Physics-Guided Machine Learning from Simulated Data with Different Physical Parameters

Shengyu Chen[1], Nasrin Kalanat[1], Yiqun Xie[2], Sheng Li[3], Jacob A. Zwart[4], Jeffrey M. Sadler[4,5], Alison P. Appling[4], Samantha K. Oliver[4], Jordan S. Read[4,6] and Xiaowei Jia[1*]

[1*]University of Pittsburgh, United States.
[2]University of Maryland, United States.
[3]University of Virginia, United States.
[4]U.S. Geological Survey, United States.
[5]Oklahoma State University, United States.
[6]Consortium of Universities for the Advancement for Hydrologic Science, Inc., United States.

*Corresponding author(s). E-mail(s): xiaowei@pitt.edu;
Contributing authors: shc160@pitt.edu; nak168@pitt.edu; xie@umd.edu; shengli@virginia.edu; jzwart@usgs.gov; jsadler@usgs.gov, jeff.sadler@okstate.edu; aappling@usgs.gov; soliver@usgs.gov; jread@usgs.gov, jread@cuahsi.org;

Abstract

Physics-based models are widely used to study dynamical systems in a variety of scientific and engineering problems. However, these models are necessarily approximations of reality due to incomplete knowledge or excessive complexity in modeling underlying processes. As a result, they often produce biased simulations due to inaccurate parameterizations or approximations used to represent the true physics. In this paper, we aim to build a new physics-guided machine learning framework to monitor dynamical systems. The idea is to use advanced machine learning model to extract complex spatio-temporal data patterns while also incorporating general scientific knowledge embodied in simulated data

generated by the physics-based model. To handle the bias in simulated data caused by imperfect parameterization, we propose to extract general physical relations jointly from multiple sets of simulations generated by a physics-based model under different physical parameters. In particular, we develop a spatio-temporal network architecture that uses its gating variables to capture the variation of physical parameters. We initialize this model using a pre-training strategy that helps discover common physical patterns shared by different sets of simulated data. Then we fine-tune it combining limited observations and adequate simulations. By leveraging the complementary strength of machine learning and domain knowledge, our method has been shown to produce accurate predictions, use less training samples and generalize to out-of-sample scenarios. We further show that the method can provide insights about the variation of physical parameters over space and time in two domain applications: predicting temperature in streams and predicting temperature in lakes.

Keywords: Physics-guided machine learning, Spatio-temporal data, Deep learning, Freshwater science, Stream networks, Simulated data

1 Introduction

Physics-based models, which are also referred to as process-based models or mechanistic models, have been widely used to study scientific and engineering systems in domains such as hydrology [1], climate science [2], and material science [3]. Even though physics-based models are based on known physical laws that govern relations between input and output variables, most physics-based models are necessarily approximations of reality due to incomplete knowledge of certain processes or excessive complexity in modeling these processes; e.g., see a series of debate papers in hydrology [4–6]. For example, existing physics-based approaches for predicting river networks simulate target variables (e.g., streamflow and temperature) based on general physical relations such as energy and mass conservation. However, the model predictions still rely on parameterizations of land surface and subsurface processes based on soil and surficial geologic classification along with topography, land cover, and climate input. Hence, such models have limits of prediction performance even after parameter calibration due to constraints from the model structure and simplified representation (e.g., by assuming physical parameters are static in space and/or time). Furthermore, calibration of physics-based models often requires extensive expert knowledge of the system and can be extremely time intensive due to the complex (sometimes chaotic) dynamics in the system and uncertainty in observations, initial conditions, and model error. The limitations of physics-based models cut across disciplinary boundaries and are well known in the scientific community.

Recent advances in machine learning (ML) make it possible to capture spatial and temporal dependencies from data of great complexity. These ML

techniques have found tremendous success in several commercial applications, e.g., computer vision [7] and natural language processing [8, 9]. Given the success of ML in these commercial domains, researchers have started to use ML approaches for advancing scientific discovery [10, 11], and this idea has been pursued in diverse disciplines, such as hydrology [12], Earth systems [13], and climate science [14–16]. The use of ML models is especially promising when relevant physical processes are not completely understood by our current body of knowledge due to the inherent complexity of the underlying phenomenon. State-of-the-art ML models (e.g., deep learning models), given enough data, can often achieve better predictive performance than physics-based models [17, 18]. Early results in isolated and relatively simple scenarios have been promising, and the expectations are rising for this paradigm to accelerate scientific discovery and help address some of the biggest challenges that are facing humanity such as food and water security. However, direct application of "black-box" ML models has had limited success in some scientific domains, given that the data available for many scientific problems are far smaller than what are needed to effectively train advanced ML models. Moreover, in the absence of adequate information about the physical mechanisms of real-world processes, ML approaches are prone to false discoveries of patterns that cannot generalize to out-of-sample scenarios.

In recent years, there has been a great interest in developing new approaches that integrate scientific knowledge into ML models (e.g., see a recent survey [11]). From the earliest residual modeling approaches, where an ML model is trained to predict the discrepancy between observations and simulations made by a physics-based model [12, 19, 20], researchers have now shifted their focus to new methods that leverage knowledge of physics to guide the learning process of ML models. This includes new loss functions to preserve consistency with established physical laws [21–27], new model initialization methods by transferring physics [21, 24, 28, 29], and new model architectures by encoding specific physical relationships [30–36]. In particular, previous work has shown that ML models can learn more generalizable patterns from limited observation data by transferring knowledge from simulations produced by physics-based models [21, 24, 37].

However, there are two major challenges faced by these methods when applied to real-world scientific problems. First, these methods can require access to a physics-based model that well simulates the target system, which is often not feasible given the high cost of calibration and/or parameterization and the prediction errors that persist even after these procedures. The parameters of a physics-based model modulate the translation of input drivers to predictions of target variables. For example, given the same meteorological drivers for a lake system, the physics-based model can simulate different water temperature profiles by varying the parameter of water clarity, which controls how much light can penetrate into the water column and warm deeper waters. When transferring physics knowledge to an ML model, existing methods are likely to be affected by the inherent bias due to uncertainties remaining

after parameterization thereby limiting the model's potential to extract general physical knowledge from the physics-based model. Second, existing methods commonly use physical simulations in a separate training stage [21] or for feature augmentation [23] without fully exploring the relationships between simulations and true observations. Learning such relationships has the potential to identify simulation biases and variations of physical parameters over space and time.

In this paper, we propose a new framework, **SIM**ulation-guided **L**ea**R**ning (SIMLR), which extracts the general physical knowledge jointly from multiple sets of physical simulations with imperfect parameterizations. We also explore the relationship between observation data and simulated data and identify parameter settings that produce the most accurate predictions over different locations and time periods. In particular, we first build a spatial-temporal network (STN) architecture to represent the spatial and temporal relationships in the dynamical system. Given that most physical parameters determine specific conditions that control how the system states react to external changes, we represent such conditioning factors using a set of gating variables in the ML architecture. The gating variables are used to filter the information from the current time step, previous time steps, and the spatial neighborhood. The filtered information is combined to update the state of the ML model.

Then we propose a new pre-training strategy that leverages general physical patterns from different sets of simulated data to inform the initialization of the STN model. [1] This pre-training process can also leverage many existing simulation datasets [39–41] and does not require true observed labels, which are often expensive to collect. The idea is that the initialized model obtained through pre-training can be easily adjusted to fit each set of simulated data by slightly altering gating variables. After the initialization, we further refine the model using true observations and simulations. Specifically, we propose two fine-tuning approaches, the contrastive learning method and the attention-based ensemble learning method. The contrastive learning process aims to explore the similarity of relations between observations and different sets of simulated data and further transfer the knowledge from specific simulations that are closer to the observed reality. The ensemble learning method aggregates the output of STN and different sets of simulated data and assigns them different weights based on an attention mechanism.

We evaluate the performance in two societally relevant applications, modeling water temperature in a lake system and water temperature in river networks. Although predicting the same variable, these two applications have distinct spatiotemporal drivers of water temperature and focal parameters for physics-based calibration. We demonstrate the effectiveness of model initialization using general physical knowledge and show that our method can achieve good predictive performance even with very sparse observation data. We also analyze the similarity relationships learned from the fine-tuning process and

[1] This is an extension of our previous conference paper [38].

provide scientific interpretability. Our method has shown promise in discovering variations of physical parameters across space and time while traditional physics-based model can often take fixed parameter values. Moreover, we show that our method under the guidance of general physical relationships can better generalize to different scenarios. We have released our code and the river dataset in a Google Drive link[2].

Our contributions can be summarized as follows:

- We build a new model architecture STN and use it to extract general physical knowledge from multiple sets of simulated data during the pre-training process.
- We leverage both the observation data and simulated data in the model fine-tuning process. In particular, we use a contrastive loss function to explore the similarity between observations and each set of simulated data. The ensemble learning method dynamically adjusts different weights for combining observations and simulated data.
- We have implemented the proposed methods in two societally relevant applications. The proposed method not only improves the predictive performance using limited data, but also reveals the variation of physical parameters over space and time.

2 Related Work

Recently, we have seen an increasing interest of integrating physics into ML models for a variety of scientific applications. The objective is to improving the predictive performance and generalizability in addressing scientific problems. This is commonly conducted in several ways, including applying additional loss functions to enforce physical consistency [21, 23, 24], developing new model architectures to encode intermediate physical variables or relationships [32, 42, 43], and transferring knowledge from physical simulations [21, 29, 44, 45].

The idea of including an additional term in the loss function to prefer solutions that are consistent with domain specific knowledge is beginning to find extensive use in many scientific applications. For example, Karpatne et al. [23] use a physics-based loss that ensures that denser water are located at lower depths in lake systems. Jia et al. [21, 37] and Read et al. [24] further extend this work to capture even more complex and general physical relationships that happen on a temporal scale. Specifically, they create a loss function to ensure that the lake thermal energy gain across time is consistent with the net thermodynamic fluxes in and out of the lake, which is a known as the energy conservation law in the lake system. Another benefit of this approach is that the physics-based loss allows training in absence of labels, since it can often be computed even in absence of class labels or target variables. Moreover, the regularization by the physics-based loss can reduce the search space for training the ML model, and thus requires less amount of labeled training samples. Despite the promise of this method, many dynamical systems are driven by

[2] https://drive.google.com/open?id=12l9RhiaGZqwZEp3URFY8GrQ4VAMpRtvy

complex physical processes that are non-differentiable or difficult to explicitly include in the loss function. More importantly, many equations that are used to build physics-based models (except first principles) are only approximations of reality due to the incomplete knowledge about underlying processes. Hence, it requires additional effort to adjust the weight of physics-based loss to avoid performance degradation in real scenarios.

