Deep learned process parameterizations provide better representations of turbulent heat fluxes in hydrologic models

Enter authors here: Andrew Bennett[1], Bart Nijssen[1]

[1]Department of Civil and Environmental Engineering, University of Washington, Seattle, WA, USA

Corresponding author: Andrew Bennett (andrbenn@uw.edu)

Key Points:

- Deep learned process parameterizations of turbulent heat fluxes outperform physically-based parameterizations.
- Deep learned process parameterizations can be dynamically coupled into process-based hydrologic models.
- Incorporation of process-based model derived states into deep learning introduces feedbacks that improve long-term simulations.

Abstract

Deep learning (DL) methods have shown great promise for accurately predicting hydrologic processes but have not yet reached the complexity of traditional process-based hydrologic models (PBHM) in terms of representing the entire hydrologic cycle. The ability of PBHMs to simulate the hydrologic cycle makes them useful for a wide range of modeling and simulation tasks, for which DL methods have not yet been adapted. We argue that we can take advantage of each of these approaches by embedding DL methods into PBHMs to represent individual processes. We demonstrate that this is viable by developing DL-based representations of turbulent heat fluxes and coupling them into the Structure for Unifying Multiple Modeling Alternatives (SUMMA), a modular PBHM modeling framework. We developed two DL parameterizations and integrated them into SUMMA, resulting in a one-way coupled implementation (NN1W) which relies only on model inputs and a two-way coupled implementation (NN2W), which also incorporates SUMMA-derived model states. Our results demonstrate that the DL parameterizations are able to outperform calibrated standalone SUMMA benchmark simulations. Further we demonstrate that the two-way coupling can simulate the long-term latent heat flux better than the standalone benchmark and one-way coupled configuration. This shows that DL methods can benefit from PBHM information, and the synergy between these modeling approaches is superior to either approach individually.

Plain Language Summary

Machine learning (ML) and process-based methods are two approaches to hydrologic modeling. Process-based hydrologic models (PBHMs) represent the hydrologic cycle by solving equations which have been developed from physical theory or experimentation, while ML models make predictions based on patterns learned from large amounts of data. A particular sub-field of machine learning called deep learning (DL) has been shown to often outperform process-based models. However, current DL models do not represent all aspects of the hydrologic cycle (such as streamflow, evaporation, groundwater storage, and snowpack) at once, as is often done in PBHMs. As a result, DL models in hydrology are often single purpose, while PBHMs can be used for many different scientific and/or engineering purposes.

We show how individual DL models that simulate evaporation and convective heat transport at the land surface can be incorporated into a PBHM. We show that deep learning simulated evaporation and convective heat transport better than the PBHM. We also show how the incorporation of deep learning into process-based models can further improve the DL model itself. We conclude that taking advantage of both modeling perspectives is better than either on its own.

1 Introduction

The debates amongst the hydrologic modeling community about the use and utility of machine learning (ML) to simulate hydrologic processes indicate that much work remains to be done to understand the role and potential of machine learning in hydrologic modeling (Nearing et al., 2020; Shen, 2018). While it is true that deep learning (DL) models have shown great promise and superior performance in many cases it is yet unclear how to make models that are both composable (that is, easy to combine with other models) and transferable for scientific studies (that is, the same model configuration can be used to explore disparate scientific questions). In this paper we outline an approach for coupling DL models of individual processes into existing hydrologic modeling frameworks. This coupling approach allows us to represent individual physical processes within a larger model using ML methods and to introduce feedbacks between model components. The ability to couple model components will address these composability and transferability questions, as well as allow use of these types of machine-learned models in areas which do not have readily available training data.

There are several reasons for the rapid advancement of ML-based approaches in hydrology (and other fields), including a greater abundance of publicly available data, increased computational resources, and better frameworks for selecting, fitting, and applying models. Along with this increase in interest, the community has also begun to think about how to incorporate aspects of physical theory into these data driven models. This desire for physics-based machine learning is enticing for a number of reasons. As scientists we hope that the use of models which are based in, or constrained by, physical properties will allow us to learn about the underlying processes of the systems we are modeling. Not only that, we hope that such approaches will be able to efficiently extract information from a variety of datasets, from in situ observations to satellite remote sensing data, or be able to represent complex phenomena in a more efficient way.

While inclusion of empirical or statistical relationships of individual processes in hydrologic models is common, this is not yet the case for ML methods. One reason for this is that it is not clear how to combine ML models in the same way that we have been able to include processes for which we have parsimonious descriptions. Additionally, methodologies for representing physical relationships between ML-based process representations have not been developed in the hydrology community. In part, this is not surprising since machine learning is good at resolving relationships that we have not been able to decompose into easily describable parts. This “whole-system” or “black box” approach is conceptually appealing due to its simplicity, and is exemplified by rainfall-runoff modeling, which deep learning has proven to be very good at (Hu et al., 2018; Kratzert et al., 2018; Moshe et al., 2020). However, by taking a more granular approach, we will show that DL models can be successfully incorporated as process modules into existing models. Doing so allows us to see how changes in a single component affect the entire system.

In this paper, we look at turbulent heat fluxes, for which high-quality, long-term, local observations from eddy covariance towers (here, from FluxNet; Pastorello et al., 2020) are available across a range of hydroclimates. While machine learning has been used for modeling of turbulent heat fluxes and evaporation (Jung et al., 2009; Tramontana et al., 2016; Zhao et al., 2019) there have not yet been model intercomparisons with land surface models, much less integrations into land surface models. However, Best et al. (2015) showed that even simple statistical models are often able to outperform state of the art land surface models in simulation of latent and sensible heat fluxes. Best et al. (2015) postulated that the statistical models were better able to use the information in the meteorological forcing data than the physics-based approaches. This indicates there is strong motivation for incorporating data-driven techniques into complex land surface and hydrologic models. We believe that if these types of approaches are able to provide better performance than the physically motivated relationships we should work to understand how and why this performance is better and use them where appropriate and applicable.

Despite the statistical benchmarks’ superior ability for predicting turbulent heat fluxes in Best et al. (2015), land surface models remain more suitable for a wide range of applications, because they represent a wider range of hydrologic processes and may be better suited for studies of environmental change. Such studies include drought prediction (Li et al., 2012), snow melt predictions under climate change (Musselman et al., 2017), and predicting volatile organic compound emissions (Lathière et al., 2006). That is not to say that ML models cannot be used in this way or incorporated into larger frameworks. Both Kratzert et al. (2018) and Jiang et al. (2020) make qualitative comparisons of internal ML model states to snowpack, but do not later use the models for prediction of snowpack. We believe that it is likely that ML models will be used for such purposes in the near future, but the question remains open how to extract process information from statistical models.

Because the hydrology community is still learning the best ways to build and use ML models, there remains considerable room for incorporation of machine learning into more conventional process-based hydrologic models (PBHMs), which have the flexibility needed for general purpose modeling. This approach has been adopted recently by Brenowitz & Bretherton (2018) as well as Rasp et al. (2018) for parameterizing sub-gridcell scale processes, such as cloud convection, in atmospheric circulation models. Similarly, in oceanography, neural networks have been used to parameterize the turbulent vertical mixing in the ocean surface (Ramadhan et al., 2020).

In this study, we demonstrate how coupling ML models into a hydrologic model can yield better performance at estimating turbulent heat fluxes without sacrificing mass and energy balance closure or the ability to represent other processes such as runoff or snowpack. We have developed two ML models to simulate latent and sensible heat fluxes. We embed these ML models as process parameterizations inside of a PBHM. These ML-based process parameterizations replace the turbulent heat flux equations of the original PBHM. Our first model was only allowed to learn from the same meteorological data that is used to force the hydrologic model, while our second ML model is additionally trained with the inclusion of states derived from the hydrologic model. We show that both ML models are able to outperform the routines for simulating turbulent heat fluxes at subdaily timescales. We also show that the configuration which was trained using model states is better able to reproduce the long-term water balance. Our results indicate that approaches to coupling machine learning with PBHMs offer a promising avenue, which has only begun to be explored.

