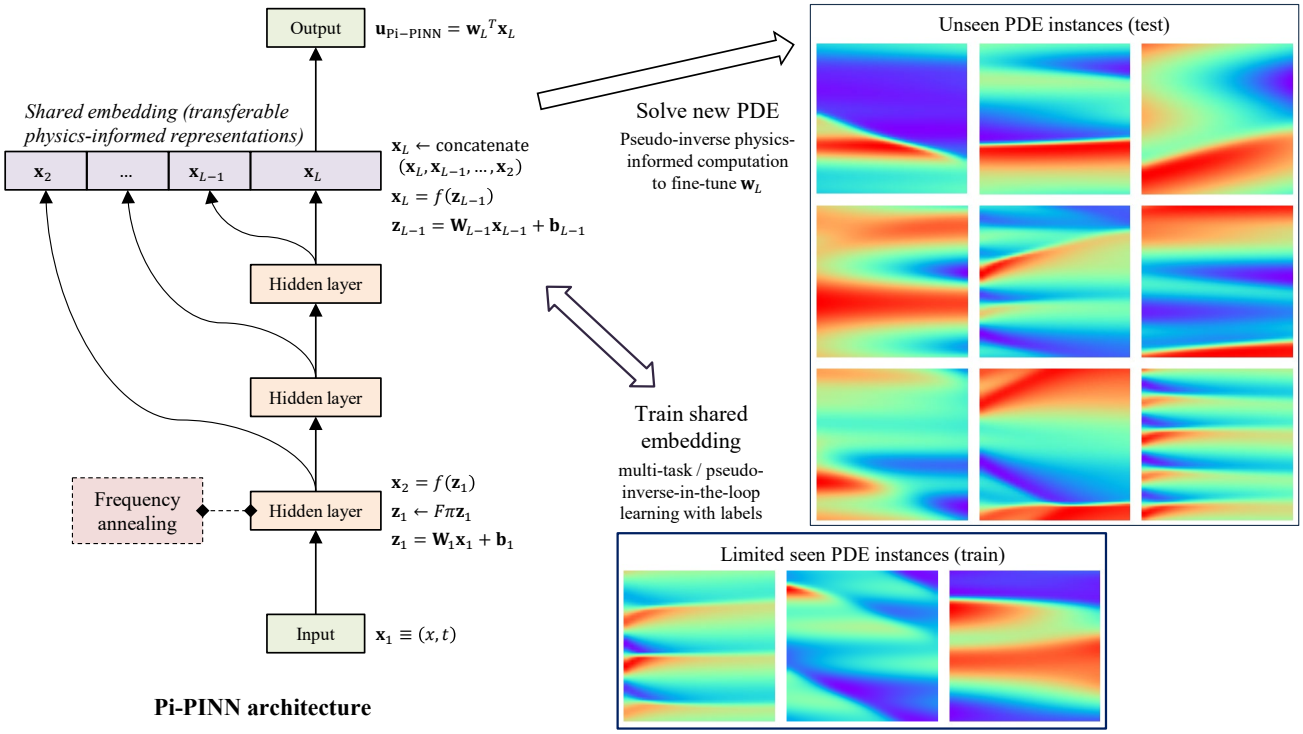


Fig. 1. The Schematic diagram of the fast Pseudoinverse PINN framework, with the example PDE instances (Burgers' equation).

perceptron (MLP), as the baseline model. We can define the mapping from the input variables (x, t, ϑ) to the output u_{MLP}.

The resulting learned MLP can then be used to make predictions for new PDE instances with a different ϑ. However, if the training data is sparse in ϑ, i.e., K is small, one may expect such predictions to have a very high chance of being inaccurate [19]. Naturally, the level of inaccuracy with respect to the actual K depends on the complexity of the underlying functions, as evident in the **Experiments** section.

Given the proliferation of data-driven models in the past decade, an interesting question is whether we can take the deep embeddings $\mathbf{x}_L$ learned from a data-driven MLP model and adapt it as a PINN model to solve new PDE instances more accurately, i.e., via the Pseudoinverse physics-informed computation. Crucially, this is a significant paradigm shift from the majority of previous related studies which have successfully demonstrated transfer learning purely between PINN models. This is significant as the training of data-driven models can be significantly less computationally expensive than current PINN methods, and we note the existence of some small set of training data in most real-world problems.

We refer to such data-driven-adapted PINN model as MLP+$[Pi]^2$. Although convenient for transferring prior knowledge from an existing data-driven-trained MLP on seen instances (via $\mathbf{x}_L$) to benefit the prediction of unseen PDE instances, the MLP+$[Pi]^2$ model is not the most effective approach for learning physics-informed representations. While the optimal balance of width and depth can vary with problem, it is common practice in deep learning to enhance learning and generalizability by stacking more hidden layers in an MLP model while maintaining a moderate number of nodes in each hidden layer (going deep rather than wide) [20]. When such an MLP model is used, there may be insufficient expressivity and/or degrees of freedom in the $\mathbf{x}_L$ for adaptation. Hence, the MLP+$[Pi]^2$ model may not adequately satisfy the PDE, BC/IC constraints on the target PDE instance even with the best-fit weight parameters $\mathbf{w}_L$, resulting in inferior predictions. This is empirically observed in the comparison study results in the **Experiments** section.

