ClimSim: An open large-scale dataset for training high-resolution physics emulators in hybrid multi-scale climate simulators

Sungduk Yu[1]*, **Walter M. Hannah**[2], **Liran Peng**[1], **Mohamed Aziz Bhouri**[3], **Ritwik Gupta**[4], **Jerry Lin**[1], **Björn Lütjens**[8], **Justus C. Will**[1], **Tom Beucler**[9], **Bryce E. Harrop**[5], **Benjamin R. Hilman**[6], **Andrea M. Jenney**[1,7], **Savannah L. Ferretti**[1], **Nana Liu**[1], **Anima Anandkumar**[10], **Noah D. Brenowitz**[10], **Veronika Eyring**[11], **Pierre Gentine**[3], **Stephan Mandt**[1], **Jaideep Pathak**[10], **Carl Vondrick**[3], **Rose Yu**[12], **Laure Zanna**[13], **Ryan P. Abernathey**[3], **Fiaz Ahmed**[14], **David C. Bader**[2], **Pierre Baldi**[1], **Elizabeth A. Barnes**[15], **Gunnar Behrens**[11], **Christopher S. Bretherton**[16], **Julius J. M. Busecke**[3], **Peter M. Caldwell**[2], **Wayne Chuang**[3], **Yilun Han**[17], **Yu Huang**[3], **Fernando Iglesias-Suarez**[11], **Sanket Jantre**[18], **Karthik Kashinath**[10], **Marat Khairoutdinov**[19], **Thorsten Kurth**[10], **Nicholas J. Lutsko**[12], **Po-Lun Ma**[5], **Griffin Mooers**[1], **J. David Neelin**[14], **David A. Randall**[15], **Sara Shamekh**[3] **Akshay Subramaniam**[10], **Mark A. Taylor**[6], **Nathan M. Urban**[17], **Janni Yuval**[8] **Guang J. Zhang**[12], **Tian Zheng**[3], **Michael S. Pritchard**[1,10]

[1]UCI, [2]LLNL, [3]Columbia, [4]UCB, [5]PNNL, [6]SNL, [7]OSU, [8]MIT, [9]UNIL, [10]NVIDIA
[11]DLR, [12]UCSD, [13]NYU, [14]UCLA, [15]CSU, [16]Allen AI, [17]Tsinghua, [18]BNL, [19]SUNY

Abstract

Modern climate projections lack adequate spatial and temporal resolution due to computational constraints. A consequence is inaccurate and imprecise prediction of critical processes such as storms. Hybrid methods that combine physics with machine learning (ML) have introduced a new generation of higher fidelity climate simulators that can sidestep Moore's Law by outsourcing compute-hungry, short, high-resolution simulations to ML emulators. However, this hybrid ML-physics simulation approach requires domain-specific treatment and has been inaccessible to ML experts because of lack of training data and relevant, easy-to-use workflows. We present ClimSim, the largest-ever dataset designed for hybrid ML-physics research. It comprises multi-scale climate simulations, developed by a consortium of climate scientists and ML researchers. It consists of 5.7 billion pairs of multivariate input and output vectors that isolate the influence of locally-nested, high-resolution, high-fidelity physics on a host climate simulator's macro-scale physical state.

The dataset is global in coverage, spans multiple years at high sampling frequency, and is designed such that resulting emulators are compatible with downstream coupling into operational climate simulators. We implement a range of deterministic and stochastic regression baselines to highlight the ML challenges and their scoring. The data (https://huggingface.co/datasets/LEAP/ClimSim_high-res[2]) and code (https://leap-stc.github.io/ClimSim) are released openly to support the development of hybrid ML-physics and high-fidelity climate simulations for the benefit of science and society.

*Corresponding author: sungduk@uci.edu

[2]Also available in a low-resolution version (https://huggingface.co/datasets/LEAP/ClimSim_low-res) and an aquaplanet version https://huggingface.co/datasets/LEAP/ClimSim_low-res_aqua-planet

Preprint. Under review.

1 Introduction

1.1 Overview

Predictions from numerical physical simulations are the primary tool informing policy on climate change. However, current climate simulators poorly represent cloud and extreme rainfall physics [1, 2] despite stretching the limits of the world's most powerful supercomputers. The complexity of the Earth system imposes significant restrictions on the spatial resolution we can use in these simulations [3]. Physics occurring on scales smaller than the temporal and/or spatial resolutions of climate simulations are commonly represented using empirical mathematical representations called "parameterizations". Unfortunately, assumptions in these parameterizations often lead to errors that can grow into inaccuracies in the future predicted climate.

Machine learning (ML) is an attractive approach to emulate the complex nonlinear sub-resolution physics—processes occurring on scales smaller than the resolution of the climate simulator—at a lower computational complexity. Their implementation has the exciting possibility of resulting in climate simulations that are both cheaper and more accurate than they currently are [4, 5]. Current climate simulators have a typical smallest resolvable scale of 80–200 km, equivalent to the size of a typical U.S. county. However, accurately representing cloud formation requires a resolution of 100 m or finer, demanding six orders of magnitude increase in computational intensity. Exploiting ML remains a conceivable solution to sidestep the limitations of classical computing [5]: resulting hybrid-ML climate simulators combine traditional numerical methods—which solve the equations governing large-scale fluid motions of Earth's atmosphere—with ML emulators of the macro-scale effects of small-scale physics. Instead of relying on heuristic assumptions about these small-scale processes, the emulators learn directly from data generated by short-duration, high-resolution simulations [4, 6–18]. The task is essentially a regression problem: in the climate simulation, an ML parameterization emulator returns the large-scale outputs—changes in wind, moisture, or temperature—that occur due to unresolved small-scale (sub-resolution) physics, given large-scale resolved inputs (e.g., temperature, wind velocity; see Section 4).

While several proofs of concept have emerged in recent years, hybrid-ML climate simulators have yet to be advanced to operational use. Obtaining sufficient training data is a major challenge impeding interest from the ML community. This data must contain all macro-scale variables that regulate the behavior of sub-resolution physics and be compatible with downstream hybrid ML-climate simulations. Addressing this using training data from uniformly high-resolution simulations has proven to be very expensive and can lead to issues when coupled to a host climate simulation.

A promising solution is to utilize multi-scale climate simulation methods to generate training data. Crucially, these provide a clean interface between the emulated high-resolution physics and the host climate simulator's planetary-scale dynamics [19]. In theory, this makes downstream hybrid coupled simulation approachable and tractable. In practice, the full potential of multi-scale methods remains largely untapped due to a scarcity of existing datasets, exacerbated by the combination of operational simulation code complexity and the need for domain expertise in choosing variables.

We introduce ClimSim, the largest and most physically comprehensive dataset for training ML emulators of atmospheric storms, clouds, turbulence, rainfall, and radiation for use in hybrid-ML climate simulations. ClimSim is a comprehensive collection of inputs and outputs from physical climate simulations using the multi-scale method. ClimSim was prepared by atmospheric scientists and climate simulator developers to lower the barriers to entry for ML experts on this important problem. Our benchmark dataset serves as a foundation for developing robust frameworks that emulate parameterizations for cloud and extreme rainfall physics, and their interaction with other sub-resolution processes. These frameworks enable online coupling within the host coarse-resolution climate simulator, ultimately improving the performance and accuracy of climate simulators used for long-term projections.

1.2 Concepts and Terminology from Earth Science

Convective parameterization: In atmospheric science, "convection" refers to storm cloud and rain development, as well as the associated turbulent air motions. Convective parameterizations represent the integrated effects of these processes, such as the vertical transport of heat, moisture, and momentum within the atmosphere, and condensational heating and drying, on the temporal and spatial

scale of the host climate simulator [20–22]. Stochastic parameterizations represent sub-resolution ("sub-grid scale" in the terminology of Earth science) effects as stochastic processes, dependent on grid-scale variable inputs [23, 24] to capture variations arising from sub-grid scale dynamics.

Multi-scale climate simulators: Multi-scale climate simulation is a technique that represents convection without a convective parameterization, by deploying a smaller-scale, high-resolution cloud-resolving simulator nested within each host grid column of a climate simulator [25–29]. The smaller-scale simulator explicitly resolves the detailed behavior of clouds and their turbulent motions at both a higher spatial and temporal resolution (but with a smaller domain) than the host simulator. This improves the accuracy of the host simulations, but comes at a high computational cost [30, 31]. The time-integrated and horizontally averaged influence of the resolved convection is fed upscale to the host climate simulator, and is the target of hybrid ML-climate simulation approaches.

Significance of precipitation processes for climate impacts: In climate simulation, changes in precipitation with warming is a particularly important issue. The frequency of extreme precipitation events increases with warming [32–34], with corresponding societal impacts [35]. Current climate simulators agree on the direction of this change, but exhibit large spread in the quantitative rate of increase with warming [36, 37].

2 Related Work

There have been several recent efforts to produce hybrid-ML emulators using multi-scale climate simulations, analogous to ClimSim [4, 10–16]. Most of these focused on simple aquaplanets [4, 10–13, 16] and those that included real geography [14, 15] did not include enough variables for complete land-surface coupling, to our knowledge. Most examine simple multi-layer perceptrons except for [12, 15], who used a ResNet architecture, and [38] who used a variational encoder-decoder that accounts for stochasticity. Although downstream hybrid testing in real-geography settings is error-prone, [15] demonstrates some hybrid stability. Compressing input data to avoid causal confounders may improve downstream accuracy [16], and methods have been proven to enforce physical constraints [39, 40].

Compared to the training data used above, ClimSim's comprehensive variable coverage is unprecedented, including all variables needed to couple to and from a land system simulator and enforce physical constraints. Its availability across coarse-resolution, high-resolution, aquaplanet and real-geography use cases is also new to the community. Successful ML innovations with ClimSim can have a downstream impact since it is based on a state-of-the-art multi-scale climate simulator that is actively supported by a mission agency (U.S. Department of Energy).

In non-multi-scale settings, an important body of related work [6–9] has made exciting progress on using analogous hybrid ML approaches to reduce biases in uniform resolution climate simulations, including in an operational climate code with land coupling and downstream hybrid stability [17, 18] (see Supplemental Information). Other related work includes full model emulation (FME) for short-term weather prediction [41–43]. Whether this approach is possible for climate simulation using the high-frequency output of its state variables remains an open question. For instance, it has recently been shown that incorporating spherical geometry and resolution invariance through spherical Fourier neural operators leads to stability of long rollouts [42]. While ClimSim is focused on hybrid-ML climate simulation and we do not demonstrate FME baselines, ClimSim contains full atmospheric state variable sampling well suited for the task.

3 ClimSim Dataset Construction

Experiment outline: ClimSim presents a regression problem with mapping from a multivariate input vector, with inputs $x \in \mathbb{R}^{d_i}$ of size $d_i = 617$ and targets $y \in \mathbb{R}^{d_o}$ of size $d_o = 368$ (Figure 1). The input represents the local vertical structure (in horizontal location and time) of macro-scale state variables in a multi-scale physical climate simulator before any adjustments from sub-grid scale convection and radiation are made. The input also includes concatenated scalars containing boundary conditions of incoming radiation at the top of the atmospheric column, and land surface model constraints at its base. The target vector contains the tendencies of the same state variables representing the redistribution of mass and water, microphysical water species conversions, and radiative heating feedbacks associated with explicitly resolved convection. This brackets the change

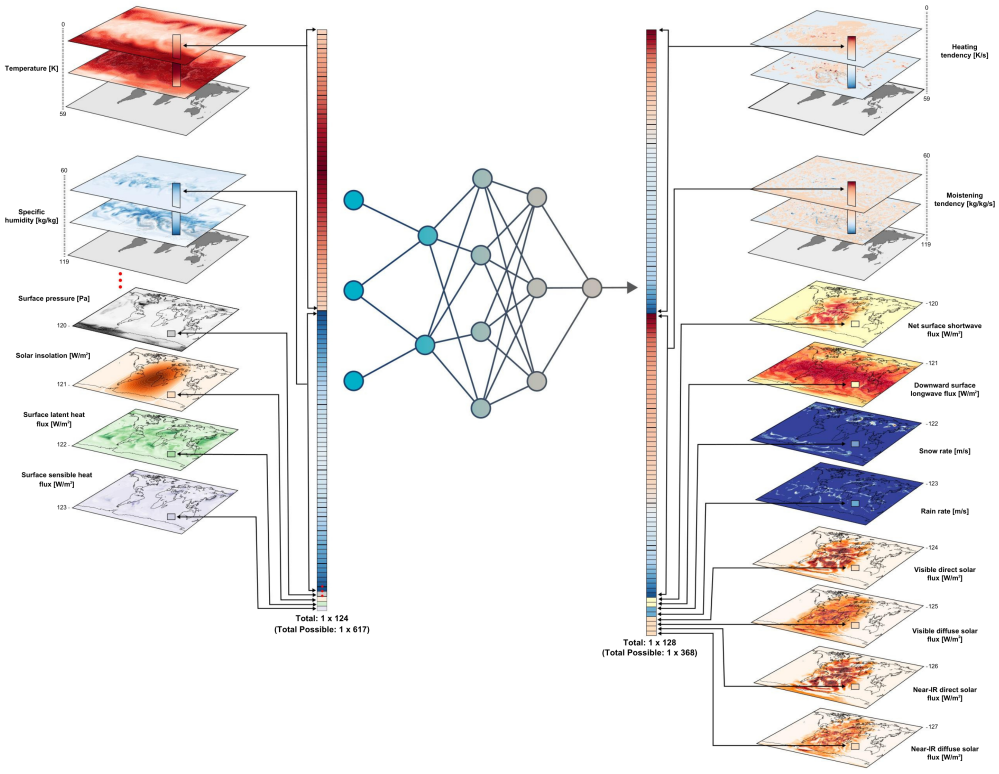

Figure 1: The spatially-local version of ClimSim that our baselines are scored on. A spatially-global version of the problem that expands to the full list of variables would be useful to try.

in atmospheric state after tens of thousands of computationally intensive, spatially nested simulators of explicit cloud physics have completed a temporally-nested integration. The ultimate goal is to outsource these physics to ML by mapping inputs to targets at comparable fidelity. The target vector includes scalar fields and fluxes from the bottom of the atmospheric column expected by the land surface model component that it must couple to; land-atmosphere coupling is important to predicting regional water cycle dynamics [44, 45].

Locality vs. nonlocality: A spatially-global version of the problem could be of practical use for improving ML via helpful spatial context [46, 47]. In such a case, the problem becomes 2D→2D regression and would encompass inputs $x \in \mathbb{R}^{d_i}$ of size $d_i = 617 \times 21{,}600$ (grid columns) and targets, $y \in \mathbb{R}^{d_o}$, of size $d_o = 368 \times 21{,}600$. In contrast to typical image-to-image translation or spatio-temporal prediction problems in ML that involve RGB images of mega-pixel dimensionality, the task at hand involves many more channels but of lower dimensionality. Further details about the climate simulations and data, including complete variable lists, can be found in the Supplementary Information.

Dataset collection: We ran the E3SM-MMF multi-scale climate simulator [28, 29, 48, 49], using multiple NVIDA A100 GPUs for a total of ∼9,800 GPU-hours. We saved global instantaneous values of the atmospheric state before and after high-resolution calculations occurred, isolating state updates due to explicitly-resolved moist convection, boundary layer turbulence, and radiation; details of the climate simulator configuration can be found in the Supplementary Information. These data were saved at 20 minute intervals for 10 simulated years, resulting in 5.7 billion samples for the high-resolution ($1.5^\circ \times 1.5^\circ$) simulation that uses a horizontal grid with 21,600 grid columns spanning the globe (5.7 billion ≈ 21,600 samples per timestep × 72 samples per day × 3,650 days). We also ran two additional simulations that use a low-resolution ($11.5^\circ \times 11.5^\circ$) horizontal grid with only 384 grid columns spanning the globe, resulting in 100 million samples for each simulation. These low-resolution options allow for fast prototyping of ML models, due to smaller training data volumes and less geographic complexity. One low-resolution simulation uses an "aquaplanet" configuration,

i.e., a lower boundary condition of specified sea surface temperature, invariant in the longitudinal dimension with no seasonal cycle. This is the simplest prototyping dataset, removing variance associated with continents and time-varying boundary conditions. The total data volume is 41.2 TB for the high-resolution dataset and 744 GB for each of the low-resolution datasets.