2 Materials and Methods

2.1 Data and study sites

We used data from 60 FluxNet sites (Pastorello et al., 2020) to run our experiments. These sites cover a large variety of vegetation and climate classifications. Our site selection process considered several criteria. We first filtered the full FluxNet dataset to make sure we only included sites which had energy balance corrected measurements of both sensible and latent heat fluxes, which will be discussed later. We then made sure that these sites had the necessary variables to force our models, which include precipitation, air temperature, incoming shortwave radiation, incoming longwave radiation, specific humidity, air pressure, and wind speed. We then removed sites which had either fewer than three years of contiguous data or more than 20% missing observations during the longest continuous period with observations. For the remaining sites, we used gap-filled data provided as part of the FluxNet dataset. Gap-filling was based on ERA-Interim (ERAI) (Dee et al., 2011) and includes downscaling and postprocessing explicitly for the purpose of model forcing. Time steps flagged as gap-filled were excluded from our performance analysis to ensure that we did not simply measure the ability of our simulations to model ERAI data. However, the gap-filled data is included when analyzing the water balance.

We also limited our analysis to sites which had an observed ET/P ratio of less than 1.1, calculated using the mean FluxNet-reported values of ET and P over the simulation period. This was done to accommodate our model structure, which enforces mass and energy balances on a point (or lumped) scale. Larger observed ET/P ratios likely occur at sites which have strong spatial gradients and flow convergence, so that moisture available for ET is not just the result of local precipitation. Our filtering process resulted in 60 sites with 508 site-years of data. A breakdown of the site names, data periods, locations and site characteristics are given in Table 1. Figure 1 shows the locations and vegetation classes for these same sites.

Table 1. A listing of the sites, locations, IGBP vegetation types, and dates of simulation. Locations are given as (Latitude (°N), Longitude (°E)). Vegetation types are given by their IGBP codes. MF is mixed forest, ENF is evergreen needleleaf forest, CRL is croplands, GRL is grasslands, SVN is savannas, OSL is open shrublands, WLD is permanent wetlands, DBF is

deciduous broadleaf forest, and WS is woody savannas. Site names are taken from FluxNet, and consist of a two-letter country code followed by a three-letter site code.

Site name	Location	Veg Type	Start Time	End Time	Site name	Location	Veg Type	Start Time	End Time
AT-Neu	(47.1, 11.3)	GRL	1-2002	12-2012	**FI-Let**	(60.6, 24)	ENF	7-2009	12-2012
AU-ASM	(-22.3, 133.2)	ENF	1-2010	12-2014	**FI-Sod**	(67.4, 26.6)	ENF	4-2002	4-2005
AU-Cpr	(-34, 140.6)	SVN	1-2010	12-2014	**FR-LBr**	(44.7, -0.8)	ENF	1-1996	12-2008
AU-DaP	(-14.1, 131.3)	GRL	6-2007	12-2013	**FR-Pue**	(43.7, 3.6)	EBF	7-2004	3-2013
AU-How	(-12.5, 131.2)	WS	4-2009	12-2014	**IT-Cpz**	(41.7, 12.4)	EBF	4-2000	1-2009
AU-Stp	(-17.2, 133.4)	GRL	4-2008	12-2014	**IT-Lav**	(46, 11.3)	ENF	1-2003	12-2014
AU-Wac	(-37.4, 145.2)	EBF	5-2005	12-2008	**IT-MBo**	(46, 11)	GRL	1-2003	12-2013
AU-Wom	(-37.4, 144.1)	EBF	1-2010	12-2014	**IT-Noe**	(40.6, 8.2)	CSL	2-2004	12-2014
BE-Lon	(50.6, 4.7)	CRL	4-2004	10-2013	**IT-Ren**	(46.6, 11.4)	ENF	8-2003	12-2013
BE-Vie	(50.3, 6)	MF	1-1996	12-2014	**IT-Ro2**	(42.4, 11.9)	DBF	1-2002	2-2007
CA-Gro	(48.2, -82.2)	MF	1-2003	12-2014	**IT-SRo**	(43.7, 10.3)	ENF	6-2000	4-2009
CA-Qfo	(49.7, -74.3)	ENF	1-2003	12-2010	**IT-Tor**	(45.8, 7.6)	GRL	4-2008	12-2014
CA-TP1	(42.7, -80.6)	ENF	1-2002	12-2014	**NL-Hor**	(52.2, 5.1)	GRL	7-2004	4-2009
CA-TP3	(42.7, -80.3)	ENF	1-2002	12-2014	**RU-Fyo**	(56.5, 32.9)	ENF	1-1998	12-2014
CA-TPD	(42.6, -80.6)	DBF	1-2012	12-2014	**US-AR2**	(36.6, -99.6)	GRL	5-2009	12-2012
CH-Cha	(47.2, 8.4)	GRL	1-2006	3-2014	**US-ARM**	(36.6, -97.5)	CRL	1-2003	12-2012
CH-Fru	(47.1, 8.5)	GRL	1-2006	2-2014	**US-Blo**	(38.9, -120.6)	ENF	5-1998	12-2007
CN-HaM	(37.4, 101.2)	GRL	1-2002	12-2004	**US-CRT**	(41.6, -83.3)	CRL	1-2011	12-2013
CZ-wet	(49, 14.8)	WLD	3-2009	12-2014	**US-GLE**	(41.4, -106.2)	ENF	9-2004	12-2014
DE-Geb	(51.1, 10.9)	CRL	1-2001	12-2014	**US-Goo**	(34.3, -89.9)	GRL	5-2002	12-2006
DE-Gri	(51, 13.5)	GRL	1-2004	12-2014	**US-IB2**	(41.8, -88.2)	GRL	1-2004	12-2011
DE-Hai	(51.1, 10.5)	DBF	1-2000	8-2011	**US-KS2**	(28.6, -80.7)	CSL	5-2003	12-2006
DE-Kli	(50.9, 13.5)	CRL	5-2006	12-2014	**US-Los**	(46.1, -90)	WLD	9-2000	2-2009
DE-Obe	(50.8, 13.7)	ENF	1-2008	12-2014	**US-NR1**	(40, -105.5)	ENF	1-1998	12-2014
DE-Tha	(51, 13.6)	ENF	1-1996	12-2014	**US-Prr**	(65.1, -147.5)	ENF	11-2010	12-2014
DK-Eng	(55.7, 12.2)	GRL	6-2005	10-2008	**US-Syv**	(46.2, -89.3)	MF	9-2001	1-2008
ES-Amo	(36.8, -2.3)	OSL	6-2007	12-2012	**US-Ton**	(38.4, -121)	WS	1-2001	12-2014
ES-LJu	(36.9, -2.8)	OSL	1-2004	12-2013	**US-Var**	(38.4, -121)	GRL	11-2000	12-2011
FI-Hyy	(61.8, 24.3)	ENF	10-2004	8-2012	**US-WCr**	(45.8, -90.1)	DBF	8-2010	12-2014
FI-Jok	(60.9, 23.5)	CRL	2-2000	11-2003	**US-Whs**	(31.7, -110.1)	OSL	1-2007	12-2014

As noted, we chose to use the FluxNet-provided energy balance corrected turbulent heat fluxes.The energy balance gap in eddy-covariance measurements is an extensively studied topic (Foken, 2008; Kidston et al., 2010; Wilson et al., 2002), though no strong consensus has been reached on how to account for gaps in the observed energy balance (or even whether one should). However, because we will be using models and methods that enforce energy conservation, we chose to use the corrected fluxes provided by the FluxNet data providers (Pastorello et al., 2020).