Another way is to modify model architectures to reflect known physics. For example, Muralidlar et al. [31] insert physical variables as the intermediate variables in the convolutional neural network (CNN) architecture which achieved significant improvement over state-of-the-art physics-based models on the problem of predicting drag force on particle suspensions in moving fluids. In the context of modeling stream networks, Jia et al. [46] introduce a PGRGrN model, which uses additional physical variables to enforce that the advected energy fluxes are propagated from upstream to downstream rivers through the graph convolution process. The PGRGrN model capture the spatial and temporal patterns using graph network layer and recurrent network layer, respectively, and uses simulations to enforce energy conservation. However, it can be affected by biased simulations with imperfect physical parameters. Our work is also inspired by the prior work on modifying gating variables to filter the effect of input data for predicting streamflow in basins [47]. The intuition of this work is that the response of streamflow given climate input drivers also depend on catchment characteristics, and thus catchment characteristics can be used to create the input gate in LSTM.

The design of physics-based loss functions and architectures often requires explicit physics. In contrast, another branch of research aims to transfer knowledge directly from simulated data produced by physics-based models. The most common approach for using simulated data is residual modeling, where an ML model is trained to make corrections to physical model outputs. Most of the work on residual modeling going back several decades has used plain regression models [12, 19], although some recent works [20] have used Long-Short Term Memory (LSTM). Karpatne et al. extended residual modeling by feeding the output of a physics model into an ML model as additional input [23]. More recently, simulated data have been used for pre-training ML models with the aim of improving the initialization of ML models. Intuitively, if physical or other contextual knowledge can be used to help inform the initialization of the weights, model training can be accelerated or improved while also requiring less training samples [21]. One way to inform the initialization to assist in model training and escaping local minima is to use an ML technique known as transfer learning. In transfer learning, a model can be pre-trained on physics-based model's simulated data prior to being fine-tuned with limited training data to fit the desired task. The pre-trained model serves as an informed initial state that ideally is closer to the desired parameters for the desired task than random initialization. For example, Jia et al. used this strategy in the context of modeling lake temperature dynamics [21]. They pre-trained their Physics-Guided Recurrent Neural Network (PGRNN) models for lake temperature

modeling on simulated data generated from a physics-based model and fine tuned the network with small observed data. They showed that pre-training, even using data from a physical model with an incorrect set of parameters, can still reduce the training data needed for a quality model. In addition, Read et al. [24] demonstrated that such models are able to generalize better to unseen scenarios than pure physics-based models. Such pre-training methods have also been explored in computational biophysics [29], chemistry [45, 48], and climate science [44].

Our method differs from these existing works in that it aims to extract general physical relationships from multiple sets of simulated data produced using several default physical parameters. Moreover, the variation of physical parameters can be reflected in the model architecture as we use different gating variables for different sets of simulated data during the pre-training process. Furthermore, the fine-tuning process combines the observations and the knowledge from simulated data to enhance the model. Our work is also relevant to existing work on learning from multiple noisy annotators [49]. The difference is that we aim to use physical simulations to initialize the ML model and then use true observations to adjust the model and identify the gaps and similarities with different parameter settings. Also, the bias introduced by simulated data are not randomly generated but caused by the deviation of physical parameters used in the governing equations.

The objective of using simulations is to enhance the learning of complex data patterns governed by physical relationships using limited observation samples. There have been multiple machine learning approaches developed to address the data paucity issues, such as semi-supervised learning [50, 51] and transfer learning [52, 53]. For example, the prior work [54] explores the class-wise domain-invariant features to facilitate the model transfer from a source domain to another target domain. Another promising work incorporates unlabeled data through posterior regularization in a Bayesian framework while also using the Tikhonov regularization to ensure the smoothness of the smoothness of the predictions [55]. These methods aim to explore certain structures or feature invariance in the input feature space, but they are unable to capture the joint distribution of input features and target variables under different scenarios. Many transfer learning methods and self-supervised learning methods [56, 57] also use a similar pre-training idea, but they are focused on extracting features that are representative of certain input data dependencies. These methods can still be limited given a large hypothesis space and limited observations. In such ill-posed problem setting, multiple mappings from input to output can be valid as they match training observations. However, many of these mappings are not consistent to underlying physical processes and thus they should not be selected. The use of simulated data generated by physics-based models can aid in reducing the hypothesis space and selecting physically consistent mappings. Recent advances on active learning for network data [58] also provide opportunities for addressing the data scarcity issue in scientific

systems with interacting processes, but active learning is beyond the scope of this paper.

3 Problem Definition

Our objective is to predict target variables for each location $i \in \{1, ..., N\}$, and on each date $t \in \{1, ..., T\}$, given input physical variables that drive the dynamics of the physical system. Specifically, we use $\mathbf{x}_i^t$ to represent input features for each location i on a specific date t, and we aim to predict the corresponding target variables $\mathbf{y}_i^t$. In aquatic systems, each location can be a specific depth layer in a lake, or a different segment in a river network. To fully capture the spatial dependencies amongst different locations, we also introduce the neighborhood $\mathcal{N}(i)$ and the adjacency matrix $\mathbf{A}$, where $\mathcal{N}(i)$ represents a set of locations that are spatial neighbors of the location i, and $\mathbf{A}_{ij}$ represents the adjacency level between each pair of locations i and j (more details in Section 5.1).

In real-world scientific applications, the observed labels $\mathbf{y}_i^t$ are often sparse due to the substantial manual labor required to collect the observation data. In this paper, we use "observations" or "observation data" to represent the observed $\mathbf{y}$ values, which can be considered the ground truth of the target variables. The sparsity of the observations makes it challenging to directly train an ML model. To address this issue, we leverage the simulated data generated by the physics-based model to guide the learning of ML model. The physics-based model takes the input features $\{\mathbf{x}_i^t\}$ and simulates target variables based on known physical theory and a set of physical parameters. For example, in the context of modeling lake systems, the physics-based model simulates water temperature based on energy conservation law, and also requires physical parameters such as lake geometry and water clarity, which directly affect the change of temperature in response to external energy fluxes. In this work, we use "simulations" or "simulated data" to represent the $\mathbf{y}$ values simulated by physics-based models. This work also considers a learning framework using an ensemble of multiple sets of simulated data. In particular, we are provided with K different sets of simulated data generated by using K different parameter values (that are commonly used by domain scientists) for a specific physical parameter (e.g., water clarity). We represent each set of simulated target variables as $\tilde{\mathbf{Y}}_k = \{\tilde{y}_{i,k}^t\}$ for each location i on each date t.

To avoid ambiguity, we use "physical parameters" in this paper to refer to parameters in the physics-based model and otherwise "parameters" refer to parameters in the ML model.

4 Method

This section presents our proposed SIMLR framework (Fig. 1). We will first describe the architecture of the spatial temporal network (STN). This model not only captures the spatial and temporal data dependencies, but also

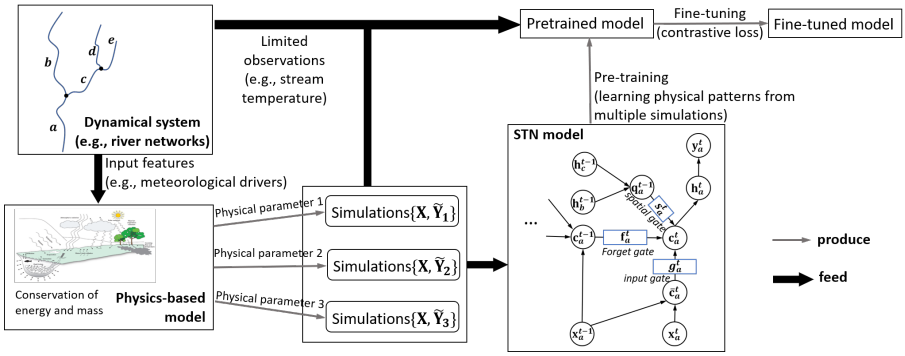


Fig. 1: The flow chart of the proposed framework. The thickened arrows represent data being fed to the next model. For example, we feed limited observations to the obtained pre-trained model and fine-tune it to the final model.

maintains separate network components for modeling general physical relationships and simulation-specific patterns. Then we will introduce a pre-training approach to initialize the STN model using multiple sets of simulated data. The pre-training approach aims to extract general physical knowledge from simulated data and leverage the separate components in the STN model to encode the extracted physical knowledge. The obtained initialized STN model needs to be fine-tuned to capture the gap between true observations and simulations. We will describe the fine-tuning process for the STN. This process also explores the relationships between observation data and simulated data, which brings two additional benefits. First, it provides additional interpretability by identifying the most suitable simulation setting for the target real system. Second, it helps better preserve the knowledge learned from the simulated data, especially for those simulations generated under the most suitable simulation setting.

4.1 Spatio-temporal model for scientific systems with different physical parameters

Physics-based models, e.g., General Lake Model [1] and PRMS-SNTemp [59], commonly use parameterized governing equations to represent physical processes that underlie the dynamical system. The physical parameters used in these models have physical definitions and often cannot be easily measured. These physical parameters determine how the model states change in response to external inputs. For example, given the same amount of solar radiation, a lake with higher water clarity will have a larger increase of water temperature at lower depths compared with a darker lake because more light can penetrate to the lower depths of the water column.

To represent these relationships and also to facilitate learning from multiple sets of simulated data, we build the STN model architecture. The STN model is essentially an extension of the LSTM structure. It uses a set of gating variables

to control the influence from different sources, including the inputs at the current time step, model states from the previous time step, and the effect from spatial neighbors.

Similar to the standard LSTM, the STN preserves a model state $\mathbf{c}_i^t$ for each location i at time t, which serves as a memory and will be updated over time (see Fig. 1). It also outputs a hidden representation $\mathbf{h}_i^t$ at every time step, which encodes the information about the location i and its spatial and temporal context. Now we describe the details of computing model states and hidden representation. First, we generate a candidate state $\bar{\mathbf{c}}_i^t$ by combining $\mathbf{x}_i^t$ and $\mathbf{h}_i^{t-1}$ using a $\tanh(\cdot)$ function, as follows:

$$\bar{\mathbf{c}}_i^t = \tanh(\mathbf{W}_c^h \mathbf{h}_i^{t-1} + \mathbf{W}_c^x \mathbf{x}_i^t + \mathbf{b}_c), \tag{1}$$

where $\{\mathbf{W}_c^h, \mathbf{W}_c^x, \mathbf{b}_c\}$ are model parameters.

For each location i, we generate hidden variables $\mathbf{q}_i^{t-1}$ by aggregating the hidden representation from its neighbors based on their adjacency level with the location i, as follows:

$$\mathbf{q}_i^{t-1} = \tanh(\mathbf{W}_q \sum_{j \in \mathcal{N}(i)} \mathbf{A}_{ji} \mathbf{h}_j^{t-1} + \mathbf{b}_q), \tag{2}$$

where $\{\mathbf{W}_q, \mathbf{b}_q\}$ are model parameters.