Dataset interface: Raw model outputs emerge from the climate simulator as standard NetCDF files which can be easily parsed in any language. Each timestep yields files containing input and target vectors separately, resulting in a total of 525,600 files for each of the three datasets. To prevent redundancy, variable metadata and grid information was saved separately.

The raw tensors from the climate simulations are initially either 2D or 3D, depending on the variable. For 2D tensors, the dimensions represent time and horizontal location, with each location on the sphere (referred to as "grid columns" earlier) indexed along a single axis. 2D variables include solar insolation, snow depth over land, surface energy fluxes, and surface precipitation rate. 3D tensors include the additional dimension representing altitude relative to the Earth's surface, for height-varying state variables like temperature, humidity, and wind vector components. Separate files are used to store each time step and variable. ClimSim includes a total of 24 2D variables and 10, 3D variables (see Table 1 in Supplementary Information).

Dataset split: The 10-year datasets are divided into: a) a training and validation spanning the first 8 years (0001-02 to 0009-01; YYYY-MM), excluding the first simulated month for numerical spin-up, and b) a test set spanning the remaining two years (i.e., 0009-03 to 0011-02). A one-month gap is intentionally introduced between the two sets to prevent test set contamination via temporal correlation. Both sets are stored separately in our data repositories.

4 Experiments

To guide ML practitioners using ClimSim, we provide an example ML workflow using the low-resolution dataset for the task described in Section 1. We focus on emulating a subset of input and target variables with inputs $x \in \mathbb{R}^{d_i}$ of size $d_i = 124$, and targets $y \in \mathbb{R}^{d_o}$ of size $d_o = 128$ (Figure 1, Table 1).

Input	Size	**Target**	Size
Temperature [K]	60	Heating tendency, dT/dt [K/s]	60
Specific humidity [kg/kg]	60	Moistening tendency, dq/dt [kg/kg/s]	60
Surface pressure [Pa]	1	Net surface shortwave flux, NETSW [W/m^2]	1
Insolation [W/m^2]	1	Downward surface longwave flux, FLWDS [W/m^2]	1
Surface latent heat flux [W/m^2]	1	Snow rate, PRECSC [m/s]	1
Surface sensible heat flux [W/m^2]	1	Rain rate, PRECC [m/s]	1
		Visible direct solar flux, SOLS [W/m^2]	1
		Near-IR direct solar flux, SOLL [W/m^2]	1
		Visible diffused solar flux, SOLSD [W/m^2]	1
		Near-IR diffused solar flux, SOLLD [W/m^2]	1

Table 1: A subset of input and target variables used in our experiment. Dimension length 60 corresponds to the total number of vertical levels of the climate simulator.

Train/validation split: We divide the 8-year train/validation set into the first 7 years (i.e., 0001-02 to 0008-01 in the raw filenames' "year-month" notation) for training and the subsequent 1 year (0008-02 to 0009-01) for validation.

Preprocessing workflow: Our preprocessing steps were (1) downsample in time by using every 7th sample, (2) collapse horizontal location and time into a single sample dimension, (3) normalize variables by subtracting the mean and dividing by the range, with these statistics calculated separately at each of the 60 vertical levels for the four variables with vertical dependence, and (5) concatenate variables into multi-variate input and output vectors for each sample (Figure 1). The heating tendency target $\frac{\partial T}{\partial t}$ (i.e., time rate of temperature T) was calculated from the raw climate simulator output as $(T_{after} - T_{before})/\Delta t$, where $\Delta t = 1200$ seconds) is the climate simulator's known macro-scale timestep. Likewise, the moisture tendency was calculated via taking the difference of humidity state variables recorded before versus after the convection and radiation calculations. This target variable

transformation is done so that we can compare the performance of our baseline models to that of previously published models that reported errors of emulated tendencies [14, 38]. Additionally, this transformation implicitly normalizes the target variables leading to better convergence properties for ML algorithms. We encourage the same transformation for users who wish to add condensate (liquid, cloud) tendencies to the target vector. Given the domain-specific nature of the preprocessing workflow, we provide scripts in the GitHub repository for workflow reproduction.

4.1 Baseline Architectures

Multi-layer Perceptron (MLP) is a fully connected, feed-forward neural network. The MLP architecture used for our experiments is optimized via an extensive random hyperparameter search with 8,257 trials across a set of hidden-layer dimensions [768, 640, 512, 640, 640, 128], and output layer activations (ReLU and linear). The optimizer (RAdam), batch size (3072), and interior activation function (with Leaky ReLU with $\alpha = 0.15$) were fixed.

Randomized Prior Network (RPN) is an ensemble model [50]. Each member of the RPNs is built as the sum of a trainable surrogate model and a non-trainable (so-called prior) surrogate model; in this example MLP for simplicity. Multiple replicas of the networks are constructed by independent and random sampling of both trainable and non-trainable parameters [51, 52]. RPNs also resort to data bootstrapping in order to mitigate the uncertainty collapse of the ensemble method when tested beyond the training data points [52]. Data bootstrapping consists of sub-sampling and randomization of the data each network in the ensemble sees during training. Hyperparameters of individual MLPs do not need to be tuned from scratch and are chosen based on the hyperparameter search mentioned in the previous point. RPN ensembles of 128 networks are considered as justified in [51].

Convolutional Neural Network (CNN) uses a 1D ResNet-style network. Each ResNet block contains two 1-D convolutional layers and a skip connection. CNNs can learn spatial structure and have outperformed MLP and graph-based networks in [53]. The inputs and outputs for the CNN are stacked in the channel dimensions, such that the mapping is $60 \times 6 \rightarrow 60 \times 10$. Accordingly, global variables have been repeated along the vertical dimension. The baseline CNN has a depth of 12, each convolutional layer has a width of 406, and a kernel size of 3. It was trained for 10 epochs; complete details on hyperparameters are in the Supplementary Information.

Heteroskedastic Regression (HSR) predicts a separate mean and standard deviation for each output variable, using a regularized MLP (see Supplementary Information for details).

Conditional Variational Autoencoder (cVAE) uses amortized variational inference to fit a deep generative model that is conditioned on the input and can produce samples from a complex predictive distribution (see Supplementary Information for details).

Variable	MAE [W/m^2]					R^2				
	CNN	HSR	MLP	RPN	cVAE	CNN	HSR	MLP	RPN	cVAE
dT/dt	**2.585**	2.845	2.683	2.685	2.732	**0.627**	0.568	0.589	0.617	0.590
dq/dt	**4.401**	4.784	4.495	4.592	4.680	–	–	–	–	–
NETSW	18.85	19.82	**13.36**	18.88	19.73	0.944	0.959	**0.983**	0.968	0.957
FLWDS	8.598	6.267	**5.224**	6.018	6.588	0.828	0.904	**0.924**	0.912	0.883
PRECSC	3.364	3.511	**2.684**	3.328	3.322	–	–	–	–	–
PRECC	37.83	42.38	**34.33**	37.46	38.81	**0.077**	-68.35	-38.69	-67.94	-0.926
SOLS	10.83	11.31	**7.97**	10.36	10.94	0.927	0.929	**0.961**	0.943	0.929
SOLL	13.15	13.60	**10.30**	12.96	13.46	0.916	0.916	**0.948**	0.928	0.915
SOLSD	5.817	6.331	**4.533**	5.846	6.159	0.927	0.923	**0.956**	0.940	0.921
SOLLD	5.679	6.215	**4.806**	5.702	6.066	0.813	0.797	**0.866**	0.837	0.796

Table 2: Summary statistics of global-mean, time-mean target variables for each baseline architecture.

4.2 Evaluation Metrics

Our evaluation metrics are computed separately for each variable in the output vector. Mean Absolute Error (MAE) and the coefficient of determination (R^2) are calculated independently at each horizontal and vertical location, and then averaged horizontally and vertically to produce the summary statistics

in Figure 2. For the vertically-varying fields, we first form a mass-weighting and then convert moistening and heating tendencies into common energy units in Watts per square meter as in [54]. We also report continuous ranked probability scores (CRPS) for all considered models in the Supplementary Information.

4.3 Baseline Model Results

Figure 2 summarizes the error characteristics. Whereas heating and moistening rates have comparable global mean MAE, behind a common background vertical structure (Fig 2 b,c) the coefficient of determination R^2 (d,e) reveals that certain architectures (RPN, HSR, cVAE, CNN) consistently perform better in the upper atmosphere (model level < 30) whereas the highly optimized MLP model outperforms in the lower atmosphere (model level > 30) and therefore the global mean (Table 2). Additional tables and figures that reveal the geographic and vertical structure of these errors, and analysis of stochastic metrics, are included in the Supplementary Information.

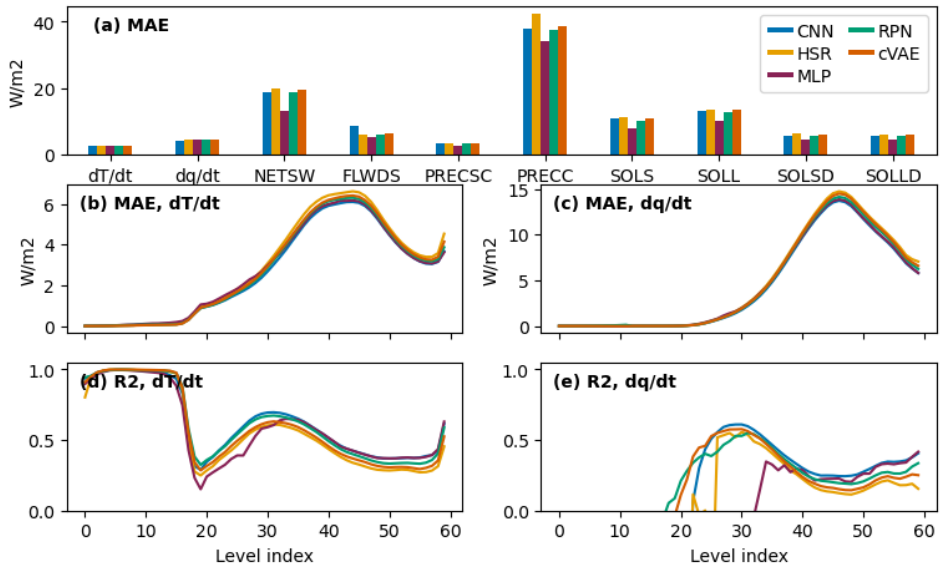

Figure 2: (a) Summary, where $\frac{\partial T}{\partial t}$ and $\frac{\partial q}{\partial t}$ are the tendencies of temperature and specific humidity, respectively, and were vertically integrated with mass weighting. (b,c) retain the vertical structure of MAE and (d,e) R^2. Refer to Table 1 for variable definitions.

4.4 Physics-Informed Guidance to Improve Generalizability and Coupled Performance

Physical constraints: Mass and energy conservation are important criteria for Earth System simulation. If these terms are not conserved, errors in estimating sea level rise or temperature change over time may become as large as the signals we hope to measure. Enforcing conservation on emulated results helps constrain results to be physically plausible and reduce the potential for errors accumulating over long time scales. We discuss how to do this and enforce additional constraints, such as non-negativity for precipitation, condensate, and moisture variables in the Supporting Information.

Stochasticity and memory: The results of the embedded convection calculations regulating d_o are chaotic, and thus worthy of stochastic architectures, as in our RPN, HSR, and cVAE baselines. These solutions are likewise sensitive to sub-grid initial state variables from an interior nested spatial dimension that has not been included in our data.

Temporal locality: Incorporating the previous time steps' target or feature in the input vector inflation could be beneficial as it captures some information about this convective memory and utilizes temporal autocorrelations present in atmospheric data.

Causal pruning: A systematic and quantitative pruning of the input vector based on objectively assessed causal relationships to subsets of the target vector has been proposed as an attractive

preprocessing strategy, as it helps remove spurious correlations due to confounding variables and optimize the ML algorithm [16].

Normalization: Normalization that goes beyond removing vertical structure could be strategic, such as removing the geographic mean (e.g., latitudinal, land/sea structure) or composite seasonal variances (e.g., local smoothed annual cycle) present in the data. For variables exhibiting exponential variation and approaching zero at the highest level (e.g., metrics of moisture), log-normalization might be beneficial.

Expanded resolution, complete inputs and outputs: Our baselines have focused on the low-resolution dataset, for ease of data volume, and using only a subset of the available inputs and outputs. This illustrates the essence of the ML challenge. We encourage users who discover competitive fits in this approachable limit to expand to all inputs/outputs in the high-resolution, real-geography dataset, for which successful fits become operationally relevant.

Further ML approaches: Recent methods to capture multi-scale processes using neural operators that learn in a discretization-invariant manner and can predict at higher resolutions than available during training time [55] may be attractive. Their performance can be further enhanced by incorporating physics-informed losses at a higher resolution than available training data [56]. Ideas on ML modeling for sub-grid closures from adjacent fields like turbulent flow physics and reactive flows can also be leveraged for developing architectures with an inductive bias for known priors [57], easing prediction of stiff non-linear behavior [58–60], generative modeling with physical constraints [61, 62] and for interpretability of the final trained models [58].

5 Limitations and Other Applications

Idealizations: The multi-scale climate simulator used to produce ClimSim (E3SM-MMF) assumes scale separation, i.e., that convection can be represented as laterally periodic within the grid size of the host simulator, and neglects sub-grid scale representations of topographic and land-surface variability. Despite these simplifications, the data adequately captures many essential aspects of the ML problem, such as stochasticity, and interactions across radiation, microphysics, and turbulence.

Hybrid testing: Inclusion of a natural path for downstream testing of learned physics emulators as fully coupled components of a hybrid-ML climate simulator is vital. However, such a workflow is not yet included in ClimSim, since there is no easy way for the ML community to run many hybridized variants of the E3SM-MMF in a distributed high-performance GPU computing infrastructure via a lightweight API. It is our eventual goal to enable such a protocol, since, in the long term, it is in this downstream environment where ML researchers should expect to have their maximum impact on the field of hybrid-ML climate simulation. Meanwhile, ClimSim provides the first step.

Stochasticity: One open problem that the dataset may allow assessing is understanding the role of stochasticity in hybrid-ML simulation. While primarily used as a dataset for regression, it would be also interesting to assess and understand the degree to which different variables are better modeled as stochastic or deterministic, or if the dataset gives rise to heavy-tailed or even multi-modal conditional distributions that are important to capture. To date, these questions have been raised based on physical conjectures [e.g., 63] but remain to be addressed in the ML-based parameterization literature. For instance, precipitation distributions have long tails that are projected to lengthen under global warming [34, 64]—and will thus tend to generate out-of-sample extremes. ClimSim could help construct optimal architectures to capture precipitation tails and other impactful climate variables such as surface temperature.

Interpretability: This dataset could also be utilized to discover physically interpretable models for atmospheric convection, radiation, and microphysics. A possible workflow would apply dimensionality reduction techniques to identify dominant vertical variations, followed by symbolic regression to recover analytic expressions [65, 66].

Generalizability: Although the impacts of global warming and inter-annual variability are absent in this initial version of ClimSim, important questions surrounding climate-convection interactions can begin to be addressed. One strategy would involve partitioning the data such that the emulator is trained on cold columns, but validated on warm columns, where warmth could be measured by surface temperatures, as in [54]. However, the results from this approach may also reflect the dependence of convection on the geographical distribution of surface temperatures in the current climate and

should be interpreted with caution. To optimally engage ML researchers in solving the climate generalization problem, a multi-climate extension of ClimSim should be developed that includes physical simulations that samples future climate states and more internal variability.