III. Deep Embeddings Neural Architecture and Learning Algorithms for Pi-PINN

The proper design of a neural architecture and learning algorithm for more performant Pi-PINN remains an open question worthy of more critical consideration. We hypothesize that a wide output layer $\mathbf{x}_L$ is critical as this greatly affects the learnability of a shared embedding space for solving new PDE instances, which in turn determines how well the Pi-PINN can effectively be fine-tuned with the weight parameters $\mathbf{w}_L$ in the output layer by the pseudoinverse computation. Effective learning of this shared embedding space is key to the transfer of prior knowledge to improve PINN performance, thereby ensuring predictions with good compliance to stipulated physics (encoded in the PDE, BC/IC constraints).

With the above requirements in mind, we propose a neural architecture that is potentially more appropriate for the Pi-PINN. Figure 1 gives the Pi-PINN schematic. Essentially, we modify the base MLP by utilizing concatenative skip

connections such that all the nonlinear hidden layers are concatenated at the output layer:

$$u_{\text{Pi-PINN}} = \mathbf{w}_L^T \mathbf{x}_L \qquad \textit{(Pi-PINN output layer)} \tag{4a}$$
$$\mathbf{x}_L \leftarrow \text{concatenate}(\mathbf{x}_L, \mathbf{x}_{L-1}, ..., \mathbf{x}_2) \tag{4b}$$
$$\mathbf{x}_L = f(\boldsymbol{z}_{L-1}) \tag{4c}$$
$$\vdots$$
$$\mathbf{x}_2 = f(\boldsymbol{z}_1) \qquad \textit{(after activation)} \tag{4d}$$
$$\boldsymbol{z}_1 \leftarrow F\pi \boldsymbol{z}_1 \qquad \textit{(frequency annealing)} \tag{4e}$$
$$\boldsymbol{z}_1 = \boldsymbol{W}_1 \mathbf{x}_1 + \mathbf{b}_1 \qquad \textit{(1st hidden layer)} \tag{4f}$$
$$\mathbf{x}_1 \equiv (x, t) \qquad \textit{(input)} \tag{4g}$$

This neural architecture design allows us to increase the output layer width by stacking multiple hidden layers while maintaining a moderate number of nodes in each hidden layer, thereby effectively improving the expressivity of the learned shared embedding and consequently, the prediction performance from data-driven-trained and physics-informed pseudoinverse computation. The proposed incorporation of a physics-informed loss computation during prediction also negates the need for explicit specification of the task parameter ϑ as input to the MLP for interpolation across different PDEs. It also encourages the learning of a generalizable embedding space for both seen and unseen PDE instances.

In addition, we note that this proposed concatenation for a more expressive shared embedding bears similarity to the way one constructs a polynomial basis space such as the monomial basis space. The additional operations at each hidden layer are analogous to the recurrence relations used in generating Chebyshev polynomials, and incorporation and concatenation of more hidden layers in the MLP essentially results in the creation of a larger (albeit finite) and more expressive basis space (with less truncation). This presents an interesting direction for design of even more performant neural architectures for Pi-PINN computations.

To more effectively learn the embedding space for high-frequency features, we use the sine activation for all the nonlinear hidden layers. In particular, we initialize the network with artificial high-frequency features through manipulating the first hidden layer with a factor of $F\pi$ (Eq. 4e), and allowing these frequencies to be naturally "reduced" during training—a process we refer to as "frequency annealing" [21] .

We propose two learning algorithms to train the shared embedding $\mathbf{x}_L$ (Eq. 4b) from a small set of *a priori*-obtained instances, and compare their adaptation performance in the **Experiments** section.

A. Multi-Task Learning with Multi-Output Model: HYDRA+$[Pi]^2$

The multi-task learning approach involves simultaneously learning multiple related tasks using a shared representation—in our case, the deep embeddings $\mathbf{x}_L$ in the Pi-PINN—for improved learning and generalization.

During training, $\mathbf{x}_L$ connects directly to as many output heads as there are different PDE instances $u_{\vartheta_1}, u_{\vartheta_2}, ..., u_{\vartheta_K}$ in the training set. Therefore, we can learn a shared embedding $\mathbf{x}_L$ that matches all seen PDE instances by optimizing the network weights $\boldsymbol{w}$ of a multi-output model to minimize the MSE between target and model outputs. It is worth noting that the model outputs for different PDE instances are controlled by different weight parameters in the output heads, thus creating a good synergy with pseudoinverse computation on new PDE instances. In contrast, the typical MLP model forces a shared output layer for different PDE instances. Multi-task learning with a multi-output model architecture has been explored in PINN literature and is called "Lernaean Hydra" [22], [23]. Hence, we follow the naming convention and refer to the first multi-task learning PINN model as HYDRA+$[Pi]^2$.