Then we generate three sets of gating variables: forget gating variables $\mathbf{f}_i^t$, input gating variables $\mathbf{g}_i^t$, and spatial gating variables $\mathbf{s}_i^t$. These gating variables are used to filter the information passed from the previous time step, the current time step, and the spatial neighborhood, respectively. Formally, these gating variables are computed using sigmoid function $\sigma(\cdot)$ as follows:

$$\begin{aligned} \mathbf{f}_i^t &= \sigma(\mathbf{W}_f^h \mathbf{h}_i^{t-1} + \mathbf{W}_f^x \mathbf{x}_i^t + \mathbf{b}_f), \\ \mathbf{g}_i^t &= \sigma(\mathbf{W}_g^h \mathbf{h}_i^{t-1} + \mathbf{W}_g^x \mathbf{x}_i^t + \mathbf{b}_g), \\ \mathbf{s}_i^t &= \sigma(\mathbf{W}_s^q \mathbf{q}_i^{t-1} + \mathbf{W}_s^x \mathbf{x}_i^t + \mathbf{b}_s), \end{aligned} \tag{3}$$

where $\Theta = \{\mathbf{W}_f^h, \mathbf{W}_f^x, \mathbf{W}_g^h, \mathbf{W}_g^x, \mathbf{W}_s^q, \mathbf{W}_s^x, , \mathbf{b}_f, \mathbf{b}_g, \mathbf{b}_s\}$ are model parameters.

Once we obtain the gating variables, we can use them to filter the information from the previous time ($\mathbf{c}_i^{t-1}$), the current time step ($\bar{\mathbf{c}}_i^t$), and the spatial neighborhood ($\mathbf{q}_i^{t-1}$) via element-wise product $\odot$, and combine the filtered information to compute the model state at time t. This can be expressed as follows:

$$\mathbf{c}_i^t = \mathbf{f}_i^t \odot \mathbf{c}_i^{t-1} + \mathbf{g}_i^t \odot \bar{\mathbf{c}}_i^t + \mathbf{s}_i^t \odot \mathbf{q}_i^{t-1}, \tag{4}$$

According to this equation, the change of model states given the inputs over space and time is conditioned on the gating variables $\mathbf{f}_i^t$, $\mathbf{g}_i^t$, and $\mathbf{s}_i^t$. This is analogous to the evolution of a dynamical system, which is conditioned on specific physical parameters. Hence, we can use these gating variables to encode variations in physical parameters. By varying parameters Θ, the STN model can represent the dynamical system using different physical parameters.

After obtaining the model state $\mathbf{c}_i^t$, we generate the output gating variables $\mathbf{o}_i^t$ and use them to filter the model state to compute the hidden representation $\mathbf{h}^t$, as follows:

$$\begin{aligned} \mathbf{o}_i^t &= \sigma(\mathbf{W}_o^h \mathbf{h}_i^{t-1} + \mathbf{W}_o^x \mathbf{x}_i^t + \mathbf{b}_o), \\ \mathbf{h}_i^t &= \mathbf{o}_i^t \odot \tanh(\mathbf{c}_i^t). \end{aligned} \tag{5}$$

Finally, we generate predicted target variables $\hat{\mathbf{y}}_i^t$ using a linear transformation, as follows:

$$\hat{\mathbf{y}}_i^t = \mathbf{W}_y \mathbf{h}_i^t + \mathbf{b}_y. \tag{6}$$

The loss function of STN is defined using true observations $\mathbf{Y} = \{\mathbf{y}_i^t\}$ that are available at certain time steps and certain locations, as follows:

$$\mathcal{L}_{\text{STN}}(\hat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{\mathbf{Y}} \sum_{\{(i,t)|\mathbf{y}_i^t \in \mathbf{Y}\}} (\mathbf{y}_i^t - \hat{\mathbf{y}}_i^t)^2. \tag{7}$$

The model has two sets of parameters. The model parameters $\Theta = \{\mathbf{W}_f^h, \mathbf{W}_f^x, \mathbf{W}_g^h, \mathbf{W}_g^x, \mathbf{W}_s^q, \mathbf{W}_s^x, \mathbf{b}_f, \mathbf{b}_g, \mathbf{b}_s\}$ can capture the difference in physical processes represented using different physical parameters and thus they are specific to each set of simulated data. Later in Section 4.2, we will create different copies of these parameters Θ for different sets of simulated data. We represent other parameters using $\Phi = \{\mathbf{W}_c^h, \mathbf{W}_c^x, \mathbf{W}_q, \mathbf{W}_o^h, \mathbf{W}_o^x, \mathbf{W}_y, \mathbf{b}_c, \mathbf{b}_q, \mathbf{b}_o, \mathbf{b}_y\}$. The parameters Φ are shared across different sets of simulated data.

4.2 Pre-training: Extract knowledge shared across simulations

Although different sets of simulated data are generated using different physical parameters (some are close to reality and some are different), the simulations still share some common patterns of general physical relationships embodied in the physics-based model. At the same time, each set of simulated data also shows patterns specific to its own parameter set. Here we introduce a pre-training strategy for the STN model, which aims to estimate the initial value of Φ_0 and Θ_0 by extracting the general physical relationships. The goal is that the initial value of Θ_0 can later be quickly adjusted to fit different simulation settings while keeping the Φ_0 parameters the same across different simulations.

Our method is inspired by the Model Agnostic Meta Learning (MAML) [60]. The original MAML is designed for learning a model that can be easily adapted to a new task using limited samples. Although the objective of MAML is different from our task, we use the similar method to construct to learning objective so that the pre-trained model can be easily adapted to each set of simulations after slight fine-tuning. In particular, we divide each set of simulated data $\{\mathbf{X}, \tilde{\mathbf{Y}}_k\}$ into a separate training set $\{\mathbf{X}^{\text{tr}}, \tilde{\mathbf{Y}}_k^{\text{tr}}\}$ and validation set $\{\mathbf{X}^{\text{val}}, \tilde{\mathbf{Y}}_k^{\text{val}}\}$. For the k^{th} simulation, we update Θ_0 using its training data $\{\mathbf{X}^{\text{tr}}, \tilde{\mathbf{Y}}_k^{\text{tr}}\}$ with a learning rate α while not changing the other parameters Φ_0, as follows:

$$\Theta_k = \Theta_0 - \alpha \nabla_\Theta \mathcal{L}_{\text{STN}}(f_{\text{STN}}(\mathbf{X}^{\text{tr}};\ \Theta_0, \Phi_0), \tilde{\mathbf{Y}}_k^{\text{tr}}), \tag{8}$$

where $f_{\text{STN}}(\cdot)$ represents the mapping relation from input features to target variables defined by the STN (Eqs. (1)-(6)), Θ_k represents the simulation-specific parameters for gating variables. Here Eq. (8) just shows the adjustment of Θ using a one-step gradient descent. This can be easily extended to multiple update steps, which allows more flexible adjustment of Θ to fit each simulation. In our implementation, we found the update with no more than five steps can already lead to good performance. More discussions on the selection of the number of update steps are in Section 5.6.

Once we gather the Θ_k that are specific to each set of simulated data, we define the pre-training loss using the k^{th} simulation's validation set, as follows:

$$\mathcal{L}_{\text{pre}} = \sum_k \mathcal{L}_{\text{STN}}(f_{\text{STN}}(\mathbf{X}^{\text{val}};\ \Theta_k, \Phi_0), \tilde{\mathbf{Y}}_k^{\text{val}})/K. \tag{9}$$

During the pre-training process, we minimize the loss $\mathcal{L}_{\text{pre}}$ with respect to the initial parameters Φ_0 and Θ_0. These estimated parameters are used to initialize the STN model, which will then be fine-tuned using true observations. We also collect the obtained intermediate parameters Θ_k values for k=1 to K, which encode the information specific to each set of simulated data. These simulation-specific parameters will also be used for model fine-tuning.

4.3 Fine-tuning with True Observations

After initializing the parameters Φ and Θ using the values Φ_0 and Θ_0, we refine these parameters using the available observed target variables. Moreover, we aim to further leverage the simulated data to enhance the fine-tuning phase. In the following, we will introduce two approaches, contrastive learning and attention-based ensemble learning.

4.3.1 Contrastive Learning

Our goal for using contrastive learning is to explore the relationships between the observation data and different sets of simulated data during the fine-tuning phase. Because each set of simulated data can be considered as an ideal version of real data under certain physical parameter settings, for each observed sample, it is possible to find its matched counterpart in the set of simulated data. Here we will introduce a new loss function for fine-tuning that captures this relationship.

In particular, we first generate the hidden representation for each location i at time t, as follows:

$$\mathbf{h}_i^t = g_{\text{STN}}(\mathbf{x}_i^{1:t};\ \Phi, \Theta), \tag{10}$$

where the function $g_{\text{STN}}(\cdot)$ represents the function defined by the STN model to extract hidden representation $\mathbf{h}_i^t$ from input data by following Eqs. (1)-(5).

Using the collected Θ_k for k^{th} simulation setting, we also generate the corresponding hidden representation $\tilde{\mathbf{h}}_{i,k}^t$. It is noteworthy that these hidden

representations are generated using gating variables that are specific to each set of simulated data. This process can be expressed as follows:

$$\tilde{\mathbf{h}}_{i,k}^{t} = g_{\text{STN}}(\mathbf{x}_i^{1:t};\ \Phi, \Theta_k). \tag{11}$$

Here the obtained $\tilde{\mathbf{h}}_{i,k}^{t}$ encodes the spatial and temporal patterns under the specific parameter settings used to generate k^{th} set of simulated data.

After gathering these hidden representations, we define a similarity mapping $\mathbf{h}_i^t \rightarrow \tilde{\mathbf{h}}_{i,k}^{t}$ for each k=1 to K using the inner product of these two vectors. Once we obtain the similarity values for all sets of simulated data (i.e., k=1 to K), we normalize the obtained similarity values and convert them into a distribution $\mathcal{Q}(\mathbf{h}_i^t \rightarrow \tilde{\mathbf{h}}_{i,k}^{t})$ via a softmax function. More formally, this can be expressed as follows:

$$\mathcal{Q}(\mathbf{h}_i^t \rightarrow \tilde{\mathbf{h}}_{i,k}^{t}) = \frac{\exp(\mathbf{h}_i^t \cdot \tilde{\mathbf{h}}_{i,k}^{t})}{\sum_{k'} \exp(\mathbf{h}_i^t \cdot \tilde{\mathbf{h}}_{i,k'}^{t})} \tag{12}$$

We aim to ensure that the patterns extracted from observation data are similar to certain sets of simulated data that use more accurate physical parameters, but are different from other simulation settings. Specifically, we define a contrastive loss based on the entropy of the similarity probability, as follows:

$$\mathcal{L}_{\text{ctr}} = -\sum_{i=1}^{N}\sum_{t=1}^{T}\sum_{k=1}^{K} \mathcal{Q}(\mathbf{h}_i^t \rightarrow \tilde{\mathbf{h}}_{i,k}^{t}) \log \mathcal{Q}(\mathbf{h}_i^t \rightarrow \tilde{\mathbf{h}}_{i,k}^{t})/NT. \tag{13}$$

As a side benefit, this method also enables the discovery of more accurate physical parameters for each location at each time step. Compared to standard physics-based model which commonly assumes static parameters in space and/or time, the proposed method has a better chance at capturing the variability of underlying physical processes.