6 Conclusion and Future Work

We introduce ClimSim, the most physically comprehensive dataset yet published for training ML emulators of atmospheric storms, clouds, turbulence, rainfall, and radiation for use in hybrid-ML climate simulation. It contains all inputs and outputs necessary for downstream coupling in a full-complexity multi-scale climate simulator. We conduct a series of experiments on a subset of these variables that demonstrate the degree to which climate data scientists have been able to fit their deterministic and stochastic components. We hope ML community engagement in ClimSim will advance fundamental ML methodology and clarify the path to producing increasingly skillful sub-grid physics emulators that can be reliably used for operational climate simulation. If so, this opens up an exciting future in which the computational limits that currently constrain climate simulation can be reconsidered.

We plan to soon extend ClimSim to include, first, a sampling of multiple future climate states. Second, we aim to provide a protocol for downstream hybrid simulation testing. We hope lessons learned in our chosen limit of multi-scale atmospheric simulation will have applicability in other sub-fields of Earth System Science where computational constraints are currently a barrier to including explicit representations of more systems of nested complexity.

Acknowledgements

This work is broadly supported across countries and agencies. Primary support is by the National Science Foundation (NSF) Science and Technology Center (STC) Learning the Earth with Artificial Intelligence and Physics (LEAP), Award # 2019625-STC and the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy (DOE) Office of Science (SC), the National Nuclear Security Administration, and the Energy Exascale Earth System Model project, funded by DOE grant DE-SC0022331. M.S.P, S.Y., L.P., A.M.J., J.L., N.L., and G.M. further acknowledge support from the DOE (DE-SC0023368) and NSF (AGS-1912134). R.Y, S.M, P.G, M.P. acknowledge funding from the DOE Advanced Scientific Computing Research (ASCR) program (DE-SC0022255). V.E., P.G., G.B., and F.I.-S. acknowledge funding from the European Research Council Synergy Grant (Agreement No. 855187) under the Horizon 2020 Research and Innovation Programme. E.A.B. was supported, in part, by NSF grant AGS-2210068. S.J. acknowledges funding from DOE ASRC under an Amalie Emmy Noether Fellowship Award in Applied Mathematics (B&R #KJ0401010). M.A.B acknowledges NSF funding from an AGS-PRF Fellowship Award (AGS-2218197). R.G. acknowledges funding from the NSF (DGE-2125913) and the U.S. Department of Defense (DOD). S.M. acknowledges support from an NSF CAREER Award and NSF grant IIS-2007719. This research used resources of the National Energy Research Scientific Computing Center (NERSC), a DOE SC User Facility operated under Contract No. DE-AC02-05CH11231. The Pacific Northwest National Laboratory is operated by Battelle for the DOE under Contract DE-AC05-76RL01830. This work was performed under the auspices of the DOE by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. This work used Bridges-2 at the Pittsburgh Supercomputing Center through allocation ATM190002 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by NSF grants #2138259, #2138286, #2138307, #2137603, and #2138296. This work also utilized the DOD High Performance Computing Modernization Program (HPCMP).

References

[1] IPCC, *Climate Change 2021: The Physical Science Basis. Contribution of Working Group I to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*. 2021.

[2] S. Sherwood, M. J. Webb, J. D. Annan, K. C. Armour, P. M. Forster, J. C. Hargreaves, G. Hegerl, S. A. Klein, K. D. Marvel, E. J. Rohling, *et al.*, "An assessment of earth's climate sensitivity using multiple lines of evidence," *Rev. Geophys.*, vol. 58, no. 4, p. e2019RG000678, 2020.

[3] T. Schneider, J. Teixeira, C. S. Bretherton, F. Brient, K. G. Pressel, C. Schär, and A. P. Siebesma, "Climate goals and computing the future of clouds," *Nat. Clim. Change*, vol. 7, no. 1, pp. 3–5, 2017.

[4] P. Gentine, M. Pritchard, S. Rasp, G. Reinaudi, and G. Yacalis, "Could machine learning break the convection parameterization deadlock?," *Geophys. Res. Lett.*, vol. 45, no. 11, pp. 5742–5751, 2018.

[5] V. Eyring, V. Mishra, G. P. Griffith, L. Chen, T. Keenan, M. R. Turetsky, S. Brown, F. Jotzo, F. C. Moore, and S. van der Linden, "Reflections and projections on a decade of climate science," *Nat. Clim. Change*, vol. 11, no. 4, pp. 279–285, 2021.

[6] C. S. Bretherton, B. Henn, A. Kwa, N. D. Brenowitz, O. Watt-Meyer, J. McGibbon, W. A. Perkins, S. K. Clark, and L. Harris, "Correcting coarse-grid weather and climate models by machine learning from global storm-resolving simulations," *J. Adv. Model. Earth Syst.*, vol. 14, no. 2, p. e2021MS002794, 2022.

[7] S. K. Clark, N. D. Brenowitz, B. Henn, A. Kwa, J. McGibbon, W. A. Perkins, O. Watt-Meyer, C. S. Bretherton, and L. M. Harris, "Correcting a 200 km resolution climate model in multiple climates by machine learning from 25 km resolution simulations," *Journal of Advances in Modeling Earth Systems*, vol. 14, no. 9, p. e2022MS003219, 2022.

[8] A. Kwa, S. K. Clark, B. Henn, N. D. Brenowitz, J. McGibbon, O. Watt-Meyer, W. A. Perkins, L. Harris, and C. S. Bretherton, "Machine-learned climate model corrections from a global storm-resolving model: Performance across the annual cycle," *J. Adv. Model. Earth Syst.*, vol. 15, no. 5, p. e2022MS003400, 2023.

[9] C. H. Sanford, A. Kwa, O. Watt-Meyer, S. K. Clark, N. D. Brenowitz, J. McGibbon, and C. S. Bretherton, "Improving the reliability of ml-corrected climate models with novelty detection," *Authorea Preprints*, 2023.

[10] S. Rasp, M. S. Pritchard, and P. Gentine, "Deep learning to represent subgrid processes in climate models," *Proc. Natl. Acad. Sci. USA*, vol. 115, no. 39, pp. 9684–9689, 2018.

[11] N. D. Brenowitz, T. Beucler, M. Pritchard, and C. S. Bretherton, "Interpreting and stabilizing machine-learning parameterizations of convection," *J. Atmos. Sci.*, vol. 77, no. 12, pp. 4357–4375, 2020.

[12] Y. Han, G. J. Zhang, X. Huang, and Y. Wang, "A moist physics parameterization based on deep learning," *J. Adv. Model. Earth Syst.*, vol. 12, no. 9, p. e2020MS002076, 2020.

[13] J. Ott, M. Pritchard, N. Best, E. Linstead, M. Curcic, and P. Baldi, "A fortran-keras deep learning bridge for scientific computing," 2020.

[14] G. Mooers, M. Pritchard, T. Beucler, J. Ott, G. Yacalis, P. Baldi, and P. Gentine, "Assessing the potential of deep learning for emulating cloud superparameterization in climate models with real-geography boundary conditions," *J. Adv. Model. Earth Syst.*, vol. 13, no. 5, p. e2020MS002385, 2021.

[15] X. Wang, Y. Han, W. Xue, G. Yang, and G. J. Zhang, "Stable climate simulations using a realistic general circulation model with neural network parameterizations for atmospheric moist physics and radiation processes," *Geosci. Model Dev.*, vol. 15, no. 9, pp. 3923–3940, 2022.

[16] F. Iglesias-Suarez, P. Gentine, B. Solino-Fernandez, T. Beucler, M. Pritchard, J. Runge, and V. Eyring, "Causally-informed deep learning to improve climate models and projections," 2023.

[17] J. Yuval and P. A. O'Gorman, "Stable machine-learning parameterization of subgrid processes for climate modeling at a range of resolutions," *Nature Comm.*, vol. 11, no. 1, p. 3295, 2020.

[18] J. Yuval, P. A. O'Gorman, and C. N. Hill, "Use of neural networks for stable, accurate and physically consistent parameterization of subgrid atmospheric processes with good performance at reduced precision," *Geophys. Res. Lett.*, vol. 48, no. 6, p. e2020GL091363, 2021.

[19] S. Rasp, "Coupled online learning as a way to tackle instabilities and biases in neural network parameterizations: general algorithms and lorenz 96 case study (v1. 0)," *Geosci. Model Dev.*, vol. 13, no. 5, pp. 2185–2196, 2020.

[20] K. A. Emanuel, *Atmospheric convection*. 1994.

[21] D. Randall, *Atmosphere, clouds, and climate*, vol. 6. 2012.

[22] A. P. Siebesma, S. Bony, C. Jakob, and B. Stevens, *Clouds and climate: Climate science's greatest challenge*. 2020.

[23] J. W.-B. Lin and J. D. Neelin, "Influence of a stochastic moist convective parameterization on tropical climate variability," *Geophys. Res. Lett.*, vol. 27, no. 22, pp. 3691–3694, 2000.

[24] J. D. Neelin, O. Peters, J. W.-B. Lin, K. Hales, and C. E. Holloway, "Rethinking convective quasi-equilibrium: observational constraints for stochastic convective schemes in climate models," *Phil. Trans. Royal Soc. A*, vol. 366, no. 1875, pp. 2581–2604, 2008.

[25] W. W. Grabowski and P. K. Smolarkiewicz, "Crcp: A cloud resolving convection parameterization for modeling the tropical convecting atmosphere," *Phys. D: Nonlinear Phenom.*, vol. 133, no. 1-4, pp. 171–178, 1999.

[26] J. J. Benedict and D. A. Randall, "Structure of the madden–julian oscillation in the superparameterized cam," *J. Atmos. Sci.*, vol. 66, no. 11, pp. 3277–3296, 2009.

[27] D. A. Randall, "Beyond deadlock," *Geophys. Res. Lett.*, vol. 40, no. 22, pp. 5970–5976, 2013.

[28] W. M. Hannah, C. R. Jones, B. R. Hillman, M. R. Norman, D. C. Bader, M. A. Taylor, L. Leung, M. S. Pritchard, M. D. Branson, G. Lin, *et al.*, "Initial results from the super-parameterized e3sm," *Journal of Advances in Modeling Earth Systems*, vol. 12, no. 1, p. e2019MS001863, 2020.

[29] M. R. Norman, D. C. Bader, C. Eldred, W. M. Hannah, B. R. Hillman, C. R. Jones, J. M. Lee, L. Leung, I. Lyngaas, K. G. Pressel, *et al.*, "Unprecedented cloud resolution in a gpu-enabled full-physics atmospheric climate simulation on olcf's summit supercomputer," *Int. J. High Perform. Compu. Appl.*, vol. 36, no. 1, pp. 93–105, 2022.

[30] D. Randall, M. Khairoutdinov, A. Arakawa, and W. Grabowski, "Breaking the cloud parameterization deadlock," *Bull. Am. Meteorol. Soc.*, vol. 84, no. 11, pp. 1547–1564, 2003.

[31] M. Khairoutdinov, C. DeMott, and D. Randall, "Evaluation of the simulated interannual and subseasonal variability in an amip-style simulation using the csu multiscale modeling framework," *J. Clim.*, vol. 21, no. 3, pp. 413–431, 2008.

[32] P. Pall, M. R. Allen, and D. A. Stone, "Testing the clausius – clapeyron constraint on changes in extreme precipitation under co2 warming," *Clim. Dyn.*, vol. 28, no. 4, pp. 351–363, 2007.

[33] S. B. Guerreiro, H. J. Fowler, R. Barbero, S. Westra, G. Lenderink, S. Blenkinsop, E. Lewis, and X. F. Li, "Detection of continental-scale intensification of hourly rainfall extremes," *Nat. Clim. Change*, vol. 8, no. 9, pp. 803–807, 2018.

[34] J. D. Neelin, C. Martinez-Villalobos, S. N. Stechmann, F. Ahmed, G. Chen, J. M. Norris, Y.-H. Kuo, and G. Lenderink, "Precipitation extremes and water vapor: Relationships in current climate and implications for climate change," *Current Clim. Change Rep.*, vol. 8, no. 1, pp. 17–33, 2022.

[35] F. V. Davenport, M. Burke, and N. S. Diffenbaugh, "Contribution of historical precipitation change to us flood damages," *Proc. Natl. Acad. Sci. USA*, vol. 118, no. 4, p. e2017524118, 2021.

[36] A. G. Pendergrass and D. L. Hartmann, "Two modes of change of the distribution of rain," *J. Clim.*, vol. 27, no. 22, pp. 8357–8371, 2014.

[37] C. Martinez-Villalobos and J. D. Neelin, "Regionally high risk increase for precipitation extreme events under global warming," *Sci. Rep.*, vol. 13, p. 5579, 2023.

[38] G. Behrens, T. Beucler, P. Gentine, F. Iglesias-Suarez, M. Pritchard, and V. Eyring, "Non-linear dimensionality reduction with a variational encoder decoder to understand convective processes in climate models," *J. Adv. Model. Earth Syst.*, vol. 14, no. 8, p. e2022MS003130, 2022.

[39] T. Beucler, M. Pritchard, S. Rasp, J. Ott, P. Baldi, and P. Gentine, "Enforcing analytic constraints in neural networks emulating physical systems," *Phys. Rev. Lett.*, vol. 126, no. 9, p. 098302, 2021.

[40] C. J. Reed, R. Gupta, S. Li, S. Brockman, C. Funk, B. Clipp, K. Keutzer, S. Candido, M. Uyttendaele, and T. Darrell, "Scale-mae: A scale-aware masked autoencoder for multiscale geospatial representation learning," 2023.

[41] J. Pathak, S. Subramanian, P. Harrington, S. Raja, A. Chattopadhyay, M. Mardani, T. Kurth, D. Hall, Z. Li, K. Azizzadenesheli, P. Hassanzadeh, K. Kashinath, and A. Anandkumar, "Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators," 2022.

[42] B. Bonev, T. Kurth, C. Hundt, J. Pathak, M. Baust, K. Kashinath, and A. Anandkumar, "Spherical fourier neural operators: Learning stable dynamics on the sphere," in *Proc. ICLR*, 2023.

[43] R. Lam, A. Sanchez-Gonzalez, M. Willson, P. Wirnsberger, M. Fortunato, A. Pritzel, S. Ravuri, T. Ewalds, F. Alet, Z. Eaton-Rosen, W. Hu, A. Merose, S. Hoyer, G. Holland, J. Stott, O. Vinyals, S. Mohamed, and P. Battaglia, "Graphcast: Learning skillful medium-range global weather forecasting," 2022.

[44] E. M. Fischer, S. I. Seneviratne, P. L. Vidale, D. Lüthi, and C. Schär, "Soil moisture–atmosphere interactions during the 2003 european summer heat wave," *J. Clim.*, vol. 20, no. 20, pp. 5081–5099, 2007.

[45] S. I. Seneviratne, T. Corti, E. L. Davin, M. Hirschi, E. B. Jaeger, I. Lehner, B. Orlowsky, and A. J. Teuling, "Investigating soil moisture–climate interactions in a changing climate: A review," *Earth-Sci. Rev.*, vol. 99, no. 3-4, pp. 125–161, 2010.

[46] P. Wang, J. Yuval, and P. A. O'Gorman, "Non-local parameterization of atmospheric subgrid processes with neural networks," *J. Adv. Model. Earth Syst.*, vol. 14, no. 10, p. e2022MS002984, 2022.

[47] B. Lütjens, C. H. Crawford, C. D. Watson, C. Hill, and D. Newman, "Multiscale neural operator: Learning fast and grid-independent pde solvers," 2022.

[48] W. M. Hannah, K. G. Pressel, M. Ovchinnikov, and G. S. Elsaesser, "Checkerboard patterns in e3smv2 and e3sm-mmfv2," *Geosci. Model Dev.*, vol. 15, no. 9, pp. 6243–6257, 2022.