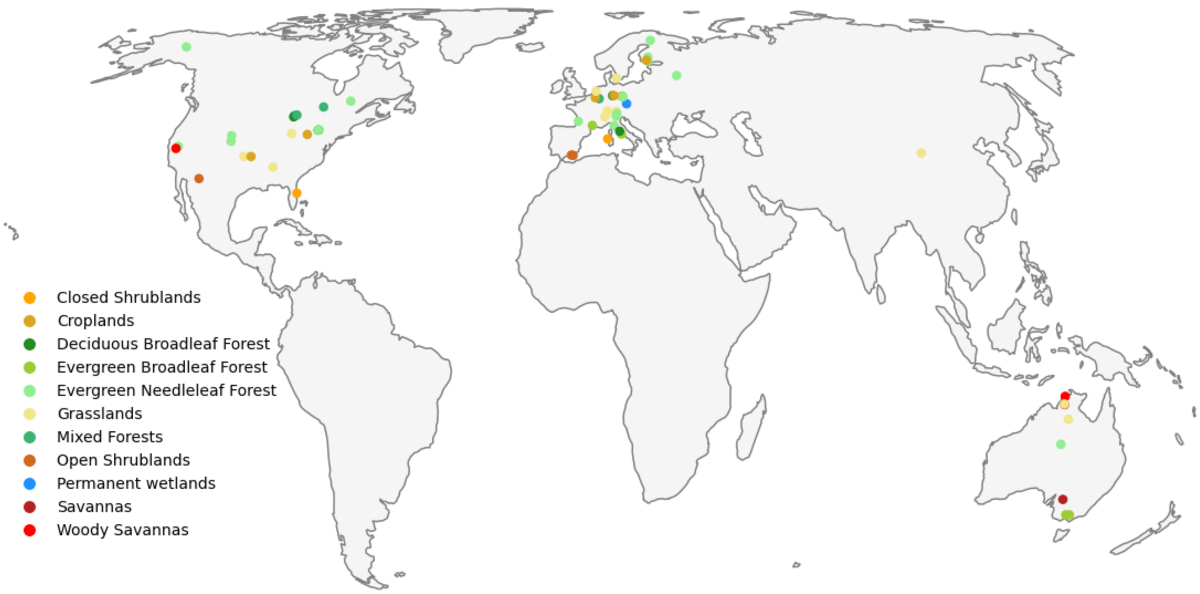

Figure 1. A map of the FluxNet sites used in the analysis, coded by the IGBP vegetation type.

2.2 SUMMA standalone simulations

We used the Structure for Unifying Multiple Modeling Alternatives (SUMMA) to simulate the hydrologic cycle (Clark et al., 2015) including the resulting turbulent heat fluxes. SUMMA is a hydrologic modeling framework that allows users to select between different model configurations and process parameterizations. The clean separation between the numerical solver and flux parameterizations allowed us to be confident that coupled DL parameterizations embedded into SUMMA did not affect any model components in unintentional ways. The core numerical solver in SUMMA enforces closure of the mass and energy balance and is used in all of our simulations.

SUMMA provides multiple flux parameterizations and process representations for many hydrologic processes. Because we were primarily interested in turbulent heat fluxes, we used a configuration for the other processes which would be suitable for general purpose hydrologic modeling, including runoff and snowpack simulations. For simulation of transpiration we used a Ball-Berry approach for simulating stomatal conductance (Ball et al., 1987), an exponentially decaying root density profile, and soil moisture controls that mimic the Noah land surface model (Niu et al., 2011). Similarly, the radiative transfer parameterizations which are the primary controls on the sensible heat fluxes are also set up to mimic the Noah land surface model. The functional forms of the turbulent heat fluxes in SUMMA is similar to many other land surface and hydrologic models, given by the bulk transfer equations (in resistance terms) as in Bonan (2015).

At each of the sites described in section 2.1 we independently calibrated a standalone SUMMA model using the dynamically dimensioned search algorithm (Tolson & Shoemaker, 2007) as implemented in the OSTRICH optimization package (Matott, 2017) using the mean squared error as the optimization criteria. A summary of the calibration variables and test ranges is shown in table S1 of the supporting information. The first year of available data was used for calibration. Because of the limited length of the data record at some sites, the calibration period was not excluded from subsequent analysis. The 10 parameters we chose to calibrate largely control water movement through the vegetation and soil domains. In the soil domain these include the residual and saturated moisture contents, field capacity, and controls on anisotropy of flows. In the vegetation domain these include controls on photosynthesis, rooting depth, wilting and transpiration water contents, amount of throughfall of precipitation through the canopy, and a generic scaling factor for the amount of vegetation.

The calibrations were run to a maximum of 500 trial iterations, which provided good convergence across sites (see the supporting information for convergence plots). We used the mean square error at a half hourly timestep for both the latent and sensible heat as the objective function and saved the best set of parameters for each site to use as our comparison to the DL parameterizations. To provide good estimates of the initial soil moisture and temperature states we spun up the standalone SUMMA simulations for 10 years both before and after calibration (for a total of 20 spinup years). We will refer to the standalone calibrated SUMMA simulations as SA (StandAlone) for the remainder of the paper. To summarize, we independently calibrated a set of parameters for each site, whose resulting best parameter set was used as an in-sample benchmark for comparison with our DL parameterizations. A brief description of the computational cost and runtimes associated with calibrating SA is provided in the supporting information.

2.3 DL parameterization and simulations

To build DL parameterizations of turbulent heat fluxes we constructed our neural networks using the Keras python package (Chollet , 2015). The neural networks take in a variety of input data such as meteorologic forcing data and output the bulk latent and sensible heat fluxes as shown in panel b) of figure 2.

Our neural networks were constructed using only dense layers where every node in one layer is connected to all nodes in the preceeding and following layers. We used the deep-dense architecture because it is the only network architecture that could easily be coupled to SUMMA, given the capabilities of the coupling tools. We will discuss the details of how we coupled the neural networks to SUMMA later in this section. We tested networks with as few as one layer and 12 nodes and up to 10 layers and 64 nodes were tested. After manual trial and error we settled on 6 layers each with 48 nodes. Smaller architectures were not as well able to capture the extremes of the turbulent heat fluxes and larger networks showed diminishing additional improvement. A simple schematic of the neural network architecture is shown in figure S2 of the supporting information.

We used hyperbolic tangent (tanh) activations in all of the nodes of the network. Stochastic gradient descent (SGD) with an exponential learning rate decay curve was used as the optimizer to train the weights and biases of the neural networks. We used the mean square error (the same as our objective function in the calibration of SA) in the 30-minute turbulent heat flux estimates as our loss function, similar to the objective function in our calibration of the SUMMA-SA simulations. Dropout was applied after the first layer and before the final layer with a retention rate of 0.9 to regularize. Dropout works by randomly pruning some fraction (one minus the retention rate) of the nodes in a given layer during training. This reduces the likelihood of overfitting the network as there is some stochasticity in the model architecture during training.

When training the networks we performed a 5-fold cross validation. We used 48 sites to train each network and then applied it out of sample to each of the remaining 12 sites. The data from the 48 sites used to train each network were randomly shuffled and split into 80% training and 20% validation data. The validation data was used to define an early stopping criterion for the training procedure where training was stopped if the validation loss was not decreased for 10 training epochs. This procedure keeps the model from overfitting on the training data. The maximum number of training epochs was set to 500 epochs, with a batch size of 768 data points (or 14 days of data points). All data was shuffled before training to remove any temporal bias that the model could learn, which also reduces overfitting.