Combining the contrastive loss and the standard supervised loss (Eq. (7)) using the observation data, we get the final fine-tuning loss, as follows:

$$\mathcal{L}_{\text{ft}} = \mathcal{L}_{\text{STN}} + \lambda \mathcal{L}_{\text{ctr}}, \tag{14}$$

where λ is a hyper-parameter.

4.3.2 Attention-based Ensemble Learning

The contrastive learning method enforces that the STN predictions are close to a small set of simulations and regularizes the model training though the similarity mapping over the hidden space. An alternative approach is to directly aggregate STN predictions with simulated data in the output space. In this aggregation process, each set of the simulations may contribute differently over

time due to the variability of underlying physical characteristics. To address the issue, we use the attention mechanism to dynamically adjust the contribution from each data source. The attention mechanism was designed to intelligently switch the focus of the ML model to important features or data portions [61, 62]. Our attention model weigh the STN outputs and simulation vectors differently by automatically learning their importance to the final prediction.

In particular, as we need to weigh both STN outputs and simulated data in the attention mechanism, we first create an augmented hidden representation $\mathbf{ha}_i^t$ for each location i by concatenating its STN representation and the simulation-specific hidden representation, as

$$\mathbf{ha}_i^t = [\mathbf{h}_i^t, \tilde{\mathbf{h}}_{i,1}^t, \tilde{\mathbf{h}}_{i,2}^t, ..., \tilde{\mathbf{h}}_{i,K}^t]. \tag{15}$$

Then we use the augmented representation to generate the attention weights in a similar way with the Eq. 12 in building the similarity mapping, as

$$\alpha_{i,k}^t = \frac{\exp(\mathbf{h}_i^t \cdot \mathbf{ha}_{i,k}^t)}{\sum_{k'=0}^{K} \exp(\mathbf{h}_i^t \cdot \mathbf{ha}_{i,k'}^t)}, \tag{16}$$

where $\mathbf{ha}_{i,0}^t = \mathbf{h}_i^t$, and $\mathbf{ha}_{i,k}^t = \tilde{\mathbf{h}}_{i,k}^t$ for $k = 1, 2, ..., K$. Also note that the Eq. (16) uses a softmax function, and thus the obtained attention weights $\{\alpha_{i,k}^t\}_{k=1}^K$ sum up to 1 for each location i at each time t.

After obtaining the attention weights, we combine the STN outputs $\mathbf{y}_i^t$(Eq. (6)) and simulated target variables in an ensemble way using attention weights. This aggregation process can be expressed as follows:

$$\bar{\mathbf{y}}_i^t = \alpha_{i,0}^t \mathbf{y}_i^t + \sum_{k=1}^{K} \alpha_{i,k}^t \tilde{\mathbf{y}}_{i,k}^t. \tag{17}$$

This model will be updated by minimizing the supervised loss between the obtained aggregated outputs $\{\bar{\mathbf{y}}_i^t\}$ and observed data using the supervised loss function (Eq. (7)).

In summary, there are two training phases for the STN model, the pre-training phase and the fine-tuning phase. The pre-training phase uses only the simulated data to initialize the STN model. The fine-tuning phase further adjusts the STN model parameters using observation data. At the same time, the fine-tuning process leverages the simulation-specific parameters learned from the pre-training phase to explore the relationship between observations and simulations.

5 Experiments

In this section, we first introduce two datasets that are used in our tests. Then we evaluate our proposed method to answer the following questions:

- **Q1:** Can the proposed method outperform existing methods in predicting target variables?
- **Q2:** How will the predictive performance change under data-sparse scenarios?
- **Q3:** Can the model better generalize to out-of-distribution scenario?
- **Q4:** What is the effect of pre-training in the prediction?
- **Q5:** Can the proposed method discover the physical parameters that are most suitable for our study region over long time periods?
- **Q6:** How sensitive is the performance with respect to hyper-parameters in the proposed method?

5.1 Datasets

We apply our SIMLR model to two different environmental modeling challenges, predicting depth-specific water temperatures in a lake and predicting water temperatures for segments in a stream network. Both problems require accurately accounting for variations across space (e.g. lake depth, stream reaches) and time (e.g. daily weather patterns, seasonal climate). Note that these problems differ substantially in the nature of the spatial relationships, and they need to be modeled by very different physics-based models. In particular, lakes exchange heat mostly at the water surface. At the same time, there are various processes operating at different timescales act to distribute heat through the lake vertically. In contrast, because streams are well-mixed, the entire stream warms or cools primarily due to energy exchange with the atmosphere and other water sources (e.g., groundwater). In-stream heat almost exclusively flows downhill along the river network and at the same pace as the water. The focal parameters are different in each problem and have very different effects: lake geometry affects the degree, timing, and duration of thermal stratification and affects the response of the lake to wind events, and lake water clarity controls the depths at which incoming solar energy is absorbed, while the groundwater residence time in stream reaches controls the temperature of incoming groundwater.

D1: Predicting water temperature in Lake Mendota. This dataset was collected from Lake Mendota in Wisconsin, USA [24]. This lake system is reasonably large ($\sim$40 km^2 in area) and the lake has a maximum depth of 25 meters. It also exhibits large changes in water temperatures in response to seasonal and sub-seasonal weather patterns and thermally stratifies during the summertime. Observations of lake temperature were collected from North Temperate Lakes Long-Term Ecological Research Program [24].

The input features that describe prevailing meteorological conditions are available on a continuous daily basis from April 02, 1980, to December 30, 2014 (12,690 dates). We used a set of seven input features, including shortwave and long-wave radiation, air temperature, relative humidity, wind speed, frozen and snowing indicators. These were acquired and/or computed from the North American Land Data Assimilation System. Temperature observations vary in their distribution across depths and time, i.e., there are certain days

when observations are available only on a few depths or no observations are available.

We use the observed data from April 02, 1980, to October 31, 1991, and the data from June 01, 2003, to December 30, 2014, as training data (in total 8,037 observations). Then we applied the trained model to predict the temperature at different depths for the period from November 01, 1991, to May 31, 2003 (in total 5,121 observations).

D2: Predicting water temperature in Delaware River Basin. The dataset is pulled from U.S. Geological Survey's National Water Information System [63] and the Water Quality Portal [64]. The river segments were defined by the network used for the National Hydrologic Model [65], and the river segments are split up to have roughly a one day water travel time. We study a subset of the Delaware River Basin with 42 river segments that feed into the mainstream Delaware River at Wilmington, Delaware.

We use input features at the daily scale from October 01, 1980, to September 30, 2016 (13,149 dates). The input features have 10 dimensions which include precipitation, air temperature, day of year, solar radiation, shade fraction, potential evapotranspiration, and the geometric features of each segment (e.g., elevation, length, slope, and width). Air temperature, precipitation, and solar radiation values were derived from the gridMET dataset [66]. For both datasets D1 and D2, the daily scale meteorological input are calculated as the daily average of the original hourly meteorological data from the gridMET dataset. Other input features (e.g., shade fraction, potential evapotranspiration) are difficult to measure frequently, and we use values produced by the PRMS-SNTemp model as its internal variables. Water temperature observations were available for 32 segments but the temperature was observed only on certain dates. The number of temperature observations available for each observed segment ranged from 1 to 9,810 with a total of 51,103 observations across all dates and segments. We use the observed data from October 01, 1980, to September 30, 1992, and the data from October 01, 2004, to September 30, 2016, as training data (in total 34,985 observations). Then we applied the trained model to predict the temperature for the period from October 01, 1992, to September 30, 2004 (in total 16,118 observations).

Simulated data: In D1, we use the physics-based General Lake Model (GLM) [1] to generate different simulated data by varying the lake geometry and water clarity. Specifically, we used "cone," "barrel," and "martini" shapes to define the depth-area parameters in the GLM to generate three sets of simulated data. Then we fix the geometry as "cone" and use three different clarity levels "normal" (Kw=0.45), "dark" (Kw=1.20), and "clear" (Kw=0.25). Water clarity affects the penetration of solar radiation into the deeper water. In D2, we use PRMS-SNTemp [59] to generate different simulations by setting average groundwater residence time (τ) as 10 days, 45 days, and 100 days. A shorter τ means the groundwater quickly moves through the groundwater aquifer and enters the stream at a temperature more similar to recent air temperatures, whereas a longer τ means groundwater temperatures

Table 1: Performance (as measured by root mean squared error (RMSE) in degrees Celsius) using simulations with different geometric parameters in modeling lake water temperature (D1). The first three rows represent the simulated data produced by the physics-based model using different parameters. The methods RNN, STN, and STN^{DA} (rows 4-6) do not use any simulations. The superscript $p(k)$ means that the model is first pre-trained using simulations generated using parameter k. Here % columns are percent observations used during fine-tuning phase. The +/- values represents the range of values across replicates with random starting weights, and NA's for the +/- values are for models that did not have multiple model runs. The bolded values are the best performing models in each dataset and data sparsity level.

Method	**0%**	**0.2%**	**2%**	**100%**
$\text{GLM}^{(\text{cone})}$	2.664(±NA)	-	-	-
$\text{GLM}^{(\text{barrel})}$	3.791(±NA)	-	-	-
$\text{GLM}^{(\text{martini})}$	5.919(±NA)	-	-	-
RNN	-	4.615(±0.173)	2.311(±0.240)	1.489(±0.091)
STN	-	3.349(±0.381)	1.848(±0.200)	1.393(±0.070)
STN^{DA}	-	3.867(±0.390)	1.862(±0.189)	1.394(±0.073)
$\text{STN}^{\text{comb(cone)}}$	-	2.272(±0.256)	1.703(±0.113)	1.381(±0.073)
$\text{PGRNN}^{p(\text{cone})}$	2.469(±0.168)	**2.056**(±0.184)	1.595(±0.097)	1.374(±0.074)
$\text{STN}^{p(\text{cone})}$	**2.289**(±0.175)	2.181 (±0.173)	1.591(±0.107)	1.368(±0.075)
$\text{STN}^{p(\text{barrel})}$	2.996(±0.102)	2.808(±0.187)	1.642(±0.102)	1.312(±0.075)
$\text{STN}^{p(\text{martini})}$	5.386(±0.124)	2.955(±0.074)	1.821(±0.071)	1.402(±0.081)
$\text{STN}^{\text{SIMLR}}$	2.914(±0.116)	2.103(±0.076)	1.634(±0.144)	1.373(±0.045)
$\text{STN}^{\text{SIMLR-att}}$	2.914(±0.116)	2.083(±0.183)	1.636(±0.135)	1.372(±0.058)
$\text{STN}^{\text{SIMLR-ctr}}$	2.914(±0.116)	2.431(±0.196)	**1.535**(±0.132)	**1.248**(±0.061)

are more seasonally stable. These physical parameter values represent a range of values observed across lake and stream systems, but are not tailored to our target systems.

Goals: In D1, we aim to predict water temperature in each depth of the water column. In D2, we aim to predict water temperature in each river segment. Here we assume the river temperature is the same across depth because rivers tend to be well-mixed and shallower.