[49] W. M. Hannah, A. M. Bradley, O. Guba, Q. Tang, J.-C. Golaz, and J. Wolfe, "Separating physics and dynamics grids for improved computational efficiency in spectral element earth system models," *J. Adv. Model. Earth Syst.*, vol. 13, no. 7, p. e2020MS002419, 2021.

[50] I. Osband, J. Aslanides, and A. Cassirer, "Randomized prior functions for deep reinforcement learning," 2018.

[51] Y. Yang, G. Kissas, and P. Perdikaris, "Scalable uncertainty quantification for deep operator networks using randomized priors," *Comput. Methods Appl. Mech. Eng.*, vol. 399, p. 115399, 2022.

[52] M. A. Bhouri, M. Joly, R. Yu, S. Sarkar, and P. Perdikaris, "Scalable bayesian optimization with high-dimensional outputs using randomized prior networks," 2023.

[53] S. R. Cachay, V. Ramesh, J. N. S. Cole, H. Barker, and D. Rolnick, "Climart: A benchmark dataset for emulating atmospheric radiative transfer in weather and climate models," 2021.

[54] T. Beucler, M. Pritchard, J. Yuval, A. Gupta, L. Peng, S. Rasp, F. Ahmed, P. A. O'Gorman, J. D. Neelin, N. J. Lutsko, and P. Gentine, "Climate-invariant machine learning," 2021.

[55] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Fourier neural operator for parametric partial differential equations," 2021.

[56] Z. Li, H. Zheng, N. Kovachki, D. Jin, H. Chen, B. Liu, K. Azizzadenesheli, and A. Anandkumar, "Physics-informed neural operator for learning partial differential equations," 2023.

[57] J. Ling, A. Kurzawski, and J. Templeton, "Reynolds averaged turbulence modelling using deep neural networks with embedded invariance," *J. Fluid Mech.*, vol. 807, pp. 155–166, 2016.

[58] J. F. MacArt, J. Sirignano, and J. B. Freund, "Embedded training of neural-network subgrid-scale turbulence models," *Phys. Rev. Fluids*, vol. 6, no. 5, p. 050502, 2021.

[59] V. Xing, C. Lapeyre, T. Jaravel, and T. Poinsot, "Generalization capability of convolutional neural networks for progress variable variance and reaction rate subgrid-scale modeling," *Energies*, vol. 14, no. 16, p. 5096, 2021.

[60] M. P. Brenner, J. D. Eldredge, and J. B. Freund, "Perspective on machine learning for advancing fluid mechanics," *Phys. Rev. Fluids*, vol. 4, p. 100501, 2019.

[61] A. Subramaniam, M. L. Wong, R. D. Borker, S. Nimmagadda, and S. K. Lele, "Turbulence enrichment using physics-informed generative adversarial networks," 2020.

[62] B. Kim, V. C. Azevedo, N. Thuerey, T. Kim, M. Gross, and B. Solenthaler, "Deep fluids: A generative network for parameterized fluid simulations," *Comput. Graph. Forum*, vol. 38, no. 2, pp. 59–70, 2019.

[63] J. W.-B. Lin and J. D. Neelin, "Toward stochastic moist convective parameterization in general circulation models," *Geophys. Res. Lett.*, vol. 30 (4), p. 1162, 2003.

[64] P. A. O'Gorman, "Precipitation extremes under climate change," *Current Clim. Change Rep.*, vol. 1, pp. 49–59, 2015.

[65] L. Zanna and T. Bolton, "Data-driven equation discovery of ocean mesoscale closures," *Geophys. Res. Lett.*, vol. 47, no. 17, p. e2020GL088376, 2020.

[66] A. Grundner, T. Beucler, P. Gentine, and V. Eyring, "Data-driven equation discovery of a cloud cover parameterization," 2023.

ClimSim: Supplementary Information

Sungduk Yu[1]*, **Walter M. Hannah**[2], **Liran Peng**[1], **Mohamed Aziz Bhouri**[3], **Ritwik Gupta**[4]
Jerry Lin[1], **Björn Lütjens**[5], **Justus C. Will**[1], **Tom Beucler**[6], **Bryce E. Harrop**[7],
Benjamin R. Hilman[8], **Andrea M. Jenney**[1,9], **Savannah L. Ferretti**[1], **Nana Liu**[1],
Anima Anandkumar[10], **Noah D. Brenowitz**[10], **Veronika Eyring**[11,12], **Pierre Gentine**[3],
Stephan Mandt[1], **Jaideep Pathak**[10], **Carl Vondrick**[3], **Rose Yu**[13], **Laure Zanna**[14],
Ryan P. Abernathey[3], **Fiaz Ahmed**[15], **David C. Bader**[2], **Pierre Baldi**[1], **Elizabeth A. Barnes**[16],
Gunnar Behrens[11], **Christopher S. Bretherton**[17], **Julius J. M. Busecke**[3], **Peter M. Caldwell**[2],
Wayne Chuang[3], **Yilun Han**[18], **Yu Huang**[3], **Fernando Iglesias-Suarez**[11], **Sanket Jantre**[19],
Karthik Kashinath[10], **Marat Khairoutdinov**[20], **Thorsten Kurth**[10], **Nicholas J. Lutsko**[13],
Po-Lun Ma[7], **Griffin Mooers**[1], **J. David Neelin**[15], **David A. Randall**[16], **Sara Shamekh**[3]
Akshay Subramaniam[10], **Mark A. Taylor**[8], **Nathan M. Urban**[18], **Janni Yuval**[5]
Guang J. Zhang[12], **Tian Zheng**[3], **Michael S. Pritchard**[1,10]
[1]UCI, [2]LLNL, [3]Columbia, [4]UCB, [5]MIT, [6]UNIL, [7]PNNL, [8]SNL, [9]OSU, [10]NVIDIA
[11]DLR, [12]IUP, [13]UCSD, [14]NYU, [15]UCLA, [16]CSU, [17]Allen AI, [18]Tsinghua, [19]BNL, [20]SUNY

Contents

*Corresponding author: sungduk@uci.edu

Preprint. Under review.

1 Climate Simulations

Climate models divide the Earth's atmosphere, land surface, and ocean into a three-dimensional (3D) grid, creating a discretized representation of the planet. Somewhat like a virtual Lego construction of Earth, with each brick representing a small region (grid cell). Earth system models are made up of independent component models for the atmosphere, land surface, rivers, ocean, sea ice, and glaciers. Each of these component models is developed independently and can run by itself when provided with the appropriate input data. When running as a fully coupled system the "component coupler" handles the flow of data between the components.

Within each grid cell of the component models, a series of complex calculations are performed to account for various physical processes, such as phase changes of water, radiative heat transfer, and dynamic transport (referred to as "advection"). Each component model uses the discretized values of many quantities (such as temperature, humidity, and wind speed) as inputs to parameterizations and fluid solvers to output those same values for a future point in time.

The atmosphere and ocean components are the most expensive pieces of an Earth system model, which is largely due to the computation and inter-process communication associated with their fluid dynamics solvers. Furthermore, a significant portion of the overall cost is attributed to the the atmospheric physics calculations that are performed locally within each grid column. It is important to note that atmospheric physics serves as a major source of uncertainty in climate projections,

primarily stemming from the challenges associated with accurately representing cloud and aerosol processes.

1.1 Model Description

The data that comprise ClimSim are from simulations with the Energy Exascale Earth System Model-Multiscale Modeling Framework version 2.1.0 (E3SM-MMF v2) [1]. Traditionally, global atmospheric models parameterize the clouds and turbulence using crude, low-order models that attempt to represent the aggregate effects of these processes on larger scales. However, the complexity and nonlinearity of cloud and rainfall processes makes them particularly challenging to represent accurately with parameterizations. The MMF approach replaces these conventional parameterizations with a "Cloud Resolving Model" (CRM) in each cell of the global grid, such that a range of clouds and turbulence can be explicitly represented. Each of these independent CRMs are spatially fixed and exchange coupling tendencies with a parent global grid column. This novel approach to representing clouds and turbulence can improve various aspects of the simulated climate, such as rainfall patterns [2].

The configuration of E3SM-MMF used here shares some details with E3SMv2. The dynamical code of E3SM uses a spectral element approach on a cubed-sphere geometry. Physics calculations are performed on an unstructured, finite-volume grid that is slightly coarser than the dynamics grid, following Hannah et al. (2021) [3], which is better aligned with the effective resolution of the dynamics grid. Cases with realistic topography include an active land model component that responds to atmospheric conditions with the appropriate fluxes of heat and momentum.

The embedded CRM in E3SM-MMF is adapted from the System for Atmospheric Modeling (SAM) described by Khairoutdinov and Randall (2003) [4]. While the CRM does explicitly represent clouds and turbulence, it still cannot represent the smallest scales of turbulence and microphysics, and, therefore, these processes still need to be parameterized within each CRM grid cell. Microphysical processes are parameterized with a single-moment scheme, and sub-grid scale turbulent fluxes are parameterized using a diagnostic Smagorinsky-type closure. Convective momentum transport in the nested CRM is handled using the scalar momentum tracer approach of Tulich (2015) [5]. The CRM uses an internal time step of 10 seconds, while the global calculations use a time step of 20 minutes.

Despite recent efforts to accelerate E3SM-MMF with GPUs and algorithmic techniques [6], the domain size of the CRM strongly affects the computational throughput and limits the type of experiments that can be conducted. However, the MMF approach is quite flexible in how the CRM size is specified. E3SM-MMF is typically run with a two-dimensional (2D) CRM that neglects one of the horizontal dimensions, and employs relatively coarse grid spacing that cannot represent small clouds. Increasing the size of this 2D domain by adding further columns (more CRM cells) generally improves the realism of the model solution. Reducing the model grid spacing can also improve the model to a certain degree, although the number of columns often needs to be increased to avoid the degradation associated with a small CRM. Ideally, the CRM would always be used in a 3D configuration to fully capture the complex, chaotic turbulence that dictates the life cycle of each individual cloud, but this approach is generally limited to special experiments that can justify the extra computational cost. The simulations for ClimSim utilize a 2D CRM with 64 columns and 2-km horizontal grid spacing within each grid cell of the global grid.

The atmospheric component of E3SM uses a hybrid vertical grid that is "terrain-following" near the surface, and transitions to be equivalent to pressure levels near the top (e.g., https://www2.cesm.ucar.edu/models/atm-cam/docs/usersguide/node25.html). The vertical levels are specified to be thin near the surface to help capture turbulent boundary layer processes, and are gradually stretched to be very coarse in the stratosphere. E3SM-MMF uses 60 levels for the global

dynamics with a top level around 65-km. The CRM used for atmospheric physics uses 50 levels, ignoring the upper 10 levels, to avoid problems that arise from using the anelastic approximation with very low densities. This does not create any issues, because cloud processes are generally confined to the troposphere where the anelastic approach is valid. The hybrid grid can be converted to pressure levels using equation (1), where $P_0 = 100,000$ Pa is a reference pressure, $P_s(\mathbf{x}, t)$ is the surface pressure, which varies in location $\mathbf{x}$ and time t. A_k and B_k—where the subscript k denotes the index of vertical coordinate—are the fixed, prescribed coefficients that define how the "terrain-following" and "pure pressure" coordinates are blended to define the hybrid coordinate at each vertical level.

$$P_k = A_k P_0 + B_k P_s \tag{1}$$

A_k and B_k are provided as a part of the dataset with variable names of "hyam" and "hyai" or "hybm" and "hybi", depending on whether mid-level or interface values are needed. The third character of the variable names, "a" and "b", denotes A_k and B_k coefficients, respectively. Note that the indexing of the vertical coordinate starts from the top of the atmosphere due to the construct of A_k and B_k coefficients, e.g., $k = 0$ for the top and $k = 59$ for the surface in E3SM-MMF.

In the E3SM-MMF framework, the sequencing of atmospheric processes can be conceptualized as progressively. It starts with a surface coupling step that receives fluxes from the surface component models (i.e., land, ocean, and sea ice). This is followed by a set of relatively inexpensive physics parameterizations that handle processes such as airplane emissions, boundary layer mixing, and unresolved gravity waves. The global dynamics then takes over to evolve the winds and advect tracers on the global grid. Finally, there is another set of physics calculations to handle clouds, chemistry, and radiation, which are relatively expensive. This final physics section is where the embedded CRM of E3SM-MMF is used, and is the ideal target for surrogate model emulation due to its outsized computational expense. Accordingly, this step represents the target of ClimSim.

One area where E3SM-MMF significantly differs from E3SMv2 is in the treatment of aerosols and chemistry. The embedded CRM in E3SM-MMF predicts the mass of water species (i.e., cloud and rain droplet mass mixing ratios) but does not predict the number concentration (i.e., number of drops per mass of air). One consequence of this limitation is that E3SM-MMF cannot represent complex cloud aerosol interactions that can impact droplet number concentrations and cloud radiative feedbacks. Therefore, E3SM-MMF cannot use the more sophisticated aerosol and chemistry package used by E3SMv2, and instead uses prescribed aerosol and ozone amounts to account for the direct radiative impact of these tracers. Current efforts are addressing this limitation for future versions of E3SM-MMF.

1.2 Model Configurations

The simulations used for ClimSim were performed on the NERSC Perlmutter machine. E3SM-MMF is unique among climate models in that it can leverage hybrid CPU/GPU architectures on machines such as NERSC Perlmutter (https://www.nersc.gov/systems/perlmutter), which has 4 NVIDIA A100 GPUs per node. All simulations were configured to run with 4 MPI ranks and 16 OpenMP threads per node. The low-resolution (real geography and aquaplanet) cases used 2 nodes, and the high-resolution (real geography) case used 32 nodes. The throughput of these configurations was roughly 11.5 simulated years per day (sypd) for low-resolution cases and 3.3 sypd for the high-resolution case, averaged over multiple batch submissions. The simulation length in all cases was 10 model years and 2 model months.

Boundary conditions over maritime regions are constrained by prescribed sea surface temperatures and sea ice amount. Various input data are needed for the cases with realistic topography, such as ozone

concentrations and sea surface temperatures, which have been generated to reflect a climatological average of the 2005-2014 period. The aquaplanet configuration does not have a land component, but otherwise has similar input requirements using idealized data to produce a climate that is symmetric along lines of constant latitude.

2 Dataset and Code Access

2.1 Code Access

We have uploaded our datasets to Hugging Face:

- E3SM-MMF High-Resolution Real Geography dataset: https://huggingface.co/datasets/LEAP/ClimSim_high-res,
- E3SM-MMF Low-Resolution Real Geography dataset: https://huggingface.co/datasets/LEAP/ClimSim_low-res
- E3SM-MMF Low-Resolution Aquaplanet dataset: https://huggingface.co/datasets/LEAP/ClimSim_low-res_aqua-planet

We have documented all code (including the code to preprocess the data, create, train, and evaluate the baseline models, and visualize data and metrics) in an openly-available GitHub repository: https://leap-stc.github.io/ClimSim.

2.2 Variable List

All variables included in our dataset are listed in Table 1.

2.3 Dataset Statistics

Here, we present some distribution statistics to aid in understanding the dataset. Detailed distributions for all variables are provided in https://github.com/leap-stc/ClimSim/tree/main/variable_distributions. These statistics are calculated for each vertical level individually for the vertically-resolved variables (e.g., state_t and state_q0001). For each variable (additionally, at each level for the vertically-resolved variables), a histogram is provided to visualize the distribution using 100 bins. Additionally, a text file accompanies each histogram, containing key statistical measures such as the mean, standard deviation, skewness, kurtosis, median, deciles, quartiles, minimum, maximum, and mode. The text file also includes the bin edges and the corresponding frequency values used to generate the histogram figures. This comprehensive approach allows for a detailed analysis of the dataset's distributions.