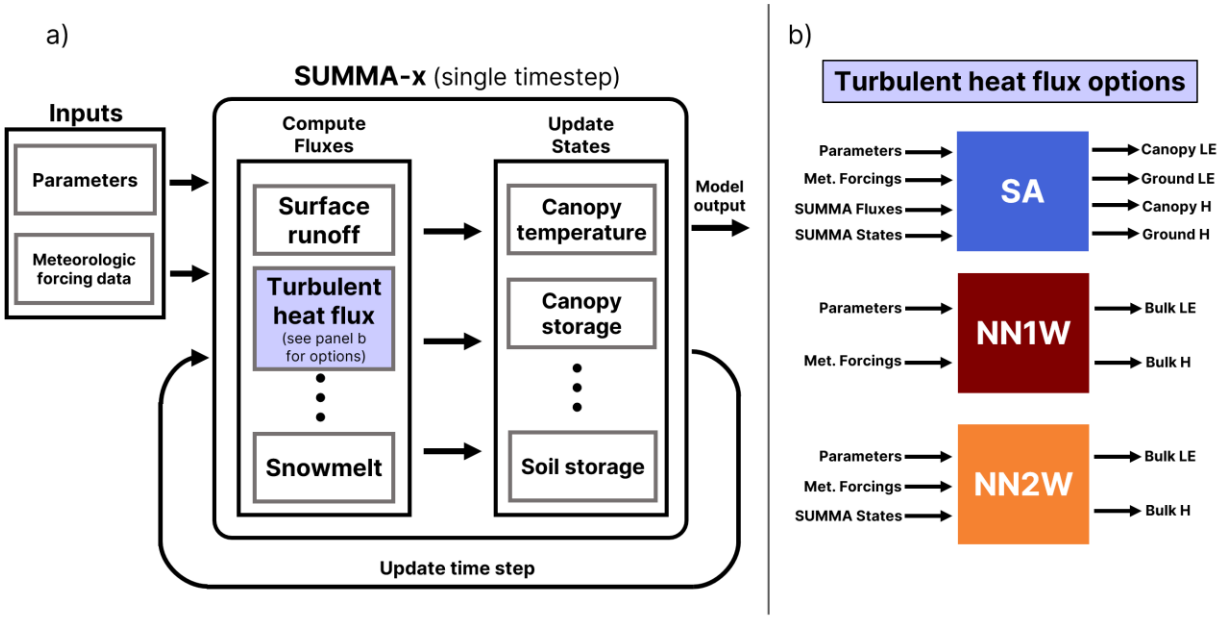

Figure 2. A schematic representation of the model setup. Panel a) shows the SUMMA runtime process. Parameters and meteorologic forcing data, as well as the state variables from the previous timestep, are fed to SUMMA to compute all fluxes, which are used to update the state variables for the subsequent timestep. The purple box labeled "Turbulent heat flux" highlights the process representation that we modify in our experiment. Panel b) shows the ways we represent the turbulent heat fluxes. One of the options from panel b) replaces the purple box in panel a). SA is the standalone SUMMA representation, as described in section 2.2. NN1W and NN2W are our DL-based representations described in section 2.3. Thus, SUMMA-x represents one of the three model configurations where x is one of SA, NN1W, or NN2W.

The first network we trained took meteorological forcing data for the current timestep, vegetation and soil types, and the calibrated SUMMA parameter values as input. We chose to include the calibration parameters to provide the same information to the neural networks as was provided to the calibrations, allowing for a more direct comparison and because the calibrated parameter values might be a proxy for site characteristics that can be associated with different responses among the sites. The neural network outputs the bulk latent and sensible heat fluxes at the half hourly timescale. We denote this network NN1W, for Neural-Network-1-Way, because this configuration only takes meteorological forcing data and parameters, which cannot be changed by the rest of the SUMMA calculations. That is, the neural network provides information about turbulent heat fluxes to SUMMA, but SUMMA does not provide any internally-derived information to the neural network.

The second network we trained took all of the same input data as the NN1W configuration, as well as a number of additional inputs that are derived states taken from the output of the coupled SUMMA-NN1W simulations. We included surface vapor pressure, leaf area index, surface soil layer volumetric water content, depth averaged transpirable water (as a volumetric fraction), surface soil layer temperature, depth averaged soil temperature, and a snow-presence indicator. These variables were chosen because they are used in the process-based SUMMA parameterizations for either latent or sensible heat, or affect the way in which the partitioning of the heat flux is distributed to the soil, vegetation, or snow domains. At runtime this network uses the additional variables as calculated internally by SUMMA, rather than the ones provided during training from NN1W. We denote this network NN2W, for Neural-Network-2-Way, because SUMMA internal states provide feedback to the ML model. That is, the neural network is provided inputs which are dependent on the state variables derived internally by SUMMA, which in turn depend on the turbulent heat fluxes that are predicted by the neural network.

After training each of these networks they were saved and translated into a format that could be loaded into Fortran via the Fortran Keras Bridge (FKB) package (Ott et al., 2020). The FKB package allows for translation of a limited subset of Keras model files (architecture, weights, biases, and activation functions) to be translated into a file format which can be loaded into the FKB Fortran library which implements several simple components for building and evaluating neural networks in Fortran, such as the deep-dense architecture used here.

We then extended SUMMA (which is written in Fortran) to allow for the use of these neural networks to simulate the turbulent heat fluxes. Normally SUMMA breaks the calculation of turbulent heat fluxes into several domains to delineate between heat exchanges in the vegetation and soil domains. Because we estimate these as bulk quantities we implemented this as only heat fluxes in the soil domain, and specified that the model should skip any computation of vegetation fluxes. We then specified that all ET resulting from the neural network's estimate of latent heat be taken from the soil domain as transpiration, according to SUMMA's internal routines. We chose this rather than taking all of the ET as soil evaporation because this allowed for a wider range of ET behaviors. In our simulations, the domain was split into nine soil layers, with a 0.01 m deep top layer. In SUMMA soil evaporation is only taken from the top soil layer and the shallow surface soil depth in our setup would not have allowed for sufficient storage to satisfy the predicted ET for many of the vegetated sites. Water removed as transpiration is weighted by the root density in each soil layer, which generally provides a large enough reservoir to satisfy the evaporative demand predicted by the neural networks. Another side-effect of our decision for taking all ET as transpiration is the removal of snow sublimation from the model entirely. As we will show in the results, the amount of snow sublimation in the SA simulations is negligible at most of our FluxNet sites, so we believe that this is an acceptable simplification for our initial demonstration. In cases where the neural network predicts greater evaporation than is available in the soil SUMMA enforces the water balance and limits the evaporation to an amount it can satisfy. A brief comparison of the computational cost and runtimes associated with training both NN1W and NN2W is provided in the supporting information.

3 Results

We present our results in two categories. First, we compare the performance of the coupled neural network simulations to the standalone calibrated simulations (SA). We use two commonly used metrics for determining the performance of the simulated turbulent heat fluxes, the Nash-Sutcliffe efficiency (NSE) and Kling-Gupta efficiency (KGE) scores. Using two metrics in tandem allows us to be sure that our results are robust (Knoben et al., 2019). Then, we explore how the inclusion of NN-based parameterizations for turbulent heat fluxes affects the overall model dynamics. This analysis is crucial to ensure that the new parameterizations do not lead to unrealistic simulations of other processes

3.1 Performance analysis

Figure 3 shows the cumulative density functions of the performance metrics across all sites, evaluated on the half-hourly data for all non-gap-filled periods. For all cases we see that both NN1W and NN2W outperformed the SA simulations. NN1W showed a median increase in NSE of 0.07 for latent heat and 0.12 for sensible heat, while NN2W showed a median increase in NSE of 0.10 for latent heat and 0.14 for sensible heat. Similarly, for KGE these were 0.10 (latent) and 0.21 (sensible) for NN1W and 0.17 (latent) and 0.23 (sensible) for NN2W. Examination of the individual KGE components (bias, variance, and correlation) shows that the NNs showed consistent improvements in all components. Overall we see that the NN2W configuration slightly outperforms the NN1W configuration. However, it is possible that in both cases that there are additional performance gains to be made with better model architectures and/or training procedures. We will come back to this in the Discussion.