Implementation details: We implement STN using Tensorflow and GTX 2080 GPU. All the hidden variables and gating variables have 20 dimensions. We use five update steps for obtaining simulation-specific parameters Θ_k during each epoch of pre-training. The model is pre-trained for 150 epochs with learning rate 0.001 before being fine-tuned for 100 epochs with learning 0.0005. The hyper-parameter λ is set as 0.5.

We generate the adjacency matrix $\mathbf{A}$ based on the inverse relation of the distance between each pair of locations i and j. We represent the distance as $\text{dist}(i, j)$. In D1, we use the distance between different layers across depth. In D2, this represents the stream distance between the endpoints of each pair of river segments and we only consider i as a neighbor of j when i is anywhere upstream from j (so j is affected by water flow from i). We standardize the distance and then compute the adjacency level as $\mathbf{A}_{ij} = 1/(1 + \exp(\text{dist}(i, j)))$ for each pair of locations (i, j).

Table 2: Performance using simulations with different clarity parameters in modeling lake water temperature (D1). The bolded values are the best performing models in each dataset and data sparsity level.

Method	0%	0.2%	2%	100%
$GLM^{(normal)}$	2.664(±NA)	-	-	-
$GLM^{(dark)}$	3.053(±NA)	-	-	-
$GLM^{(clear)}$	1.723(±NA)	-	-	-
RNN	-	4.615(±0.173)	2.311(±0.240)	1.489(±0.091)
STN	-	3.349(±0.381)	1.848(±0.200)	1.393(±0.070)
STN^{DA}	-	3.867(±0.390)	1.862(±0.189)	1.394(±0.073)
$STN^{comb(clear)}$	-	2.141(±0.181)	1.692(±0.132)	1.380(±0.075)
$PGRNN^{p(clear)}$	2.518(±0.135)	2.050(±0.120)	1.648(±0.128)	1.371(±0.076)
$STN^{p(normal)}$	2.289(±0.175)	2.179 (±0.206)	1.594(±0.100)	1.377(±0.074)
$STN^{p(dark)}$	2.582(±0.164)	2.084(±0.195)	1.634(±0.099)	1.326(±0.031)
$STN^{p(clear)}$	**2.214**(±0.133)	1.847(±0.205)	1.645(±0.116)	1.308(±0.056)
STN^{SIMLR}	2.425(±0.044)	1.817(±0.049)	1.601(±0.035)	1.372(±0.034)
$STN^{SIMLR\text{-}att}$	2.425(±0.044)	1.819(±0.042)	1.525(±0.031)	1.366(±0.031)
$STN^{SIMLR\text{-}ctr}$	2.425(±0.044)	**1.806**(±0.036)	**1.503**(±0.029)	**1.263**(±0.031)

Table 3: Performance using simulations with different parameters of average groundwater residence time (τ) in modeling river water temperature (D2). The bolded values are the best performing models in each dataset and data sparsity level.

Method	0%	0.2%	2%	100%
$PRMS\text{-}SNTemp^{(\tau 10)}$	**2.618**(±NA)	-	-	-
$PRMS\text{-}SNTemp^{(\tau 45)}$	3.558(±NA)	-	-	-
$PRMS\text{-}SNTemp^{(\tau 100)}$	5.840(±NA)	-	-	-
RNN	-	2.867(±0.147)	1.732(±0.083)	1.445(±0.027)
STN	-	2.356(±0.135)	1.858(±0.105)	1.397(±0.030)
STN^{DA}	-	2.624(±0.138)	1.863(±0.103)	1.420(±0.032)
$STN^{comb(\tau 10)}$	-	2.396(±0.109)	1.716(±0.094)	1.439(±0.075)
$PGRGrN^{p(\tau 10)}$	2.852(±0.103)	2.362(±0.098)	1.628(±0.063)	1.396(±0.033)
$STN^{p(\tau 10)}$	2.738(±0.094)	2.259(±0.123)	1.697(±0.096)	1.403(±0.022)
$STN^{p(\tau 45)}$	3.632(±0.084)	2.409(±0.124)	1.874(±0.079)	1.473(±0.029)
$STN^{p(\tau 100)}$	5.596(±0.079)	2.480(±0.089)	1.871(±0.092)	1.457(±0.027)
STN^{SIMLR}	3.235(±0.045)	**2.009**(±0.130)	1.636(±0.066)	1.403(±0.014)
$STN^{SIMLR\text{-}att}$	3.235(±0.045)	2.054(±0.090)	1.675(±0.057)	1.468(±0.019)
$STN^{SIMLR\text{-}ctr}$	3.235(±0.045)	2.103(±0.079)	**1.618**(±0.058)	**1.362**(±0.021)

5.2 Evaluation of predictive performance

We compare our method against multiple baselines, i.e., the physics-based model (GLM in D1 and PRMS-SNTemp in D2); Recurrent Neural Network (RNN) with the LSTM cell; and the state-of-the-art methods, PGRNN [21] and PGRGrN [46], which have shown success in modeling lake temperature and river temperature, respectively. We also compare different versions of the STN model. First, we compare to the STN model trained using only observed labels (STN). We also implement a semi-supervised version of the STN model via adversarial domain adaptation (STN^{DA}) [67], which considers the training data as the source domain and the testing data as the target domain. Next,

we consider the STN models that are pre-trained using a specific set of simulated data (as proxy labels) and then fine-tuned with observed labels ($\text{STN}^{p(c)}$, where c is a specific parameter setting). Such a comparison aims to show the advantage of our proposed pre-training strategy in extracting general physical patterns jointly from multiple sets of simulations. To show the effectiveness of pre-training, we also implement a baseline $\text{STN}^{\text{comb(c)}}$, which directly combines the simulations and observations as training labels during the training process. Specifically, in the training period, this baseline uses observed labels when they are available and uses simulated labels otherwise. Since simulated labels under different parameter settings are different with each other, we only report the simulation setting c that produces the best performance in each test. Additionally, we implement three versions of our proposed method $\text{STN}^{\text{SIMLR}}$, $\text{STN}^{\text{SIMLR-att}}$, and $\text{STN}^{\text{SIMLR-ctr}}$. They have the same pre-training process but different fine-tuning processes. The $\text{STN}^{\text{SIMLR}}$ method only uses the supervised loss on the outputs of the STN model (Eq. (7)) in fine-tuning. $\text{STN}^{\text{SIMLR-att}}$ also uses the supervised loss but it combines the STN outputs with simulation data through the ensemble method (discussed in Section 4.3.2) to generate final predictions. $\text{STN}^{\text{SIMLR-ctr}}$ follows the contrastive learning method (discussed in Section 4.3.1) and optimizes the the contrastive loss (Eq. (14)).

In Tables 1 and 2, we report the performance of different methods in predicting water temperature in lake systems using different parameters of lake geometry and clarity, respectively. In Table 3, we report the performance in predicting water temperature in river networks using different parameters for average residence time in groundwater flow (i.e., τ). For methods PGRNN (for lake temperature prediction) and PGRGrN (for river temperature prediction), because they can only learn from one set of simulated data, we show the performance of these methods using the simulated data that produce the best performance ("cone" for lake geometry, "clear" for lake clarity, and "τ=10 days" in river modeling).

We can observe that our method outperforms other methods by a considerable margin in both applications (**Q1**). The improvement from RNN to STN shows the effectiveness of incorporating spatial dependencies in modeling thermodynamic patterns. The domain adaptation method (STN^{DA}) does not improve the performance because the invariance structure of input features cannot help extract underlying effect on weather conditions to water temperature change. The physics-based models (i.e., GLM and PRMS-SNTemp) perform poorly because of their inherent model bias due to approximations and imperfect parameterizations. Nevertheless, the proposed pre-training method (i.e., $\text{STN}^{\text{SIMLR}}$) can still extract useful physical knowledge from the imperfect simulated data and thus performs better than STN, especially when we are using less training data. The performance is further improved after we use the contrastive loss in fine-tuning (i.e., $\text{STN}^{\text{SIMLR-ctr}}$). This is because $\text{STN}^{\text{SIMLR-ctr}}$ can better learn from specific sets of simulated data that are closer to the reality. In general, $\text{STN}^{\text{SIMLR-ctr}}$ performs better than $\text{STN}^{\text{SIMLR-att}}$. In the contrastive learning process, the simulated data have

been used to guide the training of STN training by exploring similarity amongst STN and each physics-based model. In contrast, the attention-based ensemble model directly combines STN and physics-based models in the output layer. Hence, the attention-based ensemble prediction can be affected by the bias in simulated data while the contrastive learning method only enforces the similarity through the hidden representation, which encodes representative patterns of water dynamics.

As we reduce the amount of training data, all of the methods produce larger prediction error. However, we can clearly see that the models that are pre-trained using simulated data (i.e., methods in the second and third blocks of each dataset) have much lower error than non-pre-trained models when we use fewer training samples. The pre-trained model also performs better than the STN^{comb} method using the simulations under the same setting. This is because the simulated labels can be biased due to approximations used in physics-based models and thus directly using them as supervision can degrade the performance. In contrast, the pre-training-based training method uses simulations and observations in two separate stages. Using simulated labels can help initialize intermediate network layers in the STN model, and the bias can be mitigated when the model is fine-tuned with observed labels. These models can learn a better initialized state from a large amount of simulated data (available at every day and every location) and thus require less observation data for fine-tuning. Also, our proposed method generally performs better than existing methods with the pre-training process (i.e., PGRNN and PGRGrN) given limited observation data (e.g., 0% and 0.2%) (**Q2**). The pre-training strategy used in STN jointly learns from multiple sets of simulated data and updates different components of the model (e.g., gating variables and other layers) in a deliberate way to reflect the difference in physical parameters. Hence, compared to PGRNN and PGRGrN, STN has a better chance at capturing general physical knowledge while reducing the effect of imperfect physical settings.

We can also observe that models pre-trained using different sets of simulated data can have very different performance. Specifically, the martini shape is very different from the true shape of Lake Mendota so the model pre-trained with the martini simulations has relatively poor performance. Similarly, the river temperature model $\text{STN}^{p(\tau 100)}$ has worse performance because the residence time for shallow groundwater is thought to be generally less than 100 days for many segments in the Delaware River Basin especially during higher stream flows (Martin Briggs, U.S. Geological Survey, written commun., Feb. 8, 2021). However, these pre-trained models ($\text{STN}^{p(\text{martini})}$ and $\text{STN}^{p(\tau 100)}$) can get much better performance when refined using even a small amount of data (e.g., 2%), and predictions can still be much better than the STN model without pre-training. The methods PGRNN and PGRGrN show similar results since they are also pre-trained using simulations.