3 Baseline Models

3.1 Multilayer Perceptron (MLP)

A multilayer perceptron (MLP) is a basic, densely connected artificial neural network. We used KerasTuner [7] with a random search algorithm for hyperparameter optimization. The following hyperparameters are optimized: the number of hidden layers, the number of nodes per layer, activation function and batch size. The search domains were:

- The number of hidden layers (N_{layers}): [3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]
- The number of nodes per layer (N_{nodes}): [128, 256, 384, 512, 640, 768, 896, 1024]

In	Out	Variable	Dimensions	Units	Description
×		pbuf_SOLIN	ncol	W/m^2	Solar insolation
×		pbuf_COSZRS	ncol		Cosine of solar zenith angle
×		pbuf_LHFLX	ncol	W/m^2	Surface latent heat flux
×		pbuf_SHFLX	ncol	W/m^2	Surface sensible heat flux
×		pbuf_TAUX	ncol	W/m^2	Zonal surface stress
×		pbuf_TAUY	ncol	W/m^2	Meridional surface stress
×		pbuf_ozone	lev, ncol	mol/mol	Ozone volume mixing ratio
×		pbuf_N2O	lev, ncol	mol/mol	Nitrous oxide volume mixing ratio
×		pbuf_CH4	lev, ncol	mol/mol	Methane volume mixing ratio
×		state_ps	ncol	Pa	Surface pressure
×	×	state_q0001	lev, ncol	kg/kg	Specific humidity
×	×	state_q0002	lev, ncol	kg/kg	Cloud liquid mixing ratio
×	×	state_q0003	lev, ncol	kg/kg	Cloud ice mixing ratio
×	×	state_t	lev, ncol	K	Air temperature
×	×	state_u	lev, ncol	m/s	Zonal wind speed
×	×	state_v	lev, ncol	m/s	Meridional wind speed
×		state_pmid	lev, ncol	Pa	Mid-level pressure
×		cam_in_ASDIR	ncol		Albedo for direct shortwave radiation
×		cam_in_ASDIF	ncol		Albedo for diffuse shortwave radiation
×		cam_in_ALDIR	ncol		Albedo for direct longwave radiation
×		cam_in_ALDIF	ncol		Albedo for diffuse longwave radiation
×		cam_in_LWUP	ncol	W/m^2	Upward longwave flux
×		cam_in_SNOWHLAND	ncol	m	Snow depth over land (liquid water equivalent)
×		cam_in_SNOWHICE	ncol	m	Snow depth over ice
×		cam_in_LANDFRAC	ncol		Land area fraction
×		cam_in_ICEFRAC	ncol		Sea-ice area fraction
	×	cam_out_NETSW	ncol	W/m^2	Net shortwave flux at surface
	×	cam_out_FLWDS	ncol	W/m^2	Downward longwave flux at surface
	×	cam_out_PRECSC	ncol	m/s	Snow rate (liquid water equivalent)
	×	cam_out_PRECC	ncol	m/s	Rain rate
	×	cam_out_SOLS	ncol	W/m^2	Downward visible direct solar flux to surface
	×	cam_out_SOLL	ncol	W/m^2	Downward near-IR direct solar flux to surface
	×	cam_out_SOLSD	ncol	W/m^2	Downward visible diffuse solar flux to surface
	×	cam_out_SOLLD	ncol	W/m^2	Downward near-IR diffuse solar flux to surface

Table 1: Overview of input (1^{st}) and output variables (2^{nd} column) of the E3SM-MMF physics calculations (including the CRM) that are stored in ClimSim. The other columns indicate for each variable its name, dimension, unit and a brief description. IR is short for infrared, which is also often referred to as "longwave" radiation among atmospheric scientists.

- Activation function: [ReLU, LeakyReLU (alpha=.15), ELU (alpha=1.)]
- Batch size: [48, 96, 192, 384, 768, 1152, 1536, 2304, 3072]
- Optimizer: [Adam, RAdam, RMSprop, SGD]

Note that N_{nodes} was selected independently for each hidden layer. For example, for $N_{layers} = k$, N_{nodes} was drawn from the search domain k times. The width of the last hidden layer was fixed at 128, and the output layer utilized two different activation functions: linear activation for the first 120 outputs (corresponding to the heating and moistening tendencies) and ReLU activation for the rest 8 variables (corresponding positive-definite surface variables). The rest of the hyperparameters are fixed as follows: loss function = mean squared errors; learning rate = a cyclic schedule (minimum: 0.00025, maximum: 0.0025, step size: 4 epochs).

Following [8], we conducted the hyperparameter search in two stages. In the first stage, a total of 8,257 randomly-drawn hyperparameter configurations were trained and evaluated with a tiny subset of the full training set, subsampled in the time dimension with a stride of 37. In the second stage, the top 0.2% candidates (160 hyperparamteter configurations) were retrained with a larger fraction of the full training set (subsampled with a stride of 7) and then evaluated for our MLP baseline. After this two-step search process, the best hyperparameter configuration we identified was: N_{layers} = 5, N_{nodes} = [768 640 512 640 640], Activation function = LeakyReLU, Batch size = 3072, and optimizer = RAdam. The model has 1,753,472 parameters and executes 3.50 MFlops on one data point. The architecture of our MLP baseline is summarized in Figure 1.

3.2 Randomized Prior Network (RPN)

A Randomized Prior Network (RPN) is an ensemble model [9]. Each member of the RPNs is built as the sum of a trainable surrogate model and a non-trainable (so-called prior) surrogate model; in this example MLP for simplicity. Multiple replicas of the networks are constructed by independent and random sampling of both trainable and non-trainable parameters [10, 11]. RPNs also resort to data bootstrapping in order to mitigate the uncertainty collapse of the ensemble method when tested beyond the training data points [11]. Data bootstrapping consists of sub-sampling and randomization of the data each network in the ensemble sees during training. Hyperparameters of individual MLPs do not need to be tuned from scratch and are chosen based on the hyperparameter search mentioned in Section 3.1. RPN ensembles of 128 networks are considered as justified in [10].

In particular, individual MLPs forming RPN are considered as fully connected neural networks with 5 hidden layers of 768, 640, 512, 640, and 640 neurons for a total of $1,736,960 \sim 1.7 \times 10^6$ parameters per MLP, and $222,330,880 \sim 2.2 \times 10^8$ parameters for the full RPN ensemble model. The full RPN ensemble model executes 0.89 GFlops on one data point. Activation functions are taken as RELU with a negative slope of 0.15 except for the output layer whose activation function are chosen as identity. The batch size is taken equal to 3072. MLPs are trained for a total of $13,140$ stochastic gradient descent steps using the ADAM optimizer. Learning rates are initialized at 5×10^{-4} with an exponential decay at a rate of 0.99 for every $1,000$ steps. Finally, data bootstrapping is performed with a fraction of 80% as justified in [10, 11].

3.3 Convolutional Neural Network (CNN)

The CNN used is a modified ResNet-style network as displayed in Figure 2. Each ResNet block is composed of two 1D convolutions with a 3x3 kernel using "same" padding with an output feature map size of 406. Each Conv1D is followed by a ReLU activation and dropout (with rate 0.175). Residuals are also 1D convolved using a 1x1 kernel and added back to the output of the main ResNet block.

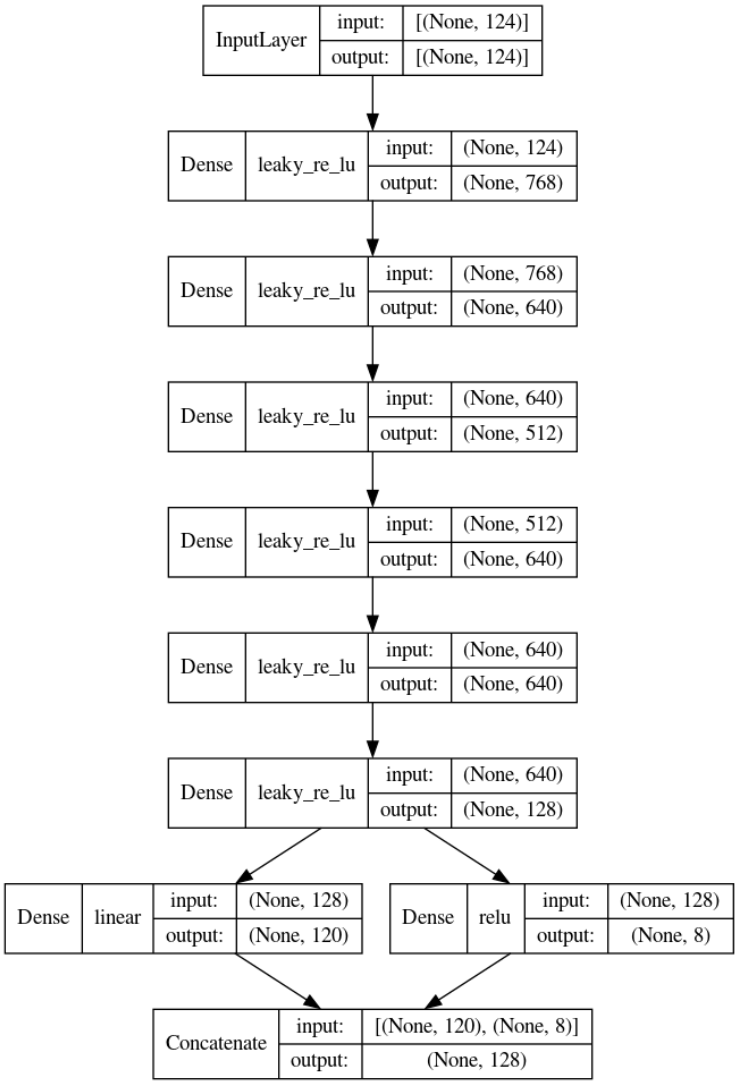

Figure 1: The architecture of the MLP baseline.

The CNN composes 12 such ResNet blocks, followed by "flattening" of the feature map via a 1x1 convolution and an ELu activation. Two separate Dense layers (and their corresponding activations) map the output feature map to their respective co-domains: one to $(-\infty, \infty)$ assuming that vertically resolved variables have no defined range, and another to $[0, \infty)$ for all globally resolved variables. These are concatenated as the output of the network. The network is visualized below in Figure 2.

In sum, the model has 13,215,420 parameters and executes 1.59 GFlops on one data point.

The network is trained for 10 epochs with an Adam optimizer with standard hyperparameters ($\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1 \times 10^{-7}$). The learning rate is defined using a cyclic learning rate scheduler with an initial learning rate of 1×10^{-4}, a maximum learning rate of 1×10^{-3}, and a step size of $2 \times \lfloor \frac{10{,}091{,}520}{12} \rfloor$. A scaling function of $\frac{1}{2.0^{x-1}}$ is applied to the scheduler per step x.

A weighted mean absolute error is used as the loss function for optimization. We downweight the standard MAE loss to de-emphasize repeated scalar values provided to the network as input. The weighted MAE function is defined below.

```
def mae_adjusted(y_true, y_pred):
```

```
    ae = K.abs(y_pred - y_true)
    vertical_weights = K.mean(ae[:,:,0:2]) * (120/128)
    scalar_weights = K.mean(ae[:,:,2:10]) * (8/128)
    return vertical_weights + scalar_weights
```

A hyper-parameter search was conducted on depth, width, kernel size, activation functions, loss functions, and normalization types before settling on the hyper-parameters defined above. The search was executed using the Hyperband [12] strategy with the Keras Tuner framework. The search space was as follows:

- Model depth/number of ResNet blocks: [2, 15]
- Model width: [32, 512]
- Kernel width: 3, 5, 7, 9
- Activation: GeLU, ELu, ReLU, Swish
- Layer normalization: True/False
- Droput: [0.0, 0.5]
- Optimizer: SGD, Adam

Hyperparameter search was conducted for 12 hours on 8 NVIDIA Tesla V100 32GB cards with one model executing on each card.

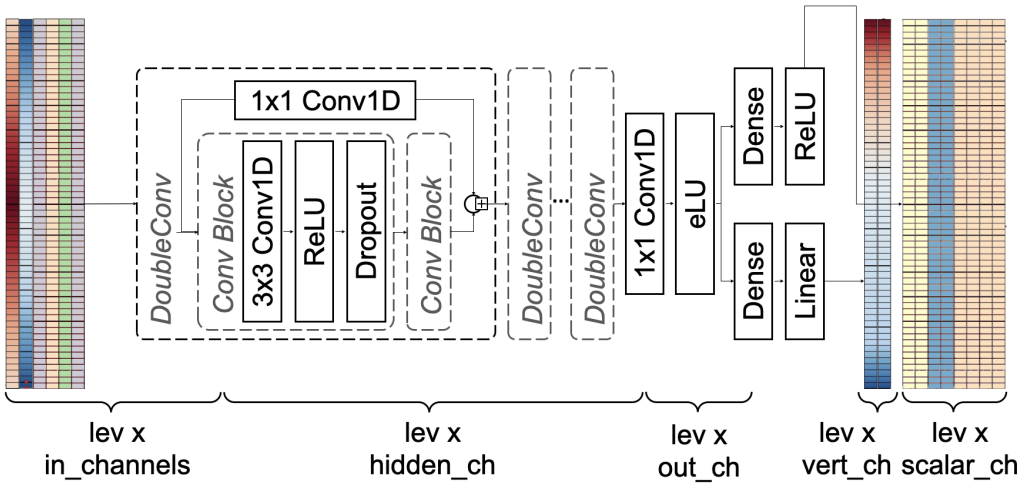

Figure 2: The ResNet-style CNN baseline is comprised of multiple ResNet blocks, called DoubleConv, and applies different activations to the outputs for vertically-resolved and global variables. The channel dimensions are (in_channels, hidden_ch, out_ch, vert_ch, scalar_ch) = $(6, 406, 10, 8, 2)$.

3.4 Heteroskedastic Regression (HSR)

We quantify the inherent stochasticity in the data $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$ and the uncertainty in our prediction by providing a distributional prediction instead of a point estimate. In Hetereoskedastic Regression, we model this predictive distribution explicitly, here as independent Gaussians with unique mean μ_k and precision (inverse variance) τ_k for each variable. Specifically, we assume

$$\mathbf{y}_i \,|\, \mathbf{x}_i \sim \mathcal{N}(\mu(\mathbf{x}_i), \mathrm{Diag}(\tau(\mathbf{x}_i)^{-1}))$$

and parameterize both μ and τ as over-parameterized feed-forward neural networks (MLPs) $\hat{\mu}_\theta(\mathbf{x})$ and $\hat{\tau}_\phi(\mathbf{x})$, respectively. This yields the corresponding predictive distribution

$$\hat{\mathbf{y}}_i \,|\, \mathbf{x}_i \sim \mathcal{N}(\hat{\mu}_\theta(\mathbf{x}_i), \mathrm{Diag}(\hat{\tau}_\theta(\mathbf{x}_i)^{-1})),$$

which can be fitted with maximum likelihood estimation (MLE) by minimizing the objective

$$\mathcal{L}(\theta, \phi) = \frac{1}{2n} \sum_{i=1}^{n} \left[\|\hat{\tau}_\phi(\mathbf{x}_i) \left(\mathbf{y}_i - \hat{\mu}_\theta(\mathbf{x}_i)\right)\|_2^2 - \mathbf{1}^T \log\left(\hat{\mu}_\theta(\mathbf{x}_i)\right) \right].$$

Note that, due to the flexibility of the neural networks, this formulation is ill-posed as it may lead to cases of extreme overfitting where $\hat{\tau}_\phi(\mathbf{x}_i) \approx \mathbf{y}_i$, $\hat{\tau}_\phi(\mathbf{x}_i) \approx \mathbf{0}$, and $\mathcal{L}(\theta, \phi)$ thus becomes completely unstable. Hence, we instead minimize a modified objective that includes L2-regularization via

$$\mathcal{L}_{\rho,\gamma}(\theta, \phi) \coloneqq \rho \mathcal{L}(\theta, \phi) + (1 - \rho) \left[\gamma \|\theta\|_2^2 + (1 - \gamma) \|\phi\|_2^2 \right],$$

where $\rho, \gamma \in (0, 1)$ determine the trade-off between MLE estimation, mean regularization, and precision regularization. For our models we set $\rho = 1 - \gamma$ to reduce the hyperparameter search space.