For predicting or solving new PDE instances, we discard the multi-output heads from the HYDRA+$[Pi]^2$ model and perform pseudoinverse physics-informed computation to find the best-fit weight parameters $\mathbf{w}_L$ on the desired PDE instance.

B. Pseudoinverse-In-The-Loop Learning: PiL-PINN

While the synergies between data-driven multi-task learning and pseudoinverse computation shed light on the success of the zero-shot PINN model, the HYDRA+$[Pi]^2$ does not ensure that the learned embedding is optimal for pseudoinverse physics-informed computation. We can further improve the adaptation performance of the Pi-PINN model through pseudoinverse-in-the-loop learning (albeit with higher training costs). This learning algorithm explicitly incorporates the pseudoinverse physics-informed computation during the training stage. The aim is to learn a shared deep embeddings $\mathbf{x}_L$ such that the outputs $u_{\vartheta_1}, u_{\vartheta_2}, ..., u_{\vartheta_K}$ can be best approximated with the pseudoinverse physics-informed computation.

We formulate a data-driven loss function $\mathcal{L}(\boldsymbol{w}^-, \Phi(\boldsymbol{w}^-))$ as the learning objective for PiL-PINN model optimization, where $\boldsymbol{w}^- = (\boldsymbol{W}_1, \mathbf{b}_1, ..., \boldsymbol{W}_{L-1}, \mathbf{b}_{L-1})$ includes all the network weights leading to the shared embeddings $\mathbf{x}_L$, and $\Phi(\boldsymbol{w}^-)$ comes from the pseudoinverse physics-informed computation for determining the best-fit weight parameters $\mathbf{w}_L$. Critically, the loss function is differentiable and gradient-based optimization methods can be applied. Moreover, we can perform mini-batch training by randomly sampling a subset of PDE instances in every training iteration for greater training efficiency. We refer to our pseudoinverse-in-the-loop learned PINN model as PiL-PINN.

IV. Experiments

A. Experimental Setup and Implementation

We conduct extensive experiments on various real-world PDE problems to show the effectiveness of the different Pi-PINN approaches, i.e., MLP+$[Pi]^2$, HYDRA+$[Pi]^2$, and PiL-PINN. On different problems, we train these models with $K(= 2, 4, 8, ...)$ PDE instances with *a priori-provided* solutions, $u_{\vartheta_1}, u_{\vartheta_2}, ..., u_{\vartheta_K}$, and apply the model to predict or solve for remaining PDE instances with no labeled data based on the pseudoinverse physics-informed computation (i.e. zero-shot adaptation). We then compute the relative L2 error on these unseen tasks.