Moreover, we can observe that some pre-trained models (e.g., $\text{STN}^{p(\text{cone})}$, $\text{STN}^{p(\text{clear})}$, and $\text{STN}^{p(\tau 10)}$) have better performance than our proposed methods ($\text{STN}^{\text{SIMLR}}$, $\text{STN}^{\text{SIMLR-att}}$, $\text{STN}^{\text{SIMLR-ctr}}$) before fine-tuning (i.e., with 0%

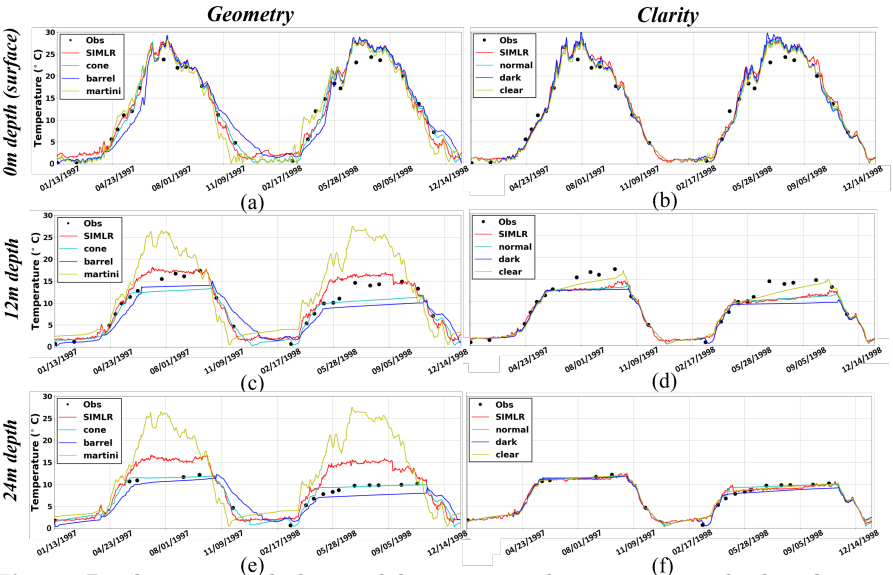


Fig. 2: Predictions made by models pre-trained using our method and using each set of simulations with different parameters for geometry (first column) and clarity (second column). All the predictions are shown at 0m depth, 12m depth, and 24m depth (Rows 1-3).

data for fine-tuning). However, in practice we may not know the most suitable parameters when training the model. We can see that our method can still get comparable performance even without access to such information. Moreover, our method ($\text{STN}^{\text{SIMLR-ctr}}$) after fine-tuning has better performance than all the baselines by using the contrastive loss to explore the relationship between observations and different sets of simulated data.

5.3 Predictions of pre-trained models

Here we discuss the effect of pre-training in different scenarios (**Q4**). In Fig. 2, we show the predictions made by the pre-trained model using our method (SIMLR) and using each set of simulated data over different depths of Lake Mendota. Pre-trained models are always biased because they are trained from simulations with imperfect parameterization. We can observe that SIMLR predictions are generally in the middle of predictions made by other pre-trained models and also follow the similar temporal patterns. This is because SIMLR extracts general patterns that are shared by all these different sets of simulated data. Besides, SIMLR is able to achieve reasonable accuracy compared with observations even without the awareness of the best parameter setting.

In Fig. 3, we show the predictions made by the pre-trained model using our proposed method and the fine-tuned model using 2% data (using $\text{STN}^{\text{SIMLR-ctr}}$). Although the pre-trained model has bias compared to true observations, it is able to capture many general physical relationships (e.g.,

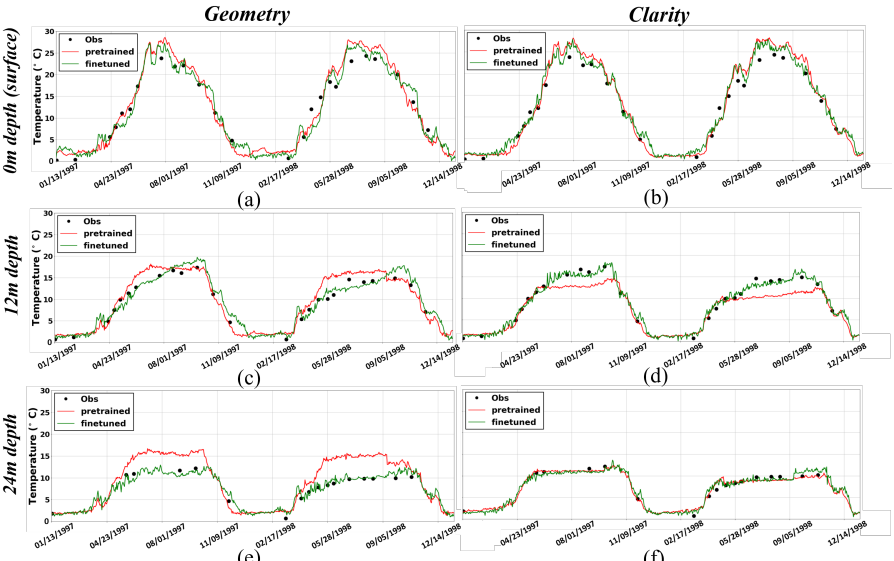


Fig. 3: Predictions made by both pre-trained and fine-tuned (using 2% observations) models using simulations with different parameters for geometry (first column) and clarity (second column). All the predictions are shown at 0m depth, 12m depth, and 24m depth (Columns 1-3).

seasonal patterns, temperature variation across depths), and thus it can be easily refined to match observations even using just 2% data.

5.4 Variation of physical parameters

A goal of this work is to better understand the relation between observations and simulations through the similarity learned from the fine-tuning process (i.e., Eq. (12)). The results in this part are produced using $\text{STN}^{\text{SIMLR-ctr}}$ and $\text{STN}^{\text{SIMLR-att}}$. In Fig. 4 (a) and (c), we show the similarity with different clarity values in different lake depths (averaged over time). In Fig. 4 (b) and (d), we show the similarity with different geometries (cone, barrel, and martini) in different months (averaged over depth). We can see that the model is closer to clear or normal simulations in shallower depths but closer to dark simulations in lower depths. As none of the physics-based models provided accurate predictions for all depths, SIMLR revealed unaccounted-for or poorly parameterized processes that could be addressed given these insights, such as introducing a clarity parameter that varies with depth (as is common in the natural environment) or modifying vertical mixing parameters that could alter bottom water warming rates. We can also see that the model is closer to cone simulations in the summer and closer to barrel simulations in the fall and winter, which indicates the fall cooling period and under ice temperatures were better simulated when cooling was slowed by using the barrel lake shape. Moreover, it

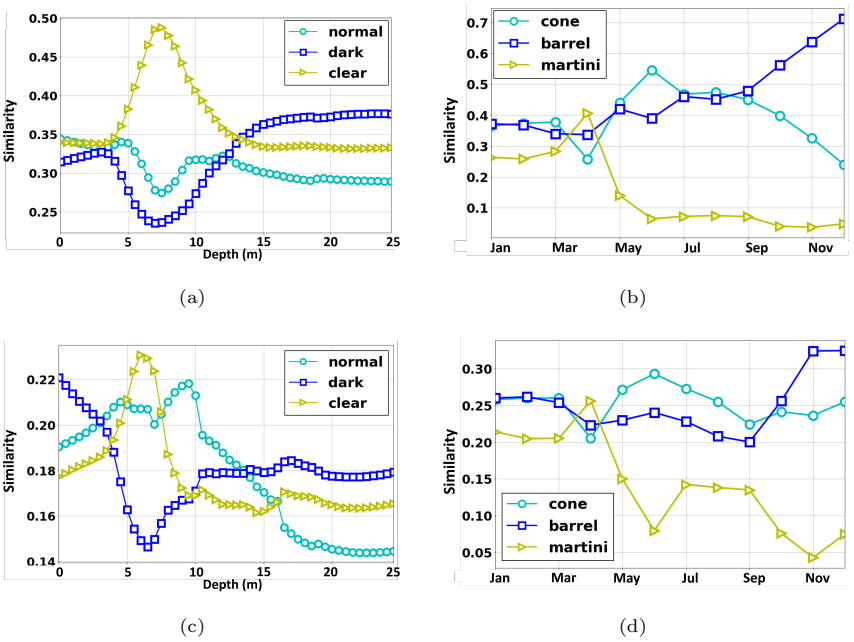


Fig. 4: (a)(b) The similarity mapping obtained from $\text{STN}^{\text{STN-ctr}}$, and (c)(d) the attention weights obtained from $\text{STN}^{\text{STN-att}}$ for each parameter setting. (a)(c) The similarity/weight to different clarity settings in different depths of the Lake Mendota. (b)(d) The similarity to different geometry settings in different months.

can be seen that the similarity relation learned by $\text{STN}^{\text{SIMLR-ctr}}$ is generally consistent to the attention weights learned by $\text{STN}^{\text{SIMLR-att}}$.

For the river modeling, SIMLR detects that most segments are more similar to simulation with groundwater residence time τ=10 days during March-May. A lower τ value indicates the groundwater temperature is more similar to recent air temperatures. Although PRMS-SNTemp encodes a constant value of τ throughout the year, seasonality in groundwater residence times has been confirmed in nearby watersheds, with shallow groundwater (having lower τ) contributing more in the spring than in other seasons [68]. These results show that the SIMLR approach transforms a constant physics-based model parameter into a flexibly time-varying parameter that is more consistent with observed temperatures and known processes (**Q5**).

5.5 Generalization test

We expect that our method has better generalizability to different scenarios given its ability to extract and transfer general physical relationships. Generalizability is important for scientific problems because most observation data may be collected from certain periods or locations for which it is easier to

Table 4: Temperature root mean squared error (RMSE) in the summer seasons of the testing period using models trained from spring, fall, and winter in the training period (the first column) and trained using all the data in the training period (the second column).

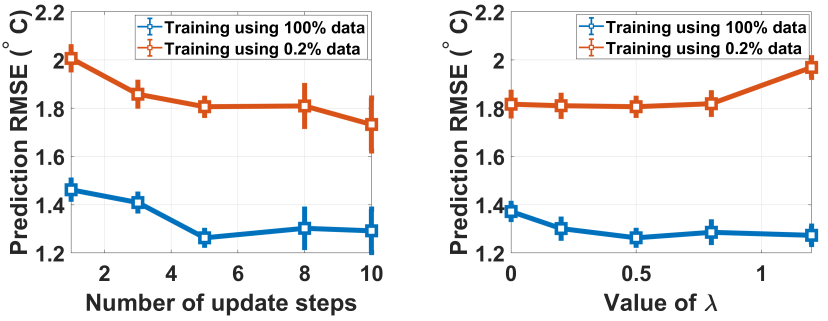
Method	Train on cold seasons	Train on all the data
GLM^{clear}	2.037(±NA)	2.037(±NA)
RNN	2.587(±0.245)	1.500(±0.035)
STN	2.180(±0.092)	1.389(±0.045)
STN^{DA}	2.932(±0.087)	1.497(±0.051)
STN^{SIMLR}	1.724(±0.061)	1.402(±0.037)
$STN^{SIMLR\text{-}att}$	1.821(±0.057)	1.398(±0.034)
$STN^{SIMLR\text{-}ctr}$	**1.685**(±0.066)	**1.325**(±0.034)

deploy sensors. As a strong test of generalizability for modeling lake temperature, we train the model using observations from colder seasons in the training period and then test in the summer time of the testing period. Although real-world temperate data collection procedures more often provide data in summer than in winter, training only on cold seasons is more challenging because Lake Mendota has highly dynamic patterns in summer and also a unique stratification across different layers due to the temperature difference between the surface and the lake bottom.