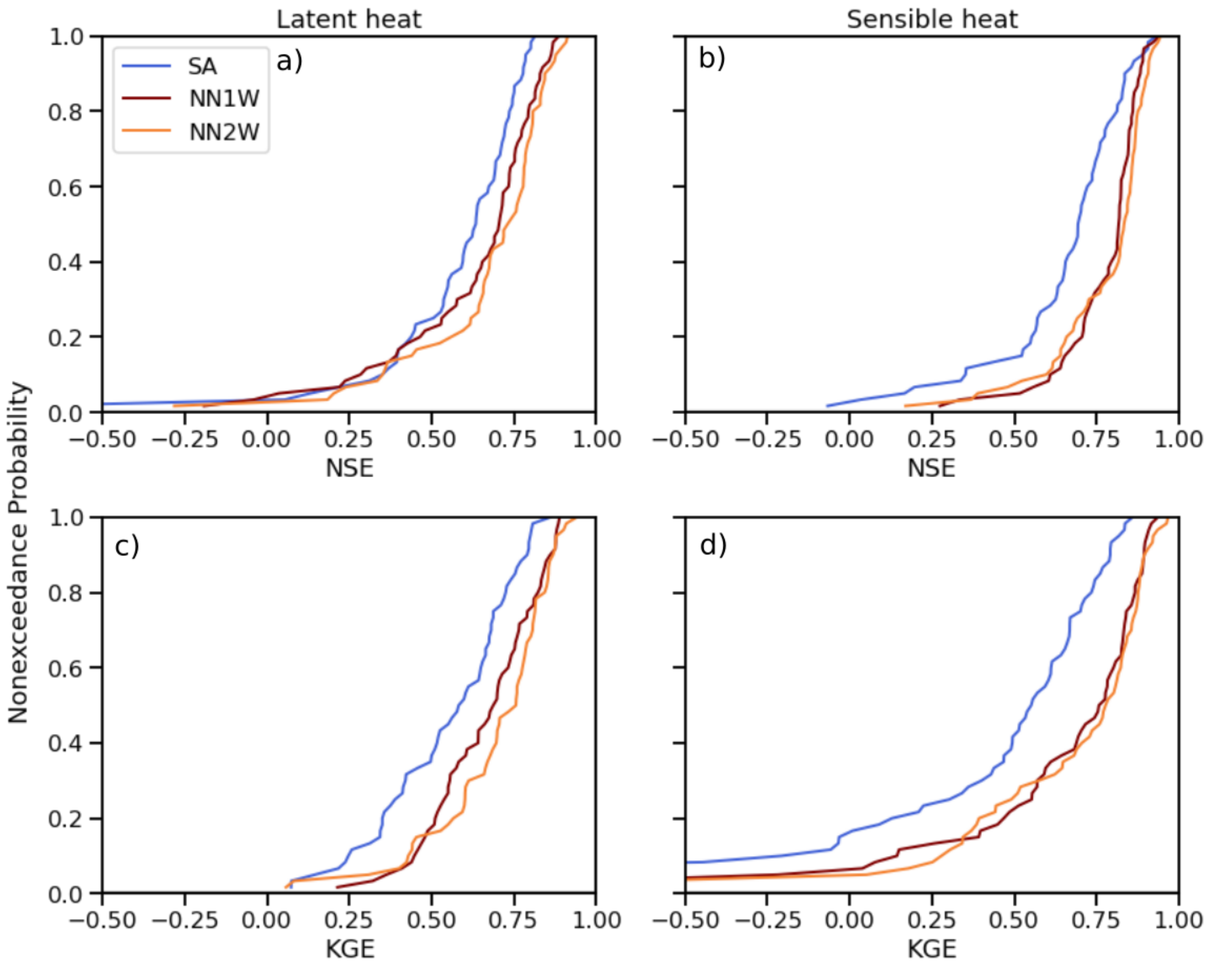

Figure 3. Empirical CDFs of performance measures for simulations across all sites. a) shows the NSE for latent heat, b) the NSE for sensible heat, c) the KGE for latent heat, and d) the KGE for sensible heat.

Even though the curves of the performance measures look quite similar between NN1W and NN2W, the performance differences from SA were not always perfectly correlated. Figure 3 shows the change in performance from SA for each site, ranked by SA performance. The maximum improvement that is possible is also shown to provide a reference to account for the fact that the range of both NSE and KGE is (-∞,1]. That is, there is more room for improvement for poorly performing sites than there is for well performing sites. For both performance measures and fluxes the general pattern of improvement follows the maximum improvement curve, with some added noise.

While on average the NN-based configurations performed better than the SA simulations, they performed worse at some locations. NN-based simulations generally had a higher NSE , but the KGE scores were more mixed for sensible heat, with SA outperforming the NN-based configurations at a number of sites. The NN-based configurations performed much worse at AT-Neu, DK-Eng, and CH-Cha (the outliers in the lowest 25th percentile of Figure 4d), where they failed in simulating large, upward, nighttime sensible heat fluxes. SA also performed poorly for these nighttime fluxes, but to a lesser extent. For latent heat, while some sites showed higher

NSE and KGE values for SA results than for the NN-based simulations, more sites showed poor performance across all configurations when evaluated by NSE. Decreases in performance relative to SA mostly occurred where the NN-based configurations consistently overestimated latent heat during winter, which most likely stems from our assumption that all latent heat is treated as transpiration. For both conditions for which SA outperformed the NN-based configurations, we believe that the performance of the NN-based configurations can be improved if more training data or more sophisticated ML methods were used, since the number of outliers was small and the average performance improvement was large.

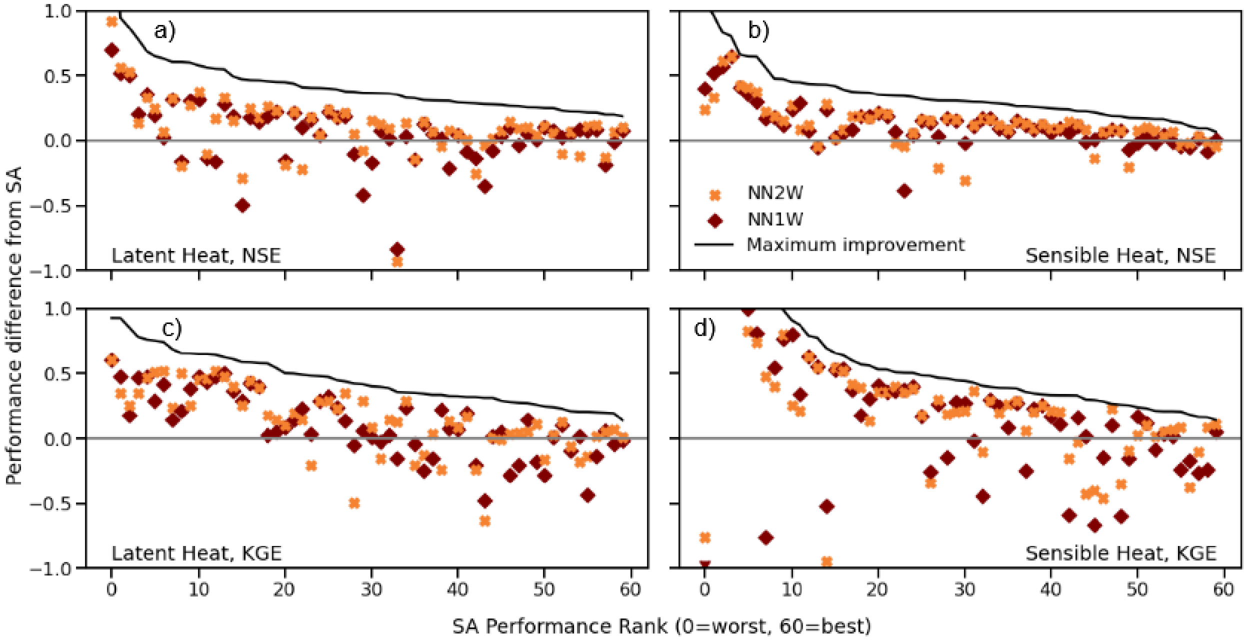

Figure 4. Scatter plots showing the performance of NN1W and NN2W against SA across all sites. Points above the grey zero line show configurations where the NN configuration improved performance over SA. The “Maximum improvement” line is based on the performance of the SA simulations, and is simply (1-NSE) in subplots a and b, and (1-KGE) in subplots c and d.