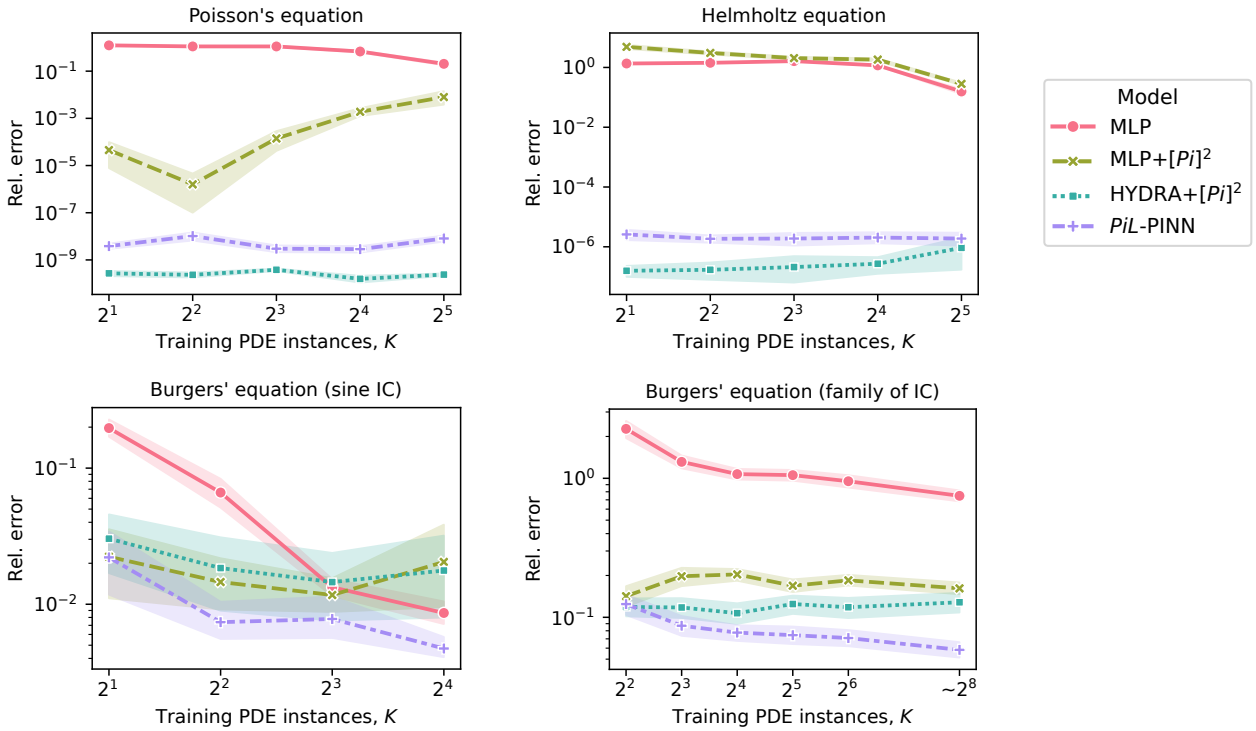


Fig. 2. Comparison of baseline MLP and different Pi-PINN approaches for predicting or solving different PDE problems.

For completeness, we also compare these PINN models with a data-driven MLP, which predicts the unseen PDE instances by interpolating along task parameter ϑ. Below are brief descriptions to the problems.

1) Poisson's Equation: Poisson's equation is a widely studied, representative problem for the broad class of elliptic PDEs. As the solution to potential fields in many physical systems, it can model diverse physics phenomena in electromagnetism and heat and mass transfer (e.g., heat conduction) among many others. In this study, the following 1D Poisson's equation is investigated [24]:

$$\frac{\partial^2 u}{\partial x^2} = h \quad , x \in [-10, 10] \tag{5}$$

with the exact solution $u(x; \omega_1, \omega_2) = \sin(\omega_1 x) + \sin(\omega_2 x) - 0.1x$. This exact solution is used to derive the corresponding BCs and source term $h(x; \omega_1, \omega_2) = -\omega_1^2 \sin(\omega_1 x) - \omega_2^2 \sin(\omega_2 x)$. We generate 100 PDE instances with ω_1 and ω_2 uniformly sampled from $(0, 1]$ and $(0, 2]$ respectively. Each PDE instance contains 201 spatial points.

2) Helmholtz Equation: Helmholtz equation is another well-known model equation that can represent interesting physics such as wave propagation in both acoustics and electromagnetics. In this study, the 2D Helmholtz equation is investigated [25]:

$$\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + u^2 = h \quad , x \in [-1, 1], y \in [-1, 1] \tag{6}$$

with the exact solution $u(x, y; \alpha_1, \alpha_2) = \left(1 - (\alpha_1 \pi)^2 - (\alpha_2 \pi)^2\right) \sin(\alpha_1 \pi x) \sin(\alpha_2 \pi y)$. This exact solution is used to derive the corresponding BCs and source term $h(x, y; \alpha_1, \alpha_2) = (1 - (\alpha_1 \pi)^2 - (\alpha_2 \pi)^2) \sin(\alpha_1 \pi x) \sin(\alpha_2 \pi y)$. We generated 100 PDE instances with (α_1, α_2) uniformly sampled from $(0, 6]$. Each PDE instance contains 64×64 spatial points.

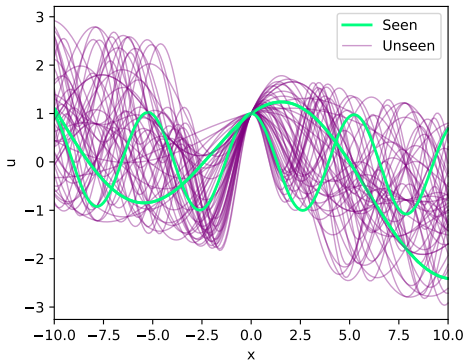


Fig. 3. The solution of different PDE instances of the Poisson equation problem. Seen examples for model training at $K = 2$ is highlighted in green.