We test our method on the D1 dataset using three sets of simulated data with different clarity values. In Table 4, we report the performance in the summer seasons of the test period. We also include the testing performance of the model trained using all observations from the training period as a baseline in the second column of Table 4. We can see that all the methods have larger errors when they are trained only on colder seasons. However, our proposed method still yields better performance than other methods (**Q3**). This is because our method learns the general physical relationships that hold in different scenarios. We can see that the GLM model (under the clear clarity setting) performs better than pure data-driven models such as RNN and STN. The domain adaptation method (STN^{DA}) has a worse performance because the predictive model cannot fully distinguish different conditions in warmer and colder seasons by referring to an invariant feature space. Our proposed method performs better than GLM^{clear} in this generalization test because SIMLR is able to intelligently transfer knowledge from multiple sets of simulated data. Also, compared to the GLM simulation, the fine-tuning process using true observations from a different season is also helpful if it is applied to a good initial model, e.g., the model learned through SIMLR that embodies general physical relationships.

5.6 Sensitivity test

Here we test the performance using different settings of hyper-parameters (**Q6**). First, we test different number of update steps (iteration of Eq. (8))



(a) (b)

Fig. 5: The prediction root mean squared error (RMSE) using (a) different number of update steps and (b) the value of λ, for the STN$^{\text{SIMLR-ctr}}$ method.

and report the predictive performance of lake temperature modeling using simulations with clarity settings. Specifically, we show the performance using 100% data and 0.2% observation data in Fig. 5 (a). We can see that the model has similar performance when we set the number of update steps to be greater or equal to five. Although more update steps can slightly improve the performance, it will bring additional computational cost to the training process.

We also study the effect of hyper-parameter λ (the weight for the contrastive loss) on the performance of STN$^{\text{SIMLR-ctr}}$ (see Fig. 5 (b)). As we increase the value from 0, the prediction error decreases gradually. Such decrease is especially obvious when we use 100% data because the contrastive loss can better explore the relationship between observation and simulated data. In particular, if we set a very high value for λ, then the prediction error becomes larger when we use limited data (i.e., 0.2% data). This is because the small data may better fit certain simulations that are not close to reality and thus mislead the training process.

6 Conclusion

In this paper, we propose a new method for modeling spatial and temporal patterns in dynamical systems while also accommodating uncertainties in physics-based model parameters. We extract the general physical relationships over space and time to inform the initialization of the ML model. Then we fine-tune the model by exploring the similarity between available observation data and simulated data. We have demonstrated the superiority of the proposed method, which is better equipped to learn from limited observation data, provide insights about the value of physical parameters, and better generalize to unseen scenarios.

The proposed method can be widely applied to other dynamical systems that are commonly simulated by physics-based models with uncertain

parameters. For example, physics-based models for hydrology and climate systems often have bias, which stems from the uncertainty in selecting physical parameters. Another example is in the agriculture domain, in which many physics-based models have been developed to simulate the carbon cycle and model the growth of crops. Such modeling process also highly depends on soil properties, vegetation, and surrounding land covers, which are often uncertain over large regions. Our proposed method can bring a great potential to these applications by jointly learning from multiple sets of uncertain simulations.

While our method has shown the improved predictive performance by considering the variations on certain physical parameters, we could certainly explore a larger number of variations in future studies. Moreover, one could explore ways to learn from variations of multiple physical parameters at the same time. Another extension is to investigate the pre-training process using simulations that are created by different physics-based models. This could be very helpful for many scientific problems where multiple physics-based models have been developed with different modeling components and parameterizations. Finally, we anticipate the knowledge discovered by our method can advance the design of both physics-based models and machine learning models.

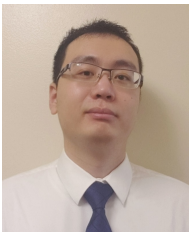
7 Acknowledgement

This work was supported by the USGS awards G21AC10207, G21AC10564, and G22AC00266, the NSF awards 2147195, 2105133, and 2126474, the NASA award 80NSSC22K1164, Google's AI for Social Good Impact Scholars program, the DRI award at the University of Maryland. This research was supported in part by the University of Pittsburgh Center for Research Computing through the resources provided. Any use of trade, firm, or product names is for descriptive purposes only and does not imply endorsement by the U.S. Government.

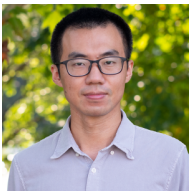
Author Biographies

Shengyu Chen is a Ph.D. student in Computer Science at the University of Pittsburgh. His current research directions include machine learning, data mining, physics-guided machine learning and their applications in environmental applications and computational fluid dynamics.

Nasrin Kalanat is a Ph.D. student in Computer Science at the University of Pittsburgh. Her current research is focused on building computer vision and transfer learning algorithms for modeling environmental systems.



Yiqun Xie received his Ph.D. degree in Computer Science from the University of Minnesota. He is currently an Assistant Professor in Geospatial Information Science at the University of Maryland, College Park. His research focuses on data mining and artificial intelligence methods for spatial data, such as satellite remote sensing data, UAV imagery, trajectories, etc. His recent research results have received the Best Paper Award from IEEE ICDM 2021, the Best Vision Paper Award from SIGSPATIAL 2019, and the Best Paper Award from SSTD 2019. His work was also highlighted by the Great Innovative Ideas program at the Computing Community Consortium at CRA.



Sheng Li received the B.Eng. degree in computer science and engineering and the M.Eng. degree from Nanjing University of Posts and Telecommunications, China, and the Ph.D. degree from Northeastern University, Boston, MA, in 2010, 2012 and 2017, respectively. He is a Tenure-Track Assistant Professor at the School of Data Science, University of Virginia since 2022. He has published over 140 papers at peer-reviewed conferences and journals, and has received over 10 research awards, such as the INNS Aharon Katzir Young Investigator Award, Fred C. Davidson Early Career Scholar Award, Adobe Data Science Research Award, Cisco Faculty Research Award, and SDM Best Paper Award. He serves as an Associate Editor of IEEE Transactions on Neural Networks and Learning Systems, IEEE Transactions on Circuits and Systems for Video Technology, and IEEE Computational Intelligence Magazine. His research interests include trustworthy representation learning, computer vision, and causal inference.

Jacob Zwart, Ph.D., works as a data scientist at the U.S. Geological Survey. He uses his expertise in computational modeling, data assimilation, and limnology to help produce short-term forecasts of water quality at regional scales to aid in water resources decision making. Jacob's research themes are: 1) improve understanding of aquatic biogeochemical processes and predicting how these processes may respond to future global change, 2) develop techniques to inject scientific knowledge into machine learning models to make accurate predictions of environmental variables (also known as "knowledge-guided machine learning"), and 3) advance methods for assimilating real-time observations into knowledge-guided machine learning models to improve near-term forecasts of water quality.

Jeffrey Sadler grew up in Utah and completed his BS and MS degrees at Brigham Young University in Provo, Utah. He graduated with his PhD from the University of Virginia in Charlottesville, Virginia. After his PhD, Dr. Sadler spent 3 years as a Postdoctoral Fellow with the US Geological Survey. Dr. Sadler is currently an assistant professor and extension specialist at Oklahoma St. University. His studies have focused on the data side of water – how to best leverage data through web-based platforms and machine learning models for better water outcomes. Dr. Sadler spent 2 years in Italy and as a hobby loves to bake sourdough bread to pizzas at home.

Alison Appling, Ph.D., is a data scientist and ecologist with the U.S. Geological Survey. She uses a combination of statistical, theoretical, and knowledge-guided machine learning methods to predict and understand water quality dynamics in rivers and lakes. She studied floodplain carbon and nitrogen cycling for her PhD at Duke University and has since worked on topics including sub-daily nutrient uptake dynamics in aquatic systems, statistical estimation of riverine nutrient loads and metabolic rates, and machine-learning-based prediction of lake and stream water temperature. She now leads a nationally distributed team to advance and integrate diverse models of water resources at basin-to-national extents.

Samantha Oliver is a researcher at the Upper Midwest Water Science Center at the U.S. Geological Survey. She has a PhD from the University of Wisconsin Madison in Freshwater and Marine Sciences and a Master's degree from the University of Minnesota Duluth in Integrated Biosciences. Her current research at the USGS is centered around broad scale water quality problems and using data-driven approaches for predicting water quality.

Jordan Read is the Executive Director at the Consortium of Universities for the Advancement for Hydrologic Science, Inc. (CUAHSI). In this role, Jordan develops and implements the overall strategy for CUAHSI, maintains and builds partner relationships, and helps to ensure sustainability of the organization. Prior to joining CUAHSI, Jordan established the U.S. Geological Survey Water Data Science Branch to advance capabilities in modeling, data visualization, and reproducible science. Jordan has a Ph.D. in Civil Engineering (Environmental Fluid Mechanics) from the University of Wisconsin - Madison.

Xiaowei Jia is an Assistant Professor in the Department of Computer Science at the University of Pittsburgh. His research interests include knowledge-guided data science and spatio-temporal data mining for societally important applications. A major highlight of his research is a general paradigm Physics-Guided Machine Learning for combining machine learning and physics-based modeling approaches. He is the recipient of the University of Minnesota Best Dissertation Award and Best Paper Awards from SDM 22, ICDM 21, SDM 21, ASONAM 16, and BIBE 14.