We also compared the KGE for different periods of temporal aggregation to evaluate whether performance improvements of the NN configurations persisted across timescales (Figure 5). The KGE score was chosen here because it shows greater variability than the NSE score in Figure 3, though the results are similar for NSE. We see that the sub-daily aggregations, on average, showed better performance for both NN configurations, demonstrating that they were able to capture the diurnal cycle of turbulent heat fluxes. This is mostly due to the strong dependence of turbulent heat fluxes on solar radiation, which we will further explore in section 3.2. Both NN1W and NN2W were able to outperform SA across all timescales for sensible heat.

However, at daily and longer temporal aggregations differences between models were seen in latent heat performance. The NN1W configuration performed better at sub-daily timescales than for daily or longer aggregations, for which performance was similar to SA. In contrast, the NN2W configuration performed better for latent heat than SA across all timescales.

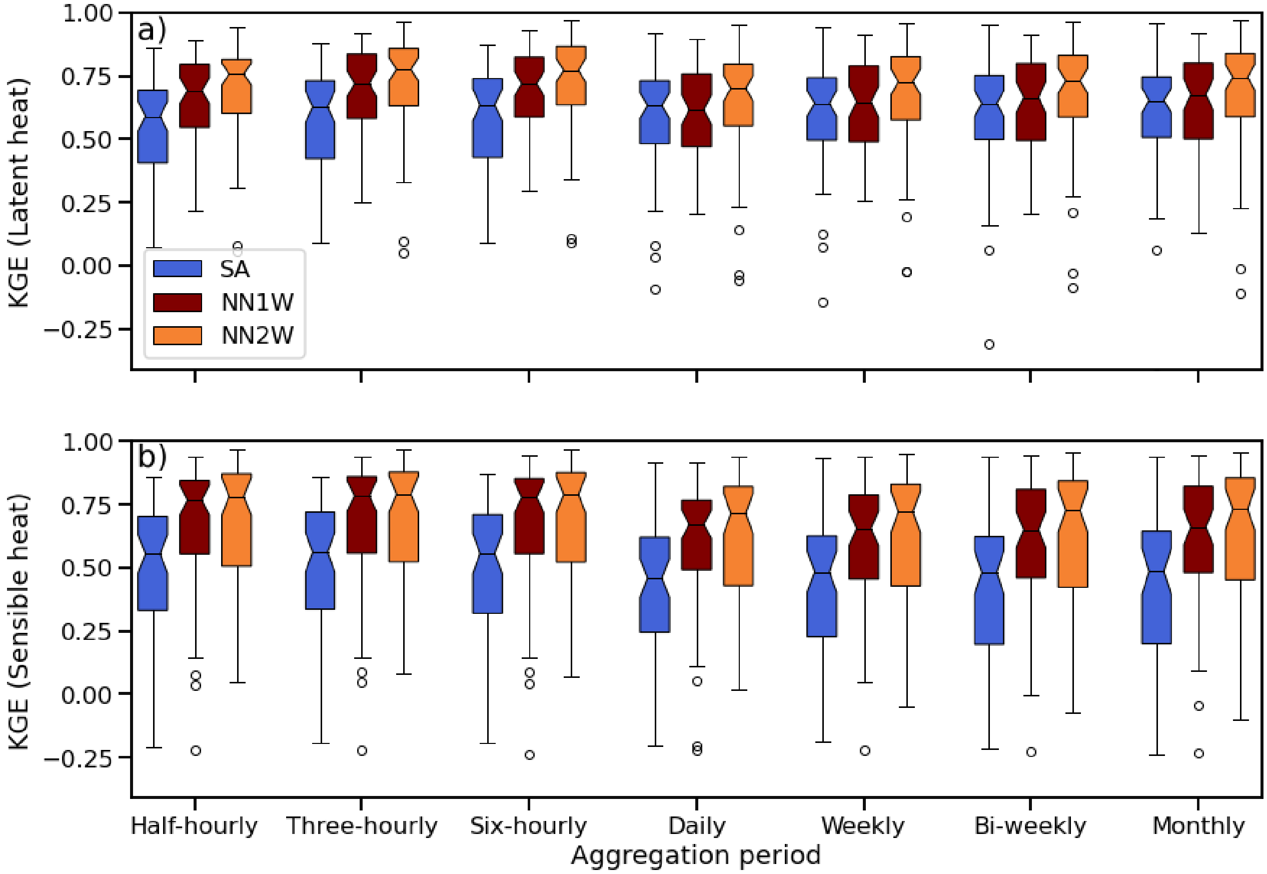

Figure 5. Performance of each model configuration for multiple temporal aggregations. Each box shows the interquartile range, with the median marked as the central line. A 95% confidence interval for the estimate of the median is represented by the notched portion. Outliers are shown as open circles.

3.2 Diagnostic analysis

In section 3.1 we demonstrated that the NN configurations were able to consistently outperform the SA configuration for both latent and sensible heat flux predictions at a half-hourly timestep. The range of performance differences shown in Figure 4 demonstrates that the NN-based simulations are significantly different from the physically-based representation in SA. Consequently, water and energy partitioning in the NN configurations is likely much different than in SA. To explore the effect of the new NN-based parameterizations on the simulated water cycle we first compared the simulated evaporative fraction (ET/P) to the observed (Figure 6). In all three model configurations the KGE values tend to be higher for sites where the simulated evaporative fraction closely matches the observed value.

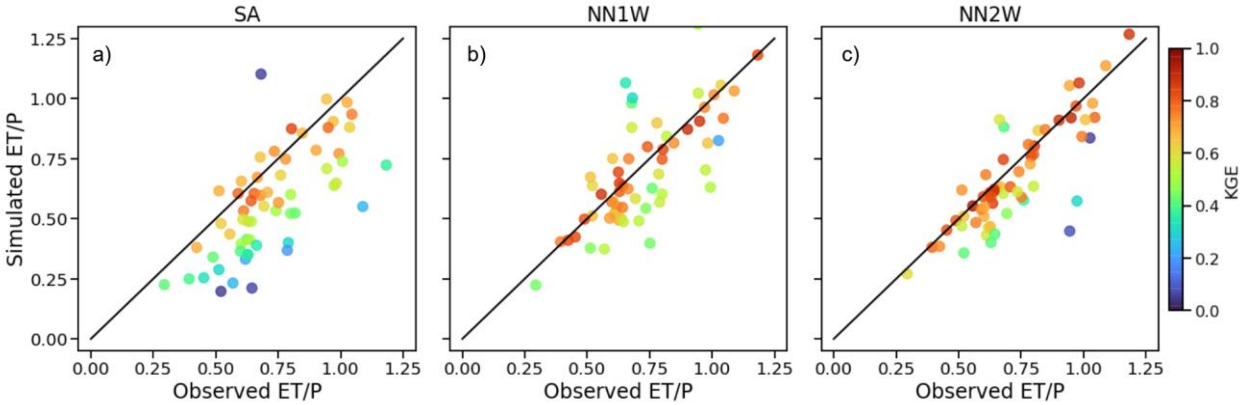

Figure 6. Comparison of evaporative fraction for each model configuration across all sites. The one-to-one line shows perfect correspondence with the observed values. Each point shows an individual site, averaged over the simulation period. Points are colored by their respective performance in terms of KGE of the latent heat at the half-hour timescale.