3) Burgers' Equation (Sine IC): The Burgers' equation is the model equation for a class of transport phenomena of quantities such as momentum, species transport and heat and mass transfer. We consider the nonlinear Burgers equation as defined by [1], [24]:

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} - \gamma \frac{\partial^2 u}{\partial x^2} = 0 \quad , x \in [-1, 1], t \in (0, 1] \tag{7}$$

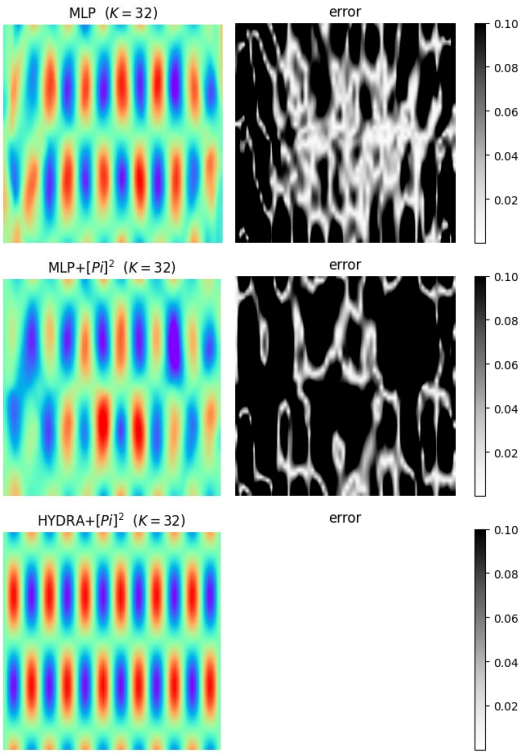


Fig. 4. The predicted results and corresponding errors for the Helmholtz equation on new PDE instance ($\alpha_1 = 1.2, \alpha_2 = 6.0$), for MLP, MLP+$[Pi]^2$, and HYDRA+$[Pi]^2$ trained with $K = 32$ PDE instances.

subject to IC $u(x, t = 0) = -\sin(\pi x)$, where the flow solution u represents nonlinear waveform propagation with both compression and rarefaction effects, and γ is the viscosity of the fluid. A steep gradient can be observed around $x = 0$ after some time t. The problem consists of 50 PDE instances with γ uniformly sampled from $[0.001, 0.05]$. Each PDE instance contains 129×51 spatial-temporal points.

4) Burgers' Equation (Family of IC): We further consider the following generalized nonlinear Burgers' equation with a source term,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} - \gamma\frac{\partial^2 u}{\partial x^2} = h \quad , x \in [0, 1], t \in (0, 0.5] \tag{8}$$

with periodic BC $u(0, t) = u(1, t)$. We fix the viscosity $\gamma = 0.005$, but randomly generate the source function $h(x, t) = \sum_{j=1}^{5} A_j \sin(\omega_j t + \frac{2\pi l_j x}{6} + \varphi_j)$ with the coefficients sampled from $A_j \in [-0.8, 0.8]$, $\omega_j \in [-2, 2]$, $l_j \in [0, 1, 2, 3, 4]$, $\varphi_j \in [-\pi, \pi]$. The problem consists of 480 PDE instances with different source terms and ICs (from $u(x, t = 0) = h(x, t = 0)$ and its perturbations), representing a rich diversity of dynamical processes. Each PDE instance contains 51×26 spatial-temporal points.

Hyperparameters setting. We fix $\lambda_{BC} = \lambda_{IC} = 1$ and perform a grid search to determine an appropriate λ_{PDE} and λ_{PI} for the MLP+$[Pi]^2$ and HYDRA+$[Pi]^2$ models based on the seen PDE instances, before applying them to solve the unseen PDE instances. For the PiL-PINN, we pass these hyper-parameters to the loss function and jointly learn them with the network weights during model optimization. This is an additional advantage of the pseudoinverse-in-the-loop learning. The number of nonlinear iterations for Burgers' equation (sine IC and family of IC) is 4 and 8 respectively.

Compute environment. We implement our models in JAX and perform the experimental study on a workstation with Intel Xeon W-2275 Processor and 2 NVIDIA RTX 3090 GPUs.

B. Results and Discussion

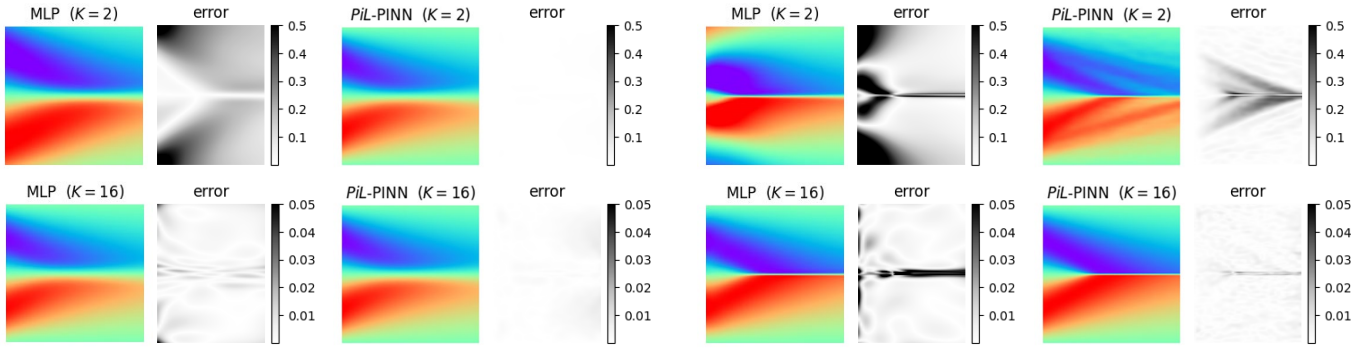
Given the underlying context that data is typically sparse in many science and engineering real-world scenarios, we evaluate the performance of the various models (MLP, MLP+$[Pi]^2$, HYDRA+$[Pi]^2$, and PiL-PINN) across all 4 problems for a range of scenarios where solutions to K randomly sampled tasks are available for training. The main results are in Figure 2. A representative plot of the solutions for Poisson's equation problem is given in Figures 3. Representative plot of the solutions obtained by the different methods for the Helmholtz equation and nonlinear Burgers' equation (Sine IC and Family of IC) are given in Figures 4, 5 and 6.