Specifically, we use two MLPs with Layer Normalization and ReLU activation function and train them with gradient-based stochastic optimization. To improve stability, the first third of training is spent on exclusively training $\hat{\mu}_\theta(\mathbf{x}_i)$ with an MSE loss. To optimize hyperparameters we select a configuration from 300 trials with random number of hidden layers n_l (2, 3, or 4), size of the hidden layers n_h (256, 512, 1024, or 2048), γ (log-uniform in (0.001, 0.1)), type of optimizer (SGD or Adam with $\beta_1 = 0.9$, $\beta_2 = 0.999$), learning rate λ (log-uniform in (10^{-6}, 10^{-3})) and batch size n_b (1024, 2048, 4096, 8192, or 16384). Each run was trained for 12 epochs total on one NVIDIA GeForce RTX 4080 16GB. We chose the run with the lowest CRPS on the validation data, yielding $n_l = 4$, $n_h = 1024$, $\gamma = 2.2 \times 10^{-2}$, $\lambda = 7 \times 10^{-6}$, $n_b = 16384$, trained with Adam. This final model has 6.83M parameters and executes 6.85 MFlops per data point.

3.5 Conditional Variational Autoencoder (cVAE)

A conditional generative latent variable model first samples, from a prior $p(\mathbf{z})$, a point $\mathbf{z}$ in a low-dimensional latent space, which then informs a conditional distribution $p_\theta(\mathbf{y}|\mathbf{z}, \mathbf{x})$ over the target domain. This allows for a complex and flexible predictive distribution. Specifically, we use feed-forward neural networks (MLPs) $\mu_\theta(\mathbf{z}, \mathbf{x})$ and $\sigma_\theta(\mathbf{z}, \mathbf{x})$ with combined parameters θ and model:

$$\begin{aligned} \mathbf{z} &\sim \mathcal{N}(\mathbf{0}, \mathcal{I}) \\ \mathbf{y}|\mathbf{z}, \mathbf{x} &\sim \mathcal{N}\big(\mu_\theta(\mathbf{z}, \mathbf{x}), \mathrm{Diag}(\sigma_\theta(\mathbf{x})^2)\big). \end{aligned} \tag{2}$$

To fit the model to data $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$, we minimize the negative evidence lower bound (NELBO) $\mathcal{L}_\theta(\mathbf{q})$ that bounds the intractable negative marginal likelihood from above via

$$\mathcal{L}_\theta(q) \coloneqq -\mathbb{E}_{\mathbf{z}_i \sim q} \left[\log \frac{p_\theta(\mathbf{y}_i, \mathbf{z}_i|\mathbf{x}_i)}{q(\mathbf{z}_i|\mathbf{x}_i)} \right] = -\log p_\theta(\mathbf{y}_i|\mathbf{x}_i) + \underbrace{\mathrm{KL}\big(q \,\|\, p_\theta(\mathbf{z}_i|\mathbf{y}_i, \mathbf{x}_i)\big)}_{\geq 0},$$

using an approximation q to the posterior $p_\theta(\mathbf{z}_i|\mathbf{y}_i, \mathbf{x}_i)$. (Conditional) Variational Autoencoder [13] use amortized variational inference to optimize θ and q jointly by approximating the latter with e.g. $q_\psi(\mathbf{z}_i) = \mathcal{N}\big(g_\psi(\mathbf{x}_i), \mathrm{Diag}(h_\psi(\mathbf{x}_i)^2)\big)$, where we again choose $g_\psi(\mathbf{x}_i)$ and $h_\psi(\mathbf{x}_i)$ to be MLPs. This

allows us to and optimize for θ and ψ by minimizing

$$\mathcal{L}_\theta(q) \overset{\beta=1}{=} \mathbb{E}_{\mathbf{z}_i \sim q_\psi} \left[\frac{1}{2} \left\| \frac{\mathbf{y}_i - \mu_\theta(\mathbf{z}_i, \mathbf{x}_i)}{\sigma_\theta(\mathbf{z}_i, \mathbf{x}_i)} \right\|_2^2 + \mathbf{1}^T \log\left(\sigma_\theta(\mathbf{z}_i, \mathbf{x}_i)\right) \right] + \beta \mathrm{KL}(q_\psi(\mathbf{z}_i) \,\|\, p(\mathbf{z}_i)) + \text{const}$$

with a Monte Carlo approximation by first sampling $\mathbf{z}_i$ (once) from the variational encoder $q_\psi(\mathbf{z}_i)$ after which we decode the predictive mean and standard deviation with the variational decoder $\mu_\theta(\mathbf{z}, \mathbf{x})$ and $\sigma_\theta(\mathbf{z}, \mathbf{x})$. We then compute the NELBO as a sum of a reconstruction term and a KL term that regularizes the latent space, average over all samples, and finally back-propagate the gradients. By letting β be a hyperparameter, we can manually determine the trade off between reconstruction quality and latent space structure. Finally, at inference time, we can use (2) to sample from the predictive distribution

$$p_\theta(\hat{\mathbf{y}}|\mathbf{x}) = \int p_\theta(\hat{\mathbf{y}}|\mathbf{x}, \mathbf{z}) p(\mathbf{z}) \, d\mathbf{z}.$$

Specifically, for both the variational encoder and decoder we use a MLP with Layer Normalization, ReLU activation function, dropout with $p = 0.05$, and two branching final layers that produce the mean and standard deviation respectively. We train both MLPs jointly, with gradient-based stochastic optimization, on the objective described above. To explore the hyperparameter space we run 300 trials with random number of hidden layers n_l (2, 3, or 4), size of the hidden layers n_h (256, 512, 1024, or 2048), size of the latent space (4, 8, 16, or 32), β (log-uniform in (0.01, 10)), type of optimizer (SGD or Adam with $\beta_1 = 0.9$, $\beta_2 = 0.999$), learning rate λ (log-uniform in $(10^{-6}, 10^{-3})$), L2 regularization α (log-uniform in $(10^{-6}, 10^{-3})$), and batch size n_b (1024, 2048, 4096, 8192, or 16384). Each run was trained for 5 epochs total on one NVIDIA GeForce RTX 4080 16GB. The run with the lowest CRPS on the validation data yielded $n_l = 3$, $n_h = 1024$, $n_b = 4096$, trained with Adam. In a second step, we fix these hyperparameters and further fine-tune β, λ, and α, now training for 20 epochs every time, for 10 trials. We find the best model with $\beta = 0.5$, $\lambda = 5 \times 10^{-5}$, $\alpha = 10^{-3}$. This model has 4.90M parameters and executes 4.88 MFlops per data point.

4 Baseline Model Evaluations

4.1 Metrics

4.1.1 Deterministic Metrics

Mean Absolute Error (MAE):

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |X_i - y| \tag{3}$$

Root Mean Squared Error (RMSE):

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (X_i - y)^2} \tag{4}$$

Coefficient of Determination (R^2):

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (X_i - y)^2}{\sum_{i=1}^{n} (X_i - \bar{X})^2} \tag{5}$$

where X_i and y represent the true and predicted values, respectively. The mean of the true values of the dependent variable is denoted by $\bar{X}$.

4.1.2 Stochastic Metric: CRPS

The Continuous Ranked Probability Score (CRPS) is a generalization of the Mean Absolute Error (MAE) for distributional predictions. CRPS penalizes over-confidence in addition to inaccuracy in ensemble predictions; a lower CRPS is better. For each variable, it compares the ground truth target y with the Cumulative Distribution Function (CDF) F of the prediction via

$$\begin{aligned}\mathrm{CRPS}(F, y) &:= \int \left(F(x) - \mathbf{1}_{\{x \geq y\}}\right)^2 dx \\ &= \mathbb{E}[|X - y|] - \frac{1}{2}\mathbb{E}[|X - X'|],\end{aligned}$$

where $X, X' \sim F$ are iid samples from the distributional prediction. We use the non-parametric "fair estimate to the CRPS" [14],

$$\hat{\mathrm{CRPS}}(\mathbf{X}, y) := \frac{1}{n}\sum_{i=1}^{n}|X_i - y| - \frac{1}{2n(n-1)}\sum_{i=1}^{n}\sum_{j=1}^{n}|X_i - X_j|, \tag{6}$$

estimating F with the empirical CDF of $n = 32$ iid samples $X_i \sim F$. The first term in (6) is the MAE between the target and samples of the predictive distribution, while the second term is small for small predictive variances and vanishes completely for point estimates. Note that this definition extends to ensemble models, where we take the prediction of each ensemble member as a sample of an implicit predictive distribution.

4.2 Results

MAE and R^2 of the baseline models are presented in the main text (e.g., Table 2 and Figure 2 in the main text). Here, we show RMSE and CRPS in Table 2 and Figures 2, 4, and 5.

Variable	RMSE [W/m^2]					CRPS [W/m^2]				
	CNN	HSR	MLP	RPN	cVAE	CNN	HSR	MLP	RPN	cVAE
dT/dt	**4.369**	4.825	4.421	4.482	4.721	–	3.284	–	**2.580**	2.795
dq/dt	**7.284**	7.896	7.322	7.518	7.780	–	4.899	–	**4.022**	4.372
NETSW	36.91	37.77	**26.71**	33.60	38.36	–	0.055	–	**0.053**	0.057
FLWDS	10.86	8.220	**6.969**	7.914	8.530	–	0.018	–	**0.016**	0.018
PRECSC	6.001	6.095	**4.734**	5.511	6.182	–	0.011	–	**0.008**	0.009
PRECC	85.31	90.64	**72.88**	76.58	88.71	–	0.122	–	**0.085**	0.097
SOLS	22.92	23.61	**17.40**	20.61	23.27	–	0.031	–	**0.028**	0.033
SOLL	27.25	27.78	**21.95**	25.22	27.81	–	0.038	–	**0.035**	0.040
SOLSD	12.13	12.40	**9.420**	11.00	12.64	–	0.018	–	**0.015**	0.016
SOLLD	12.10	12.47	**10.12**	11.25	12.63	–	0.017	–	**0.015**	0.016

Table 2: Averaged RMSE and CRPS. Each metric is first calculated at each grid point and then averaged globally (i.e., horizontally averaged and, for dT /dt and dq/dt, vertically averaged). The units of non-energy flux variables are converted to a common energy unit, W/m^2, following Section 5.2. Best model performance for each variable is highlighted in bold.

We also present the spatial structure of the metrics. Figure 6 shows the latitude-height structure of R^2.

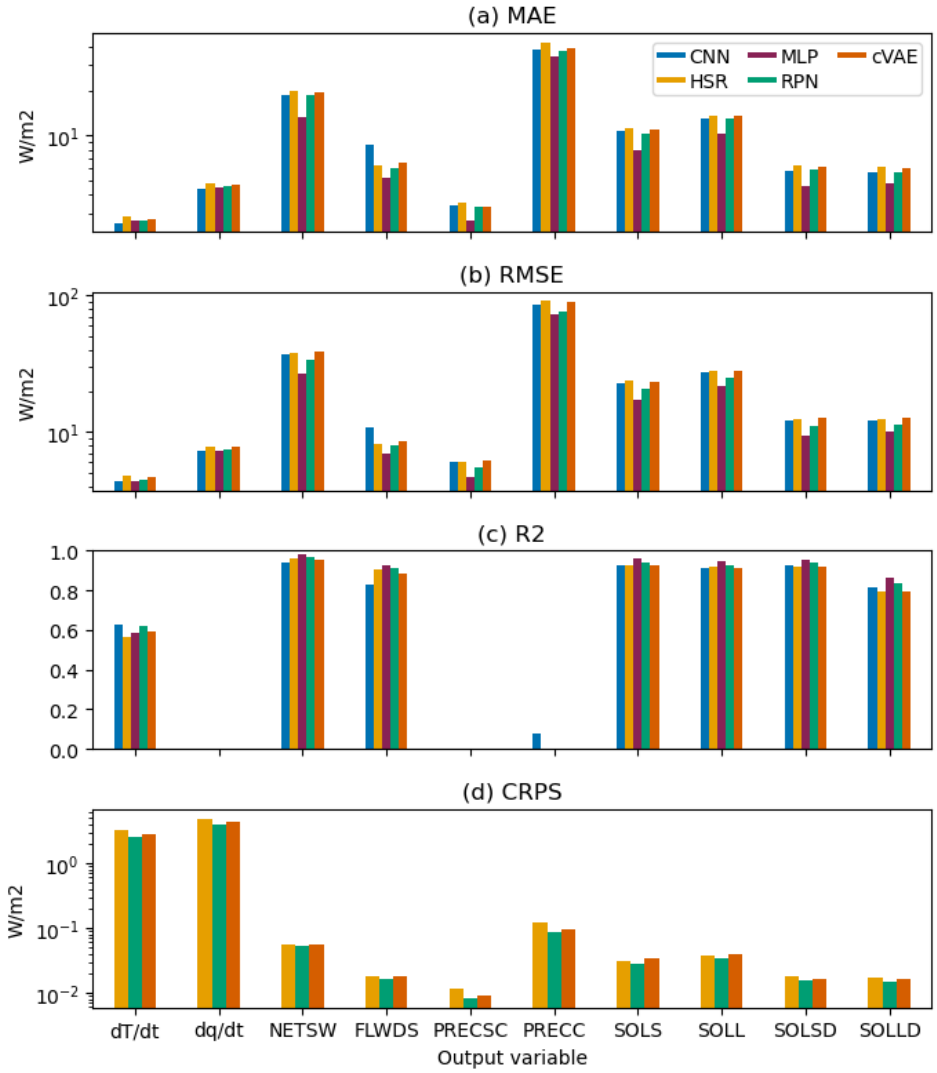

Figure 3: Averaged (a) MAE, (b) RMSE, (c) R^2, and (d) CRPS. Each metric is first calculated at each grid point and then averaged globally (i.e., horizontally averaged and, for dT/dt and dq/dt, vertically averaged). For MAE, RMSE, and CRPS, the units of non-energy flux variables are converted to a common energy unit, W/m^2, following Section 5.2. The negative values are not shown for R^2. dT/dt and dq/dt are the temperature and specific humidity tendencies. Refer to Table 1 in the main text for definitions of rest variables.

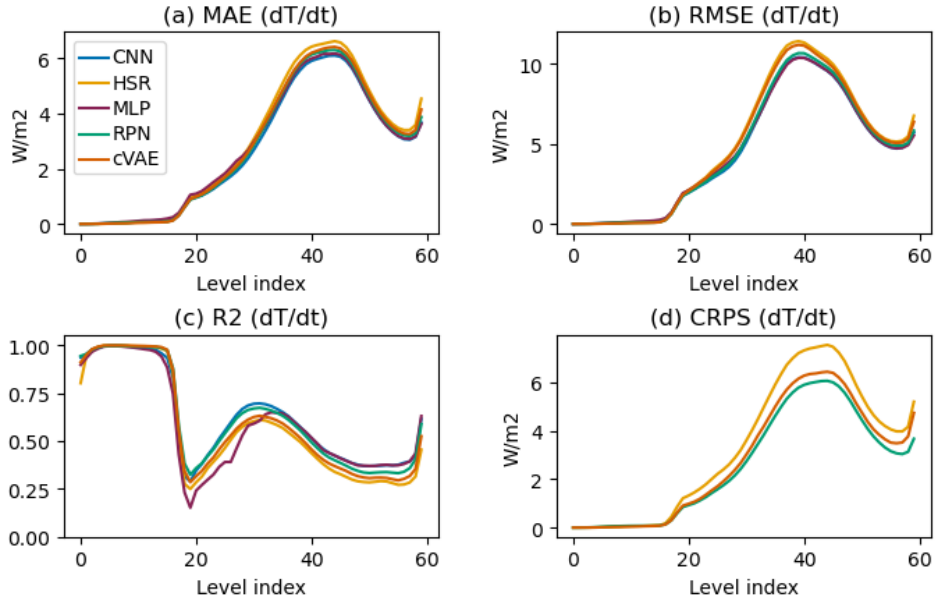

Figure 4: The vertical structures of horizontally-averaged (a) MAE, (b) RMSE, (c) R^2, and (d) CRPS of dT/dt (temperature tendency). For MAE, RMSE, and CRPS, the units of non-energy flux variables are converted to a common energy unit, W/m^2, following Section 5.2. The negative values are not shown for R^2.