However, the SA configuration has a tendency to systematically underestimate total ET, while the NN configurations tend to match the observed evaporative fraction. The NN1W configuration shows more over-evaporation than NN2W, indicating that the introduction of soil states allows the model to perform better in moisture limiting conditions. This soil moisture feedback is the reason that the NN2W was able to perform better at daily and greater temporal aggregations for the prediction of latent heat. The impacts of these changes in the long-term evaporative fraction on the other terms of the water balance are shown in figure S3 of the supporting materials.

As noted when discussing Figure 5, we hypothesize that the NN-based simulations performed better at the sub-daily timescale because of their improved ability to model the diurnal cycle in the observations. We take the approach of Renner et al. (2019) by comparing the time lag in the diurnal cycle between the turbulent heat fluxes and shortwave radiation. To compute this we fitted a regression equation of the form:

$$Q(t) = a_0 + a_1 SW(t) + a_2 \frac{dSW(t)}{dt} + \epsilon, \qquad (1)$$

where Q is the turbulent heat flux, SW is the shortwave radiation, a_i are the coefficients of the regression, and ϵ is the residual term (Camuffo & Bernardi, 1982). Then, the phase lag can be computed as

$$\phi = tan^{-1}(2\pi a_2 / a_1 n_d), \qquad (2)$$

where n_d is the number of timesteps in a day (here, 48). We calculated this phase lag for each of the simulation configurations and the observations. Figure 7 shows how each of the simulations compare to the observed phase lag across all sites. For both latent and sensible heat we see that the NN-based configurations are better able to capture the diurnal phase lag seen in the observations, confirming our conclusion from Figure 5 that the improved sub-daily performance of the NN-based configurations is due to better representation of the diurnal cycle.

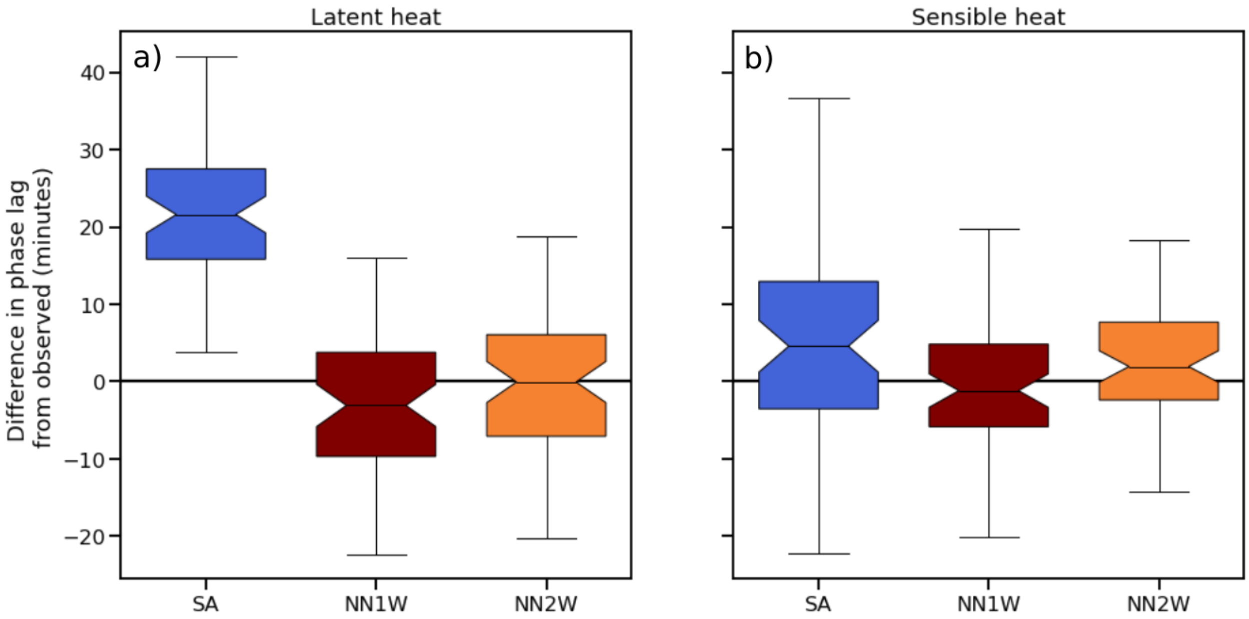

Figure 7. Difference in diurnal phase lag from observation. Positive values indicate that the simulated phase lag leads the observed phase lag.

4 Discussion

Our analysis shows that the DL parameterizations were able to outperform the standalone simulations for both latent and sensible heat fluxes. Most of the bulk gains in performance from the NN-based configurations stemmed from drastic improvements at sites where the SA configuration performed poorly. This is important to note, since our SA simulations were calibrated at site (and included the calibration period in the evaluation), while all NN-based simulations were trained out of sample in both time and space. This indicates that our NN-based configurations would likely be better able to represent turbulent heat fluxes in regions without measurements, implying that deep learning may be suitable for regionalization applications.

Both of the NN-based configurations represented the diurnal phase lag between shortwave radiation and turbulent heat fluxes better than SA. Renner et al. (2020) explored the ability of the land surface models used in the PLUMBER experiments (Best et al., 2015) to reproduce the observed diurnal phase lag, finding similar deviations from the observed phase lag as our SA simulations. This indicates that the NN-based approach has been able to learn something that has not been codified in PBHMs, and could provide better insight into how turbulent heat fluxes are generated at the scales that FluxNet towers operate. It is difficult to definitively state why the NN-based simulations provided more accurate simulations than SA’s process-based parameterizations. Even if the functional forms of the SA were correct, the model parameters may be difficult to determine. Zhao et al. (2019) were able to achieve good predictive performance out of a standalone (that is, not coupled to a larger model) machine-learning model that used a neural network to estimate the resistance term of the bulk transfer equations, and then computed the heat fluxes from the standard equations. Using such an approach would likely work well in the coupled setting as well.

We also found that the NN2W configuration maintained higher performance than either NN1W or SA at longer than daily timescales, as well as more accurately reproduced the observed long-term evaporative fraction. This indicates that the synergy between the deep-learned parameterization and the soil-moisture state evolution in SUMMA was able to better capture the long-term dynamics than either a purely machine-learned or purely process-based approach. This lends credibility to our proposition that the synergy between data-driven and physics-based approaches will likely lead to better simulations than a rigid adherence to either one of the methods by themselves.

These performance gains came at the cost of drastically simplifying the way in which we represented evapotranspiration. The SA simulations partition the latent heat fluxes amongst the soil, snow, and vegetation domains separately, while the NN simulations were set up to only represent the latent heat as a bulk flux, whose withdrawals we set to be taken from each soil layer according to the root density in that layer. This leads to the SA simulations being able to represent a more diverse range of conditions. While this was not a problem for the NN simulations on average, we were able to identify two locations where our simplification to the way in which ET is taken from the soil led to poor performance. At US-WCr and US-AR2 both NN configurations underestimated ET, because the soil was too dry to meet evaporative demand for much of the time. At these two sites the NN simulations performed significantly worse than the SA simulations, indicating a clear failure mode of the neural network based approach. This shortcoming might be be addressed by developing strategies that better partition the latent heat fluxes amongst the soil, snow, and vegetation domains. This would also allow for adding snow sublimation back in, reducing the number of modifications which must be made to SUMMA in order to run with an embedded neural network.

Other neural network architectures will likely lead to further performance improvements. Many recent studies that used neural networks to predict hydrologic systems have shown that Long-Short-Term-Memory (LSTM) networks are superior at learning timeseries behaviors compared to the methods used here (Feng et al., 2020; Frame et al., 2020; Jiang et al., 2020; Kratzert et al., 2018). Convolutional neural networks (CNN) have been used extensively to learn from spatially distributed fields (Geng & Wang, 2020; Kreyenberg et al., 2019; Liu & Wu, 2016; Pan et al., 2019). To take advantage of these specialized architectures in existing PBHMs like SUMMA will require the investment in tools and workflows. As of the time of writing, the FKB library only supports densely connected layers, and a few simple activation and loss functions. Implementing these layers in the FKB library, or some other framework that can be used to couple ML models with PBHMs, would open many possibilities for future research. Additionally, implementing more specialized activation functions and loss functions (such as NSE or KGE) will offer more flexibility for a wider range of applications.