Improvement in Prediction Performance through MLP+$[Pi]^2$. In general, we note that the data-driven MLP's predictive performance is fairly poor, with relative errors ranging from 0.1 to 1, which is intuitively consistent with the small amount of training data provided and the complexity of the model physics. The predicted solutions are visibly inaccurate in Figures 5 and 4.

Excitingly, the combination of a simple data-driven training strategy and an extra psuedoinverse computation to fine-tune the final layer to better match the underlying physics-based constraints still results in substantial improvements in relative error for the Poisson's equation (2 orders of magnitude reduction) and two Burgers' equation problems (1 order of magnitude reduction) as shown in Figure 2. These experiments clearly show the effectiveness of a physics-informed loss in improving the prediction of data-driven models under conditions where the available labeled data is small. The results also open the possibility for greater utilization of other data-driven models and architectures in future extensions. We note that a reason for the widespread usage of MLP architectures in PINN literature is the need for simple implementations given the complexity of PINN training. The potential extension with a physics-informed loss through fast pseudoinverse computations can greatly mitigate this issue.

In addition, the benefit from including the physics-informed loss is most pronounced when K is smaller. As K increases, the data-driven MLP model's relative error begins to show a decreasing trend. However, we note that the MLP+$[Pi]^2$ generally shows reasonable performance even at low K in 3 of the 4 problems. Nonetheless, the data-driven MLP training does not always learn a good shared embedding for the physics-informed loss, as exemplified by the relatively higher errors in the Helmholtz problem.

Importance of an Expressive Shared Embeddings. Given the importance of expressiveness of the learned embedding for the pseudoinverse computation to minimization of the



(b) $\gamma = 0.05$ (b) $\gamma = 0.002$

Fig. 5. The predicted results and corresponding errors for the nonlinear Burger's equation (sine IC), for MLP and PiL-PINN with different PDE instances K.

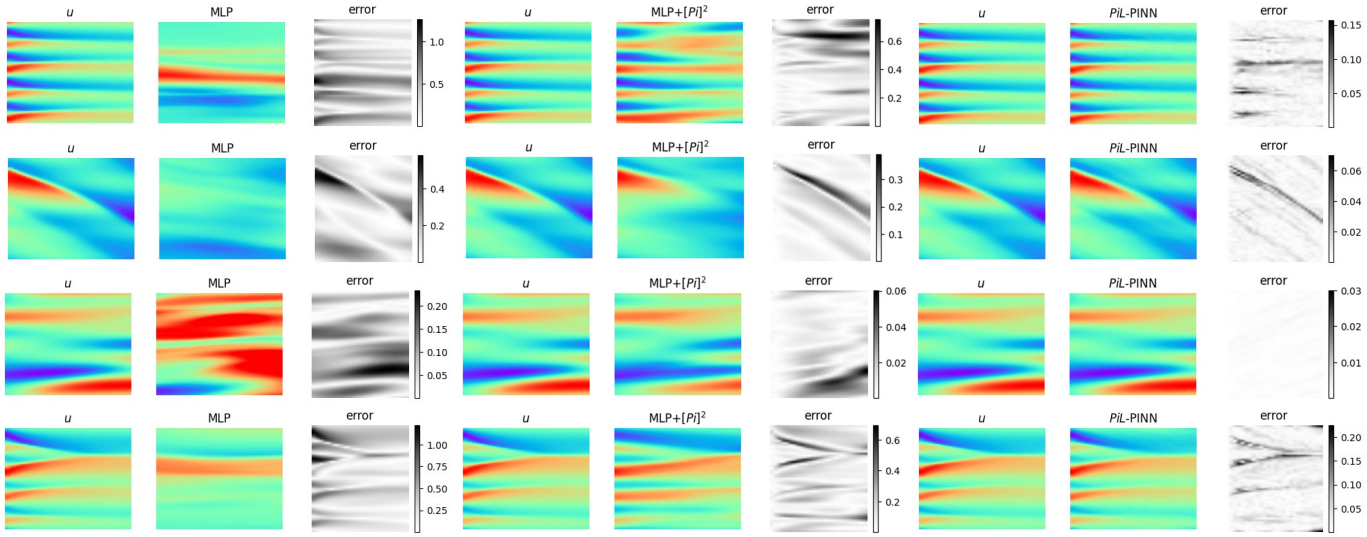


Fig. 6. The predicted results and corresponding errors for the Burgers' equation (family of IC) on new PDE instances, for MLP, MLP+$[Pi]^2$, and Pil-PINN trained with $K = 240$ PDE instances.