References

[1] Hipsey, M.R., et al.: A general lake model (glm 3.0) for linking with high-frequency sensor data from the global lake ecological observatory network (gleon) (2019)

[2] Cox, P., et al.: The impact of new land surface physics on the gcm simulation of climate and climate sensitivity. Climate Dynamics (1999)

[3] Tonks, M.R., et al.: Mechanistic materials modeling for nuclear fuel performance. Annals of nuclear energy (2017)

[4] Gupta, H.V., Nearing, G.S.: Debates—the future of hydrological sciences: A (common) path forward? using models and data to learn: A systems theoretic perspective on the future of hydrological science. WRR (2014)

[5] Lall, U.: Debates—the future of hydrological sciences: A (common) path forward? one water. one world. many climes. many souls. WRR (2014)

[6] McDonnell, J.J., Beven, K.: Debates—the future of hydrological sciences: A (common) path forward? a call to action aimed at understanding velocities, celerities and residence time distributions of the headwater hydrograph. WRR (2014)

[7] Voulodimos, A., Doulamis, N., Doulamis, A., Protopapadakis, E.: Deep learning for computer vision: A brief review. Computational intelligence and neuroscience **2018** (2018)

[8] Zhang, L., Han, J., Wang, H., Car, R., Weinan, E.: Deep potential molecular dynamics: a scalable model with the accuracy of quantum mechanics. Physical review letters **120**(14), 143001 (2018)

[9] Otter, D.W., Medina, J.R., Kalita, J.K.: A survey of the usages of deep learning for natural language processing. IEEE Transactions on Neural Networks and Learning Systems (2020)

[10] Karpatne, A., Atluri, G., Faghmous, J.H., Steinbach, M., Banerjee, A., Ganguly, A., Shekhar, S., Samatova, N., Kumar, V.: Theory-guided data science: A new paradigm for scientific discovery from data. IEEE Transactions on Knowledge and Data Engineering **29**(10), 2318–2331 (2017)

[11] Willard, J., Jia, X., Xu, S., Steinbach, M., Kumar, V.: Integrating scientific knowledge with machine learning for engineering and environmental systems. arXiv preprint arXiv:2003.04919 (2021)

[12] Xu, T., et al.: Data-driven methods to improve baseflow prediction of a regional groundwater model. Computers & Geosciences (2015)

[13] Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N., *et al.*: Deep learning and process understanding for data-driven earth system science. Nature **566**(7743), 195–204 (2019)

[14] Krasnopolsky, V.M., Fox-Rabinovitz, M.S.: Complex hybrid models combining deterministic and machine learning components for numerical climate modeling and weather prediction. Neural Networks **19**(2), 122–134

(2006)

[15] Faghmous, J.H., Kumar, V.: A big data guide to understanding climate change: The case for theory-guided data science. Big data **2**(3), 155–163 (2014)

[16] O'Gorman, P.A., Dwyer, J.G.: Using machine learning to parameterize moist convection: Potential for modeling of climate, climate change, and extreme events. Journal of Advances in Modeling Earth Systems **10**(10), 2548–2563 (2018)

[17] Graham-Rowe, D., Goldston, D., Doctorow, C., Waldrop, M., Lynch, C., Frankel, F., Reid, R., Nelson, S., Howe, D., Rhee, S., *et al.*: Big data: science in the petabyte era. Nature **455**(7209), 8–9 (2008)

[18] Goh, G.B., Hodas, N.O., Vishnu, A.: Deep learning for computational chemistry. Journal of computational chemistry **38**(16), 1291–1307 (2017)

[19] Forssell, U., Lindskog, P.: Combining semi-physical and neural network modeling: An example of its usefulness. IFAC (1997)

[20] Wan, Z.Y., et al.: Data-assisted reduced-order modeling of extreme events in complex dynamical systems. PloS one (2018)

[21] Jia, X., Willard, J., Karpatne, A., Read, J.S., Zwart, J.A., Steinbach, M., Kumar, V.: Physics-guided machine learning for scientific discovery: An application in simulating lake temperature profiles. ACM/IMS Transactions on Data Science (2021)

[22] Fioretto, F., *et al.*: Predicting ac optimal power flows: Combining deep learning and lagrangian dual methods. In: AAAI (2020)

[23] Karpatne, A., Watkins, W., Read, J., Kumar, V.: Physics-guided neural networks (pgnn): An application in lake temperature modeling. arXiv preprint arXiv:1710.11431 (2017)

[24] Read, J.S., Jia, X., Willard, J., Appling, A.P., Zwart, J.A., Oliver, S.K., Karpatne, A., Hansen, G.J., Hanson, P.C., Watkins, W., et al.: Process-guided deep learning predictions of lake water temperature. WRR (2019)

[25] Stewart, R., Ermon, S.: Label-free supervision of neural networks with physics and domain knowledge. In: AAAI (2017)

[26] Hanson, P.C., Stillman, A.B., Jia, X., Karpatne, A., Dugan, H.A., Carey, C.C., Stachelek, J., Ward, N.K., Zhang, Y., Read, J.S., et al.: Predicting lake surface water phosphorus dynamics using process-guided machine learning. Ecological Modelling (2020)

[27] Chen, S., Sammak, S., Givi, P., Yurko, J.P., Jia, X.: Reconstructing high-resolution turbulent flows using physics-guided neural networks. In: 2021 IEEE International Conference on Big Data (Big Data), pp. 1369–1379 (2021). IEEE

[28] Hurtado, D.M., et al.: Deep transfer learning in the assessment of the quality of protein models. arXiv preprint arXiv:1804.06281 (2018)

[29] Sultan, M.M., Wayment-Steele, H.K., Pande, V.S.: Transferable neural networks for enhanced sampling of protein dynamics. Journal of chemical theory and computation (2018)

[30] Daw, A., Thomas, R., Carey, C., Read, J., Appling, A., Karpatne, A.: Physics-guided architecture (pga) of neural networks for quantifying uncertainty in lake temperature modeling. arXiv:1911.02682 (2019)

[31] Muralidhar, N., *et al.*: Phynet: Physics guided neural networks for particle drag force prediction in assembly. In: SDM (2020)

[32] Ling, J., Kurzawski, A., Templeton, J.: Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. Journal of Fluid Mechanics (2016)

[33] Zhang, L., et al.: End-to-end symmetry preserving inter-atomic potential energy model for finite and extended systems. NeurIPS (2018)

[34] Schütt, K.T., et al.: Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. NeurIPS (2017)

[35] Bao, T., Jia, X., Zwart, J., Sadler, J., Appling, A., Oliver, S., Johnson, T.T.: Partial differential equation driven dynamic graph networks for predicting stream water temperature. In: 2021 IEEE International Conference on Data Mining (ICDM), pp. 11–20 (2021). IEEE

[36] Bao, T., Chen, S., Johnson, T.T., Givi, P., Sammak, S., Jia, X.: Physics guided neural networks for spatio-temporal super-resolution of turbulent flows. In: Uncertainty in Artificial Intelligence, pp. 118–128 (2022). PMLR

[37] Jia, X., Willard, J., Karpatne, A., Read, J., Zwart, J., Steinbach, M., Kumar, V.: Physics guided rnns for modeling dynamical systems: A case study in simulating lake temperature profiles. In: SDM (2019)

[38] Jia, X., Xie, Y., Li, S., Chen, S., Zwart, J., Sadler, J., Appling, A., Oliver, S., Read, J.: Physics-guided machine learning from simulation data: An application in modeling lake and river systems. In: 2021 IEEE International Conference on Data Mining (ICDM), pp. 270–279 (2021).

IEEE

[39] Data release: Process-based predictions of lake water temperature in the Midwest US. https://www.usgs.gov/data/data-release-process-based-predictions-lake-water-temperature-midwest-us

[40] Predicting water temperature in the Delaware River Basin. https://www.usgs.gov/data/predicting-water-temperature-delaware-river-basin

[41] Data release: A large-scale database of modeled contemporary and future water temperature data for 10,774 Michigan, Minnesota and Wisconsin Lakes. https://www.usgs.gov/data/data-release-a-large-scale-database-modeled-contemporary-and-future-water-temperature-data

[42] Anderson, B., Hy, T.S., Kondor, R.: Cormorant: Covariant molecular neural networks. In: NeurIPS (2019)

[43] Muralidhar, N., *et al.*: Incorporating prior domain knowledge into deep neural networks. In: IEEE Big Data (2018). IEEE

[44] Ham, Y.-G., Kim, J.-H., Luo, J.-J.: Deep learning for multi-year enso forecasts. Nature (2019)

[45] Lu, J., Yao, K., Gao, F.: Process similarity and developing new process models through migration. AIChE journal (2009)

[46] Jia, X., Zwart, J., Sadler, J., Appling, A., Oliver, S., Markstrom, S., Willard, J., Xu, S., Steinbach, M., Read, J., *et al.*: Physics-guided recurrent graph model for predicting flow and temperature in river networks. In: SDM (2021). SIAM

[47] Kratzert, F., Klotz, D., Shalev, G., Klambauer, G., Hochreiter, S., Nearing, G.: Towards learning universal, regional, and local hydrological behaviors via machine learning applied to large-sample datasets. Hydrology and Earth System Sciences **23**(12), 5089–5110 (2019)

[48] Lu, J., et al.: Model migration with inclusive similarity for development of a new process model. Industrial & engineering chemistry research (2008)

[49] Li, Y., Gao, J., Meng, C., Li, Q., Su, L., Zhao, B., Fan, W., Han, J.: A survey on truth discovery. ACM Sigkdd Explorations Newsletter (2016)

[50] Van Engelen, J.E., Hoos, H.H.: A survey on semi-supervised learning. Machine Learning **109**(2), 373–440 (2020)

[51] Ouali, Y., Hudelot, C., Tami, M.: An overview of deep semi-supervised learning. arXiv preprint arXiv:2006.05278 (2020)

[52] Weiss, K., Khoshgoftaar, T.M., Wang, D.: A survey of transfer learning. Journal of Big data **3**(1), 1–40 (2016)

[53] Wilson, G., Cook, D.J.: A survey of unsupervised deep domain adaptation. ACM Transactions on Intelligent Systems and Technology (TIST) **11**(5), 1–46 (2020)

[54] Ngo, B.H., Park, J.H., Park, S.J., Cho, S.I.: Semi-supervised domain adaptation using explicit class-wise matching for domain-invariant and class-discriminative feature learning. IEEE Access **9**, 128467–128480 (2021)

[55] Xu, L., Hu, C., Mei, K.: Semi-supervised regression with manifold: A bayesian deep kernel learning approach. Neurocomputing **497**, 76–85 (2022)

[56] Jaiswal, A., Babu, A.R., Zadeh, M.Z., Banerjee, D., Makedon, F.: A survey on contrastive self-supervised learning. Technologies **9**(1), 2 (2020)

[57] Larsson, G., Maire, M., Shakhnarovich, G.: Colorization as a proxy task for visual understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6874–6883 (2017)

[58] Appice, A., Loglisci, C., Malerba, D.: Active learning via collective inference in network regression problems. Information Sciences **460**, 293–317 (2018)

[59] Markstrom, S.L., et al.: Prms-iv, the precipitation-runoff modeling system, version 4. USGS Techniques and Methods (6-B7) (2015)

[60] Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: International Conference on Machine Learning, pp. 1126–1135 (2017). PMLR

[61] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)

[62] Luong, M.-T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. arXiv preprint arXiv:1508.04025 (2015)

[63] Geological Survey, U.S.: National water information system data available on the world wide web (usgs water data for the nation) (2016)

[64] Read, E.K., Carr, L., De Cicco, L., Dugan, H.A., Hanson, P.C., Hart, J.A., Kreft, J., Read, J.S., Winslow, L.A.: Water quality data for national-scale

aquatic research: The water quality portal. Water Resources Research **53**(2), 1735–1745 (2017)

[65] Regan, R.S., Markstrom, S.L., Hay, L.E., Viger, R.J., Norton, P.A., Driscoll, J.M., LaFontaine, J.H.: Description of the national hydrologic model for use with the precipitation-runoff modeling system (prms). Technical report, US Geological Survey (2018)

[66] gridMET - Climatology Lab. http://www.climatologylab.org/gridmet.html

[67] Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial discriminative domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7167–7176 (2017)

[68] Burns, D.A., Murdoch, P.S., Lawrence, G.B., Michel, R.L.: Effect of groundwater springs on no3- concentrations during summer in catskill mountain streams. WRR (1998)