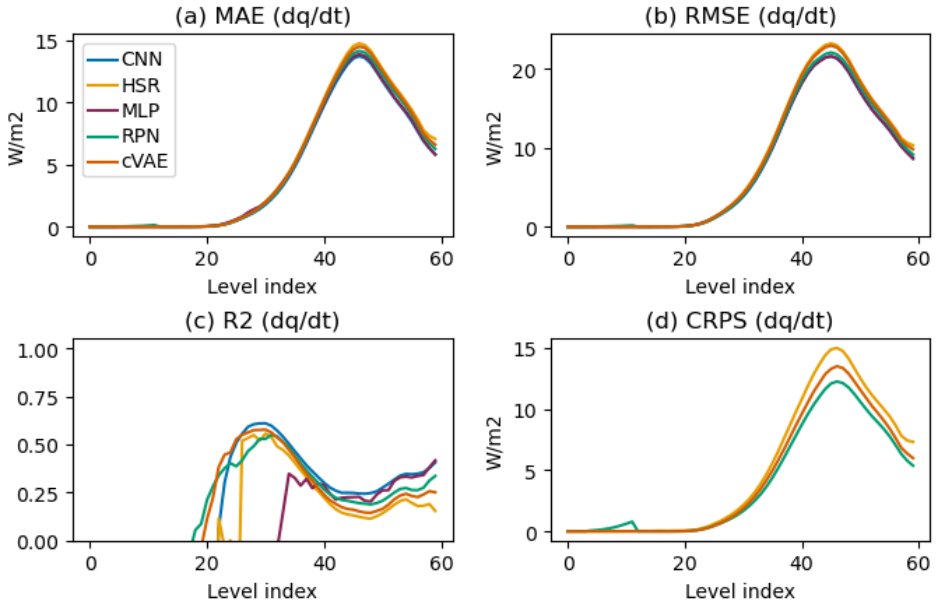

Figure 5: The vertical structures of horizontally-averaged (a) MAE, (b) RMSE, (c) R^2, and (d) CRPS of dq/dt (specific humidity tendency). For MAE, RMSE, and CRPS, the units of non-energy flux variables are converted to a common energy unit, W/m^2, following Section 5.2. The negative values are not shown for R^2.

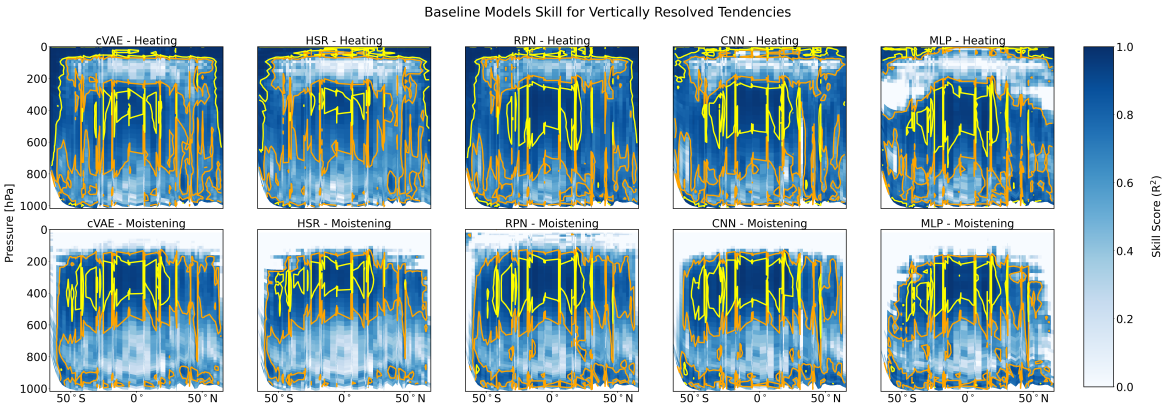

Figure 6: R^2 of daily-mean, zonal-mean (upper row) temperature tendency and (lower row) specific humidity tendency. Yellow contours surround regions of $> .9R^2$ while orange contours surround regions of $> .7R^2$. Negative values are not plotted (white). Sin(latitude) is used for x-axis to account for the curvature of Earth. The pressure levels on Y-axis are approximated values.

5 Guidance

5.1 Physical Constraints

Mass and energy conservation are important criteria for earth system modeling. If these terms are not conserved, errors in estimating sea level rise or temperature change over time may become as large as the signals we hope to measure. Enforcing conservation on emulated results helps constrain results to be physically plausible and reduce the potential for errors accumulating over long time scales.

In the atmospheric component of the E3SM climate model, the mass is composed of "dry air" (well-mixed gases such as molecular nitrogen and oxygen) and water vapor. During the physics parameterizations we seek to emulate, there is no lateral exchange of mass across columns of the host model and the model assumes that the total mass in each column and level remains unchanged. Thus, while surface pressure (`state_ps`) is part of the `state` structure we will be emulating, that surface pressure component must be held fixed. The water mass, however, is not held fixed, requiring fictitious sources and sinks of dry air, which are corrected later in the model outside of the part of the code we seek to emulate and will not be addressed within the emulator.

Changes in column water mass should balance the sources and sinks of water into and out of the column through surface fluxes. The surface source of water is an input to the emulator via the `cam_in` structure. The surface sink of water is generated by the model, and hence emulated in our case. The net surface water flux (source minus sink) should be equal to the tendency of water mass within the column (equation 7). The mass of water is held in five separate terms within the `state` structure: water vapor (q_v), cloud liquid condensate (q_l), cloud ice (q_i), rain (q_r), and snow (q_s). These terms are held as ratios of their mass to the sum of dry air plus water vapor (referred to as specific humidity). The "δ" refers to the difference (after minus before computation) in each quantity owing to the CRM physics. The layer mass (sum of dry air plus water vapor) of level k is equal to the pressure thickness of that layer (Δp_i; the difference between top and bottom interface pressure for level i) divided by the gravitational acceleration (g assumed constant). The time step length is δt. In addition to conserving water mass, we require each individual water constituent to remain greater than or equal to zero in every layer within the column. In equation 7, E is the surface source of water (evapotranspiration) and P is the surface sink of water (precipitation).

$$\sum_i \left(\delta q_v + \delta q_l + \delta q_i + \delta q_r + \delta q_s\right) \frac{\Delta p_i}{g \delta t} = E - P \tag{7}$$

For the portion of the code that we seek to emulate, the water source E is not applied such that the only surface flux to account for when constraining water conservation is the precipitation flux (P, `cam_out_PRECC`). Unfortunately, we only have the input and output `state` variables for water vapor (`state_q0001`), cloud liquid (`state_q0002`), and cloud ice (`state_q0003`). The additional storage terms related to precipitating water that have not exited the column over the course of a model time step are unavailable in the current output. Therefore, we can not exactly enforce water conservation at present. Estimates show relative errors of a couple percent resulting from the lack of these precipitation mixing ratios. We can still require that the relative error be small. To accomplish this, we compare the "expected" total water, based on the combination of the input and surface fluxes, to the predicted total water. In the equations below, superscript o denotes output and superscript i denotes input.

$$\text{Total Water (actual)} = \sum_i \left(\delta q_v^o + \delta q_l^o + \delta q_i^o\right) \frac{\Delta p_i}{g} \tag{8}$$

$$\text{Total Water (expected)} = \sum_i \left(\delta q_v^i + \delta q_l^i + \delta q_i^i\right) \frac{\Delta p_i}{g} - P \delta t \tag{9}$$

$$\text{Relative Error} = \frac{\text{Total Water (expected)} - \text{Total Water (actual)}}{\text{Total Water (actual)}} \tag{10}$$

$$\tag{11}$$

We can require the model to keep the relative error small (e.g., below 5%). Anything further is beyond the limit of the current data.

Like mass conservation, energy conservation can generally be enforced by requiring that the total change within the column is exactly balanced by the fluxes into and out of that column. Because the emulator will not predict upwelling radiative fluxes at the model top (a sink term for energy), we will not have the boundary conditions necessary to constrain column energy tendencies. We can still require certain criteria be met for physical consistency. First, the downwelling surface shortwave radiative flux cannot exceed the downwelling shortwave flux at the model top (prescribed input `pbuf_SOLIN`). Likewise, the net surface shortwave flux should also be bounded between zero (100% reflection) and the surface downwelling shortwave flux (100% absorption). Additionally, the downwelling longwave flux should not exceed the blackbody radiative flux from the warmest temperature in the column.

5.2 Unit Conversion and Weighting for Interpretable Evaluation

To facilitate the objective evaluation of the model's prediction, we provide a weight tensor of shape $(d_o, N_{\boldsymbol{x}})$ to convert raw outputs to area-weighted outputs with consistent energy flux units $\left[\mathrm{W/m^2}\right]$. We provide details below.

To ensure that our evaluation takes the Earth's spherical geometry into account, we design an area weighting factor a that depends on the horizontal position $\boldsymbol{x}$:

$$a\left(\boldsymbol{x}\right) = \mathcal{A}_{\mathrm{col}}\left(\boldsymbol{x}\right) / \left\langle \mathcal{A}_{\mathrm{col}} \right\rangle_{\boldsymbol{x}}$$

Where $\mathcal{A}_{\mathrm{col}}$ is the area of an atmospheric column and $\left\langle \mathcal{A}_{\mathrm{col}} \right\rangle_{\boldsymbol{x}}$ the horizontal average of all atmospheric columns' areas. This formula gives more weight to outputs if their grid cell has a larger horizontal

area. To ensure that our evaluation is physically-consistent, we convert all predicted variables to energy flux units $\left[\left[\mathrm{W/m^2}\right]\right.$ (power per unit area). This has to be done for each variable separately.

- For heating tendencies $\dot{T}\,[\mathrm{K/s}]$, which depend on the horizontal position $\boldsymbol{x}$ and vertical level lev, this can be done using the specific heat capacity constant at constant pressure $c_p\,[\mathrm{J/\,(K \times kg)}]$:

$$\dot{T}\left[\mathrm{W/m^2}\right] = \frac{c_p}{g} \times a\left(\boldsymbol{x}\right) \times \Delta p_i\left(\mathrm{lev}\right) \times \dot{T}\left[\mathrm{K/s}\right]$$

  Where $\Delta p_i\,[\mathrm{Pa}]$ is the layer's pressure thickness calculated as the difference between the pressure at the layer's top and bottom interfaces.

- For water concentration tendencies $\dot{q}\left[\mathrm{s^{-1}}\right]$, which also depend on the horizontal position and the vertical level, this can be done using the latent heat of vaporization of water vapor at constant pressure $L_v\,[\mathrm{J/kg}]$:

$$\dot{q}\left[\mathrm{W/m^2}\right] = \frac{L_v}{g} \times a\left(\boldsymbol{x}\right) \times \Delta p_i\left(\mathrm{lev}\right) \times \dot{q}\left[\mathrm{s^{-1}}\right]$$

  Note that there is some level of arbitrariness as the exact latent heat depends on which water phase is assumed to calculate the energy transfer; here, we chose to weigh all phases using L_v to give them comparable weights in the evaluation metrics.

- For momentum tendencies $\dot{u}\left[\mathrm{m/s^2}\right]$, which also depend on the horizontal position and the vertical level, we can use a characteristic wind magnitude $|\boldsymbol{U}|\,[\mathrm{m/s}]$ to convert these tendencies into turbulent kinetic energy fluxes, in units $\mathrm{W/m^2}$, that can be compared to $\dot{T}\left[\mathrm{W/m^2}\right]$ and $\dot{q}\left[\mathrm{W/m^2}\right]$:

$$\dot{u}\left[\mathrm{W/m^2}\right] = \frac{|\boldsymbol{U}|}{g} \times a\left(\boldsymbol{x}\right) \times \Delta p_i\left(\mathrm{lev}\right) \times \dot{u}\left[\mathrm{m/s^2}\right]$$

  Note that there is some level of arbitrariness in the choice of $|\boldsymbol{U}|\,[\mathrm{m/s}]$, which could e.g., be chosen so that the variances of $\dot{u}\left[\mathrm{W/m^2}\right]$ and $\dot{T}\left[\mathrm{W/m^2}\right]$ are comparable.

- Precipitation rate variables $P\,[\mathrm{m/s}]$ can also be converted to energy fluxes using the latent heat of vaporization of water vapor L_v and the density of liquid water $\rho_w\,\left[\mathrm{kg/m^3}\right]$ (or the density of snow/ice for solid precipitation), though they do not require vertical integration:

$$P\left[\mathrm{W/m^2}\right] = L_v \times \rho_w \times a\left(\boldsymbol{x}\right) \times P\left[\mathrm{m/s}\right]$$

- Finally, surface energy fluxes $\mathcal{F}\left[\mathrm{W/m^2}\right]$ are simply multiplied by $a\left(\boldsymbol{x}\right)$ to account for area-weighting.

Note that while these choices ensure unit consistency, which facilitates the physical interpretation of our evaluation metrics, we recommend tailoring the exact choice of physical constants to the application of interest.

5.3 Additional Guidance

Stochasticity and memory: The results of the embedded convection calculations regulating d_o come from a chaotic dynamical system and thus could be worthy of architectures and metrics beyond the deterministic baselines in this paper. These solutions are likewise sensitive to sub-grid initial state variables from an interior nested spatial dimension that have not been included in our data.

Temporal locality: Incorporating the previous time steps' target or feature in the input vector inflation could be beneficial as it captures some information about this convective memory and utilizes temporal autocorrelations present in atmospheric data.

Causal pruning: A systematic and quantitative pruning of the input vector based on objectively assessed causal relationships to subsets of the target vector has been proposed as an attractive preprocessing strategy, as it helps remove spurious correlations due to confounding variables and optimize the ML algorithm [15].

Normalization: Normalization that goes beyond removing vertical structure could be strategic, such as removing the geographic mean (e.g., latitudinal, land/sea structure) or composite seasonal variances (e.g., local smoothed annual cycle) present in the data. For variables exhibiting exponential variation and approaching zero at the highest level (e.g., metrics of moisture), log-normalization might be beneficial.

6 Other Related Work

Several benchmark datasets have been developed to facilitate AI tasks in weather and climate. ClimateNet [16] and Extremeweather [17] were both designed for AI-based feature detection of extreme weather events in forecasts of Earth's future climate made using conventional climate models. WeatherBench [18] provides data specifically designed for data-driven weather forecasting, focusing on periods ranging from 3 to 5 days into the future. PDEBench [19] provides data from numerical simulations of several partial differential equations (PDEs) for benchmarking AI PDE emulators. ClimateBench [20] was designed for emulators that produce annual mean global predictions of temperature and precipitation given greenhouse gas concentrations and emissions. ClimART [21] is designed for the development of radiative energy transfer parameterization emulators for use in weather and climate modeling. These benchmark datasets play a vital role in advancing AI and ML research within the weather and climate domains.

ClimSim, a dataset for parameterization emulators trained on high-resolution data from small-scale embedded models, is unique compared to other benchmark datasets designed for emulators in climate simulation (ClimateBench, ClimART, and PDEBench). While PDEBench provides data for developing AI emulators of the same PDEs commonly used in climate simulation, ClimSim is uniquely tailored to address the challenging task of replacing a sophisticated parameterization for the combined effects of clouds, rain, radiation, and storms. Specifically, models trained using ClimSim will learn to emulate the nonlinear effect of clouds, rain, and storms resolved on the 1-km (20 s) space (time) scale, which is a collection of hundreds of equations rather than one, to represent their upscale impacts on the 100-km (30 min) scale. Hybrid simulation is also the goal of ClimART, which is designed specifically for the narrower and lest computationally costly task of radiative energy transfer parameterization, rather than cloud and rain emulators. ClimateBench, on the other hand, is not an attempt at hybrid simulation, but rather for "whole-model" emulators that reproduce the annual mean global predictions of climate that a conventional climate model would simulate given unseen greenhouse gas concentrations and emissions. This does not attempt to sidestep Moore's Law or admit previously unattainable resolution, i.e., any error or bias related to the parameterizations used to create the training data are part of what is learned by the emulator.