physics-informed loss, we further compare the proposed neural architecture (with concatenative skip connections across the hidden layers) in the context of the multi-task learning setup.

From Figure 2, the HYDRA+$[Pi]^2$ shows significantly improved relative errors compared to the MLP+$[Pi]^2$ models. This improvement is particularly notable for the Poisson equation and the Helmholtz equation in Figure 4, showing the importance of our proposed neural architecture in enabling the learning of a more expressive and performant shared embedding for the pseudoinverse computation. We also notice that the performance of HYDRA+$[Pi]^2$ (and PiL-PINN models) can be further improved by the use of an appropriate factor $F\pi$ in Eq. 4e ($F = 2$ for Poisson's and 4 for Helmholtz problems).

While the improvement was notable for the linear PDEs (Poisson's and Helmholtz equation), the reduction in the relative error for HYDRA+$[Pi]^2$ relative to the MLP+$[Pi]^2$ was much less significant for the nonlinear Burgers' equation. This is potentially due to the fact that the current zero-shot framework fundamentally only finds the best linear solution (through the pseudoinverse).

PiL-PINN for Nonlinear PDEs. As the nonlinear PDEs are currently solved in a slightly modified process through a series of iterations where the equation is linearized, a different way of learning the shared embedding where the pseudoinverse is taken explicitly into account during training can be more appropriate. Hence, we proposed the PiL-PINN methodology to explicitly handle this nuance.

As per Figure 2, the PiL-PINN shows a further reduction in relative errors for the nonlinear Burgers' problems relative to the other 3 methods (improvement relative to the MLP method is presented in Figure 5 and Figure 6). In addition, the PiL-PINN results also show a significant reduction in relative error with K, indicating that the proposed method indeed allows for better learning of a shared embedding for zero-shot prediction as observed for Burgers' equation.

Computation Time. The training time for the PiL-PINN models range from a few seconds on the Poisson's problem to 1.5 hours, on the Burgers' (family of IC) problem ($K = 240$). The PiL-PINN model takes less than 1 second to make predictions on a new PDE instance across all the problems

tested. For example, it takes 54ms to make a prediction for the Burgers' equation (sine IC)—a benchmark problem frequently studied in PINN literature—at 129x51 spatial-temporal points (Figure 5). In contrast, others in literature have reported typical training times of $\approx$10 min to 1 hour for a wide range of canonical PDE problems [26]. This training time is also consistent with our own experience when training single PINN models to produce predictions for Burgers', Poisson, or Helmholtz equation. This means that the Pi-PINN method can produce predictions for new scenarios 100–1000$\times$ faster.

V. Conclusions

In this work, we demonstrate a transferable learning approach that enables rapid and accurate physics-informed solutions by incorporating a least-squares-optimal pseudoinverse computation within a closed-form head adaptation step. This approach underscores the potential of combining data-driven multi-task learning with physics-informed learning to improve PINNs' efficiency and generalization across PDE families and parameter regimes. Even when the labeled data are limited, our results show that Pi-PINN can learn a shared, transferable embedding that supports significantly improved generalization to new PDE instances. The empirical results also suggest that architectural modifications such as concatenative skip connections are beneficial. Nonetheless, there remains substantial scope for developing neural architectures and training algorithm designs that learn transferable embeddings even more effectively, which is particularly important given the practical constraint that datasets in many scientific and engineering domains may be limited in quantity. We hope this work encourages alternative directions that move PINNs closer to robust, reusable tools for real-world applications.

Acknowledgment

This research was in part supported by the National Research Foundation, Singapore through the AI Singapore Programme, under the project "AI-based urban cooling technology development" (Award No. AISG3-TC-2024-014-SGKR). The authors acknowledge the use of AI tools for language editing and take full responsibility for the final manuscript.

References

[1] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational physics*, vol. 378, pp. 686–707, 2019.

[2] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nature Reviews Physics*, vol. 3, no. 6, pp. 422–440, 2021.

[3] N. Wandel, M. Weinmann, M. Neidlin, and R. Klein, "Spline-pinn: Approaching pdes without data using fast, physics-informed hermite-spline cnns," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 8, 2022, pp. 8529–8538.

[4] J. Kim, K. Lee, D. Lee, S. Y. Jhin, and N. Park, "Dpm: A novel training method for physics-informed neural networks in extrapolation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 9, 2021, pp. 8146–8154.

[5] N. Kang, B. Lee, Y. Hong, S.-B. Yun, and E. Park, "Pixel: Physics-informed cell representations for fast and accurate pde solvers," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 7, 2023, pp. 8186–8194.

[6] Z. Yang, Z. Qiu, and D. Fu, "Dmis: Dynamic mesh-based importance sampling for training physics-informed neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 4, 2023, pp. 5375–5383.

[7] J. C. Wong, C. Ooi, A. Gupta, and Y.-S. Ong, "Learning in sinusoidal spaces with physics-informed neural networks," *IEEE Transactions on Artificial Intelligence*, 2022.

[8] A. Krishnapriyan, A. Gholami, S. Zhe, R. Kirby, and M. W. Mahoney, "Characterizing possible failure modes in physics-informed neural networks," *Advances in Neural Information Processing Systems*, vol. 34, pp. 26 548–26 560, 2021.

[9] S. Wang, Y. Teng, and P. Perdikaris, "Understanding and mitigating gradient flow pathologies in physics-informed neural networks," *SIAM Journal on Scientific Computing*, vol. 43, no. 5, pp. A3055–A3081, 2021.

[10] S. Cuomo, V. S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, and F. Piccialli, "Scientific machine learning through physics–informed neural networks: Where we are and what's next," *Journal of Scientific Computing*, vol. 92, no. 3, p. 88, 2022.

[11] J. C. Wong, A. Gupta, C. C. Ooi, P.-H. Chiu, J. Liu, and Y.-S. Ong, "Evolutionary optimization of physics-informed neural networks: Evo-pinn frontiers and opportunities," *IEEE Computational Intelligence Magazine*, vol. 21, no. 1, pp. 16–36, 2026.

[12] J. C. Wong, A. Gupta, and Y.-S. Ong, "Can transfer neuroevolution tractably solve your differential equations?" *IEEE Computational Intelligence Magazine*, vol. 16, no. 2, pp. 14–30, 2021.

[13] S. Goswami, C. Anitescu, S. Chakraborty, and T. Rabczuk, "Transfer learning enhanced physics informed neural network for phase-field modeling of fracture," *Theoretical and Applied Fracture Mechanics*, vol. 106, p. 102447, 2020.

[14] S. Dong and Z. Li, "Local extreme learning machines and domain decomposition for solving linear and nonlinear partial differential equations," *Computer Methods in Applied Mechanics and Engineering*, vol. 387, p. 114129, 2021.

[15] F. Calabrò, G. Fabiani, and C. Siettos, "Extreme learning machine collocation for the numerical solution of elliptic pdes with sharp gradients," *Computer Methods in Applied Mechanics and Engineering*, vol. 387, p. 114188, 2021.

[16] R. Penrose, "On best approximate solutions of linear matrix equations," *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 52, pp. 17–19, 1956.

[17] S. Wang, H. Wang, and P. Perdikaris, "On the eigenvector bias of fourier feature networks: From regression to solving multi-scale pdes with physics-informed neural networks," *Computer Methods in Applied Mechanics and Engineering*, vol. 384, p. 113938, 2021.

[18] J. C. Wong, C. C. Ooi, A. Gupta, P.-H. Chiu, J. S. Z. Low, M. H. Dao, and Y.-S. Ong, "Evolutionary optimization of physics-informed neural networks: Advancing generalizability by the baldwin effect," *IEEE Transactions on Evolutionary Computation*, 2026.

[19] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 843–852.

[20] T. Nguyen, M. Raghu, and S. Kornblith, "Do wide and deep networks learn the same things? uncovering how neural network representations vary with width and depth," *arXiv preprint arXiv:2010.15327*, 2020.

[21] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla, "Nerfies: Deformable neural radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5865–5874.

[22] Z. Zou and G. E. Karniadakis, "L-hydra: Multi-head physics-informed neural networks," *arXiv preprint arXiv:2301.02152*, 2023.

[23] S. Desai, M. Mattheakis, H. Joy, P. Protopapas, and S. Roberts, "One-shot transfer learning of physics-informed neural networks," *arXiv preprint arXiv:2110.11286*, 2021.

[24] X. Liu, X. Zhang, W. Peng, W. Zhou, and W. Yao, "A novel meta-learning initialization method for physics-informed neural networks," *Neural Computing and Applications*, vol. 34, no. 17, pp. 14 511–14 534, 2022.

[25] W. Cho, K. Lee, D. Rim, and N. Park, "Hypernetwork-based meta-learning for low-rank physics-informed neural networks," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[26] S. Wang, S. Sankaran, H. Wang, and P. Perdikaris, "An expert's guide to training physics-informed neural networks," *arXiv preprint arXiv:2308.08468*, 2023.