In contrast, the goal of ClimSim is to develop an emulator for the *explicitly resolved* effect of clouds and storms on climate, so that, down the road, the emulator can be used to replace parameterizations in a climate model, enabling more realistic climate simulation without the typical computational overhead. ClimSim builds off work by a few climate scientists who have been exploring since 2017 to apply ML for hybrid multi-scale climate modeling. [22] first demonstrated that using simple

ML models, and a simple atmosphere test-bed, certain atmospheric patterns of convective heating and moistening could be effectively predicted, particularly in the tropics and mid-latitude storm tracks. However, when these models were integrated into broader climate simulations, except for lucky fits that demonstrated the exciting potential for success [23], issues related to stability arose, a common problem when constructing hybrid climate models. Various methods were tried to improve the stability, such as coupling multiple models together and searching for better model architectures [24, 25]. These efforts led to improved error rates in the predictions. More recently, researchers have expanded this work into real-world settings, using more advanced ML architectures [26–28]. [29] even managed to create a deep-learning model that showed hybrid stability over a decade under real-world conditions. While this hybrid model had a few biases, it was successful in capturing some aspects of climate variability. Additionally, work has been done to compress input data to avoid causal confounders while maintaining accuracy [15], use latent representations that account for stochasticity [30] and enforce physical constraints within these models [31], all of which could potentially improve their reliability.

7 Datasheet

7.1 Motivation

1. **For what purpose was the dataset created?** *Our benchmark dataset was created to serve as a foundation for developing robust frameworks that emulate parameterizations for cloud and extreme rainfall physics, and their interaction with other sub-resolution processes.*
2. **Who created the dataset and on behalf of which entity?** *The dataset was developed by a consortium of climate scientists and ML researchers listed in the author list.*
3. **Who funded the creation of the dataset?** *The main funding body is the National Science Foundation (NSF) Science and Technology Center (STC) Learning the Earth with Artificial Intelligence and Physics (LEAP). Other funding sources of individual authors are listed in the acknowledgement section of the main text.*

7.2 Distribution

1. **Will the dataset be distributed to third parties outside of the entity (e.g., company, institution, organization) on behalf of which the dataset was created?** *Yes, the dataset is open to the public.*
2. **How will the dataset will be distributed (e.g., tarball on website, API, GitHub)?** *The dataset will be distributed through Hugging Face and the code used for developing baseline models through GitHub.*
3. **Have any third parties imposed IP-based or other restrictions on the data associated with the instances?** *No.*
4. **Do any export controls or other regulatory restrictions apply to the dataset or to individual instances?** *No.*

7.3 Maintenance

1. **Who will be supporting/hosting/maintaining the dataset?** *NSF-STC LEAP will support, host, and maintain the dataset.*
2. **How can the owner/curator/manager of the dataset be contacted (e.g., email address)?** *The owner/curator/manager of the dataset can be contacted through following emails: Sungduk Yu (sungduk@uci.edu), Michael S. Pritchard (mspritch@uci.edu) and LEAP (leap@columbia.edu).*

3. **Is there an erratum?** *No. If errors are found in the future, we will release errata on https://leap-stc.github.io/ClimSim*
4. **Will the dataset be updated (e.g., to correct labeling errors, add new instances, delete instances)?** *Yes, the datasets will be updated, whenever necessary, to ensure accuracy, and announcements will be made accordingly. These updates will be posted on the main web page for the dataset (https://leap-stc.github.io/ClimSim).*
5. **If the dataset relates to people, are there applicable limits on the retention of the data associated with the instances (e.g., were the individuals in question told that their data would be retainded for a fixed period of time and then deleted?)** *N/A*
6. **Will older version of the dataset continue to be supported/hosted/maintained?** *Yes, older versions of the dataset will continue to be maintained and hosted.*
7. **If others want to extend/augment/build on/contribute to the dataset, is there a mechanisms for them to do so?** *No.*

7.4 Composition

1. **What do the instance that comprise the dataset represent (e.g., documents, photos, people, countries?)** *Each instance includes both input and output vector pairs. These input and output are instantaneous snapshots of atmospheric states surrounding detailed numerical calculations to be emulated.*
2. **How many instances are there in total (of each type, if appropriate)?** *The high-resolution dataset (ClimSim_high-res) includes 5,676,480,000 instances, and each low-resolution dataset (ClimSim_low-res and ClimSim_low-res_aqua-planet) includes 100,915,200 instances.*
3. **Does the dataset contain all possible instances or is it a sample of instances from a larger set?** *The datasets contain 80% of all possible instances. The rest 20% are reserved as the holdout test set, which will be released once enough models using ClimSim are developed by independent groups.*
4. **Is there a label or target associated with each instance?** *Yes, each instance includes both input and target (prediction) variables.*
5. **Is any information missing from individual instances?** *No.*
6. **Are there recommended data splits (e.g., training, development/validation, testing)?** *We have a hard split between the train/validation set and the test set. The first 8-year worth dataset is reserved for the train/validation set, and the last 2-year worth dataset for the test set. However, we do not have specific recommendations on the split within train/validation set.*
7. **Are there any errors, sources of noise, or redundancies in the dataset?** *There is one redundancy. Input variable "state_pmid" is redundant since it is a linear function of "state_ps".*
8. **Is the dataset self-contained, or does it link to or otherwise rely on external resources (e.g., websites, tweets, other datasets)?** *The dataset is self-contained.*
9. **Does the dataset contain data that might be considered confidential?** *No.*
10. **Does the dataset contain data that, if viewed directly, might be offensive, insulting, threating, or might otherwise cause anxiety?** *No.*

7.5 Collection Process

1. **How was the data associated with each instance acquired?** *The data associated with each instance is acquired from a series of simulations of a global climate model called E3SM-MMF. References for E3SM-MMF are provided in Section 3 of the main text.*
2. **What mechanisms or procedures were used to collect the data (e.g., hardware apparatus or sensor, manual human curation, software program, software API)?** *We used many NVIDI A100 GPU nodes in a high-performance computing cluster called Perlmutter (operated by the US Department of Energy) to run E3SM-MMF simulations.*
3. **Who was involved in the data collection process (e.g., students, crowdworkers, contractors) and how were they compensated (e.g., how much were crowdworkers paid)?** *Regular employees (e.g., scientists and postdocs) at UC Irvine, LLNL, and SNL were involved in the data collection process. No crowdworkers were involved during the data collection process.*
4. **Does the dataset relate to people?** *No.*
5. **Did you collect the data from the individuals in questions directly, or obtain it via third parties or other sources (e.g., websites)?** *We obtained the dataset from computer simulations of Earth's climate.*

7.6 Uses

1. **Has the dataset been used for any tasks already?** *No this dataset has not been used for any tasks yet.*
2. **What (other) tasks could be the dataset be used for?** *Please refer to Section 5 in the main manuscript for other applications.*
3. **Is there anything about the composition of the dataset or the way it was collected and preprocessed/cleaned/labeled that might impact future uses?** *The current composition of the dataset are self-sufficient to build a climate emulator. However, it misses some extra variables, which are not essential for such climate emulators but necessary to strictly enforce physical constraints (see Section 4.4 of the main text). We plan to include these extra variables in the next release. Any changes in the next release and update use guideline will be documented and shared through the dataset webpage (https://leap-stc.github.io/ClimSim).*
4. **Are there tasks for which the dataset should not be used?** *Not to our best knowledge.*

References

References

[1] D. E3SM Project, "Energy exascale earth system model v2.1.0." [Computer Software] https://doi.org/10.11578/E3SM/dc.20230110.5, 2023.

[2] G. J. Kooperman, M. S. Pritchard, M. A. Burt, M. D. Branson, and D. A. Randall, "Robust effects of cloud superparameterization on simulated daily rainfall intensity statistics across multiple versions of the community earth system model," *J. Adv. Model. Earth Syst.*, vol. 8, no. 1, pp. 140–165, 2016.

[3] W. M. Hannah, A. M. Bradley, O. Guba, Q. Tang, J.-C. Golaz, and J. Wolfe, "Separating physics and dynamics grids for improved computational efficiency in spectral element earth system models," *J. Adv. Model. Earth Syst.*, vol. 13, no. 7, p. e2020MS002419, 2021.

[4] M. Khairoutdinov and D. Randall, "Cloud resolving modeling of the arm summer 1997 iop: Model formulation, results, uncertainties, and sensitivities," *J. Atmos. Sci.*, vol. 60, no. 4, pp. 607–625, 2003.

[5] S. N. Tulich, "A strategy for representing the effects of convective momentum transport in multiscale models: Evaluation using a new superparameterized version of the weather research and forecast model (sp-wrf)," *J. Adv. Model. Earth Syst.*, vol. 7, no. 2, pp. 938–962, 2015.

[6] M. R. Norman, D. C. Bader, C. Eldred, W. M. Hannah, B. R. Hillman, C. R. Jones, J. M. Lee, L. Leung, I. Lyngaas, K. G. Pressel, *et al.*, "Unprecedented cloud resolution in a gpu-enabled full-physics atmospheric climate simulation on olcf's summit supercomputer," *Int. J. High Perform. Compu. Appl.*, vol. 36, no. 1, pp. 93–105, 2022.

[7] T. O'Malley, E. Bursztein, J. Long, F. Chollet, H. Jin, L. Invernizzi, *et al.*, "Kerastuner." https://github.com/keras-team/keras-tuner, 2019.

[8] S. Yu, M. Pritchard, P.-L. Ma, B. Singh, and S. Silva, "Two-step hyperparameter optimization method: Acceleratinghyperparameter search by using a fraction of a training dataset," *Artificial Intelligence for the Earth Systems*, 2023, *under review*.

[9] I. Osband, J. Aslanides, and A. Cassirer, "Randomized prior functions for deep reinforcement learning," 2018.

[10] Y. Yang, G. Kissas, and P. Perdikaris, "Scalable uncertainty quantification for deep operator networks using randomized priors," *Comput. Methods Appl. Mech. Eng.*, vol. 399, p. 115399, 2022.

[11] M. A. Bhouri, M. Joly, R. Yu, S. Sarkar, and P. Perdikaris, "Scalable bayesian optimization with high-dimensional outputs using randomized prior networks," 2023.

[12] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," 2018.

[13] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *Proc. ICLR*, 2014.

[14] C. A. T. Ferro, "Fair scores for ensemble forecasts," *Quarterly Journal of the Royal Meteorological Society*, vol. 140, no. 683, pp. 1917–1923, 2014.

[15] F. Iglesias-Suarez, P. Gentine, B. Solino-Fernandez, T. Beucler, M. Pritchard, J. Runge, and V. Eyring, "Causally-informed deep learning to improve climate models and projections," 2023.

[16] D. Prabhat, K. Kashinath, M. Mudigonda, S. Kim, L. Kapp-Schwoerer, A. Graubner, E. Karaismailoglu, L. von Kleist, T. Kurth, A. Greiner, A. Mahesh, K. Yang, C. Lewis, J. Chen, A. Lou, S. Chandran, B. Toms, W. Chapman, K. Dagon, C. A. Shields, T. O'Brien, M. Wehner, and W. Collins, "Climatenet: an expert-labeled open dataset and deep learning architecture for enabling high-precision analyses of extreme weather," *Geosci. Model Dev.*, vol. 14, no. 1, pp. 107–124, 2021.

[17] E. Racah, C. Beckham, T. Maharaj, S. E. Kahou, Prabhat, and C. Pal, "Extremeweather: A large-scale climate dataset for semi-supervised detection, localization, and understanding of extreme weather events," 2017.

[18] S. Rasp, P. D. Dueben, S. Scher, J. A. Weyn, S. Mouatadid, and N. Thuerey, "Weatherbench: A benchmark data set for data-driven weather forecasting," *J. Adv. Model. Earth Syst.*, vol. 12, no. 11, p. e2020MS002203, 2020.

[19] M. Takamoto, T. Praditia, R. Leiteritz, D. MacKinlay, F. Alesiani, D. Pflüger, and M. Niepert, "Pdebench: An extensive benchmark for scientific machine learning," 2023.

[20] D. Watson-Parris, Y. Rao, D. Olivié, Ø. Seland, P. Nowack, G. Camps-Valls, P. Stier, S. Bouabid, M. Dewey, E. Fons, J. Gonzalez, P. Harder, K. Jeggle, J. Lenhardt, P. Manshausen, M. Novitasari, L. Ricard, and C. Roesch, "Climatebench v1.0: A benchmark for data-driven climate projections," *J. Adv. Model. Earth Syst.*, vol. 14, no. 10, p. e2021MS002954, 2022.

[21] S. R. Cachay, V. Ramesh, J. N. S. Cole, H. Barker, and D. Rolnick, "Climart: A benchmark dataset for emulating atmospheric radiative transfer in weather and climate models," 2021.

[22] P. Gentine, M. Pritchard, S. Rasp, G. Reinaudi, and G. Yacalis, "Could machine learning break the convection parameterization deadlock?," *Geophys. Res. Lett.*, vol. 45, no. 11, pp. 5742–5751, 2018.

[23] S. Rasp, M. S. Pritchard, and P. Gentine, "Deep learning to represent subgrid processes in climate models," *Proc. Natl. Acad. Sci. USA*, vol. 115, no. 39, pp. 9684–9689, 2018.

[24] N. D. Brenowitz, T. Beucler, M. Pritchard, and C. S. Bretherton, "Interpreting and stabilizing machine-learning parameterizations of convection," *J. Atmos. Sci.*, vol. 77, no. 12, pp. 4357–4375, 2020.

[25] J. Ott, M. Pritchard, N. Best, E. Linstead, M. Curcic, and P. Baldi, "A fortran-keras deep learning bridge for scientific computing," 2020.

[26] Y. Han, G. J. Zhang, X. Huang, and Y. Wang, "A moist physics parameterization based on deep learning," *J. Adv. Model. Earth Syst.*, vol. 12, no. 9, p. e2020MS002076, 2020.

[27] G. Mooers, M. Pritchard, T. Beucler, J. Ott, G. Yacalis, P. Baldi, and P. Gentine, "Assessing the potential of deep learning for emulating cloud superparameterization in climate models with real-geography boundary conditions," *J. Adv. Model. Earth Syst.*, vol. 13, no. 5, p. e2020MS002385, 2021.

[28] X. Wang, Y. Han, W. Xue, G. Yang, and G. J. Zhang, "Stable climate simulations using a realistic general circulation model with neural network parameterizations for atmospheric moist physics and radiation processes," *Geosci. Model Dev.*, vol. 15, no. 9, pp. 3923–3940, 2022.

[29] P. Wang, J. Yuval, and P. A. O'Gorman, "Non-local parameterization of atmospheric subgrid processes with neural networks," *J. Adv. Model. Earth Syst.*, vol. 14, no. 10, p. e2022MS002984, 2022.

[30] G. Behrens, T. Beucler, P. Gentine, F. Iglesias-Suarez, M. Pritchard, and V. Eyring, "Non-linear dimensionality reduction with a variational encoder decoder to understand convective processes in climate models," *J. Adv. Model. Earth Syst.*, vol. 14, no. 8, p. e2022MS003130, 2022.

[31] T. Beucler, M. Pritchard, S. Rasp, J. Ott, P. Baldi, and P. Gentine, "Enforcing analytic constraints in neural networks emulating physical systems," *Phys. Rev. Lett.*, vol. 126, no. 9, p. 098302, 2021.