Alongside better tools for incorporating machine learning into process-based models, the development and identification of workflows to perform machine and deep learning tasks will be necessary for wider adoption in the field. For instance, we initially trained the NN2W networks using the SA soil states, which were drastically different from the spun up states in the NN configurations. This led to almost identical performance in the NN1W and NN2W simulations, since the soil state information from the SA simulations was very different from what the network saw during training. Only after realizing this and training the NN2W on the states predicted by the NN1W simulations were we able to achieve better performance out of the NN2W simulations. Understanding whether there is a sort of iterative train-spinup-train workflow that balances overfitting and provides representative training data will be important for future studies.

Similarly, it is unclear whether there would be significant difficulties in trying to calibrate either of the NN-based models in new basins like we did for the SA simulations. Particularly, we do not know if the output of the neural networks is sensitive to the values of the calibration parameters. Our decision to include the calibrated parameter values in the training of the NN-based configurations was to provide the same types of information to both optimization procedures. In future studies it may be worthwhile to explore whether these parameters are necessary, or how regionalization of data driven approaches should best be codified. It is also unclear whether our NN-based configurations are able to be calibrated efficiently for other processes such as streamflow.

Finally, model architectures that separate process parameterizations in as clean a way as possible will allow for more robust and rapid development of ML parameterizations of other processes. Building modular and general purpose ways to incorporate machine learning into process-based models will allow researchers to more efficiently evaluate different approaches. Exploring and answering these practical questions will likely lead to community accepted practices which can be adopted to accelerate research of other applications.

5 Conclusions

We have shown that coupling DL parameterizations for prediction of turbulent heat fluxes into a PBHM outperforms existing physically-based parameterizations while maintaining mass and energy balance. We were able to couple our neural networks into SUMMA in two different ways, which both showed significant performance improvements when performed out of sample over the at-site calibrated standalone SUMMA simulations. The one-way coupling (NN1W), despite being conceptually simpler and not taking any model states as inputs, was able to improve simulations almost as much as the more complex two-way coupling (NN2W) at the sub-daily timescale. Both of the new parameterizations better represent the observed diurnal cycles and NN2W was better able to represent the long-term evaporative fraction as well as both turbulent heat fluxes at longer than daily timescales. We found that NN1W was also able to accurately predict sensible heat fluxes at greater than daily timescales, indicating that even “simple” DL parameterizations show great promise for coupling into PBHMs.

While we consider our new parameterizations a step forward in incorporating ML techniques into traditional process-based modeling, we have only scratched the surface on many of the different avenues which will surely be explored. We used the simplest possible network architecture, a deep-dense network. For spatial applications we suspect that CNN layers will prove invaluable. Recurrent layers such as LSTMs have been dominant in the timeseries domain. More sophisticated architectures such as neural ordinary differential equations (Ramadhan et al., 2020) or those discovered through neural architecture search (Geng & Wang, 2020) are bound to be both more efficient and interpretable than our dense networks. The opportunities for incorporating and learning from ML-based models into the hydrologic sciences are virtually untapped. We believe that as the community builds tools and workflows around the existing ML ecosystems we will be able to unlock this potential.

Acknowledgments, Samples, and Data

We would like to thank Yifan Cheng and Yixin Mao for reading and commenting on an early version of this manuscript. Their comments improved the clarity and framing of our work. The code to process, configure, calibrate/train, run, and analyze the FluxNet data is available at https://doi.org/10.5281/zenodo.4300929. The SUMMA model configuration for SA is available at https://doi.org/10.5281/zenodo.4300931. The SUMMA model configuration for NN1W is available at https://doi.org/10.5281/zenodo.4300932. The SUMMA model configuration for NN2W is available at https://doi.org/10.5281/zenodo.4300933. We would like to acknowledge high-performance computing support from Cheyenne (doi:10.5065/D6RX99HX) provided by NCAR's Computational and Information Systems Laboratory, sponsored by the National Science Foundation.

References

Ball, J. T., Woodrow, I. E., & Berry, J. A. (1987). A Model Predicting Stomatal Conductance and its Contribution to the Control of Photosynthesis under Different Environmental Conditions. In J. Biggins (Ed.), *Progress in Photosynthesis Research: Volume 4 Proceedings of the VIIth International Congress on Photosynthesis Providence, Rhode Island, USA, August 10–15, 1986* (pp. 221–224). Dordrecht: Springer Netherlands. https://doi.org/10.1007/978-94-017-0519-6_48

Best, M. J., Abramowitz, G., Johnson, H. R., Pitman, A. J., Balsamo, G., Boone, A., et al. (2015). The Plumbing of Land Surface Models: Benchmarking Model Performance. *Journal of Hydrometeorology*, *16*(3), 1425–1442. https://doi.org/10.1175/JHM-D-14-0158.1

Bonan, G. (2015). *Ecological Climatology: Concepts and Applications*. Cambridge University Press.

Brenowitz, N. D., & Bretherton, C. S. (2018). Prognostic Validation of a Neural Network Unified Physics Parameterization. *Geophysical Research Letters*, *45*(12), 6289–6298. https://doi.org/10.1029/2018GL078510

Camuffo, D., & Bernardi, A. (1982). An observational study of heat fluxes and their relationships with net radiation. *Boundary-Layer Meteorology*, *23*(3), 359–368. https://doi.org/10.1007/BF00121121

Chollet, F. (2015). Keras. Retrieved from https://github.com/fchollet/keras

Clark, M. P., Nijssen, B., Lundquist, J. D., Kavetski, D., Rupp, D. E., Woods, R. A., et al. (2015). A unified approach for process-based hydrologic modeling: 1. Modeling concept: A unified approach for process-based hydrologic modeling. *Water Resources Research*, *51*(4), 2498–2514. https://doi.org/10.1002/2015WR017198

Dee, D. P., Uppala, S. M., Simmons, A. J., Berrisford, P., Poli, P., Kobayashi, S., et al. (2011). The ERA-Interim reanalysis: configuration and performance of the data assimilation system. *Quarterly Journal of the Royal Meteorological Society*, *137*(656), 553–597. https://doi.org/10.1002/qj.828

Feng, D., Fang, K., & Shen, C. (2020). Enhancing streamflow forecast and extracting insights using long-short term memory networks with data integration at continental scales. *ArXiv:1912.08949 [Cs, Stat]*. Retrieved from http://arxiv.org/abs/1912.08949

Foken, T. (2008). The Energy Balance Closure Problem: An Overview. *Ecological Applications*, *18*(6), 1351–1367. https://doi.org/10.1890/06-0922.1

Frame, J., Nearing, G., Kratzert, F., & Rahman, M. (2020). *Post processing the U.S. National Water Model with a Long Short-Term Memory network* (preprint). EarthArXiv. https://doi.org/10.31223/osf.io/4xhac

Geng, Z., & Wang, Y. (2020). Automated design of a convolutional neural network with multi-scale filters for cost-efficient seismic data classification. *Nature Communications*, *11*(1), 3311. https://doi.org/10.1038/s41467-020-17123-6

Hu, C., Wu, Q., Li, H., Jian, S., Li, N., & Lou, Z. (2018). Deep Learning with a Long Short-Term Memory Networks Approach for Rainfall-Runoff Simulation. *Water*, *10*(11), 1543. https://doi.org/10.3390/w10111543

Jiang, S., Zheng, Y., & Solomatine, D. (2020). Improving AI System Awareness of Geoscience Knowledge: Symbiotic Integration of Physical Approaches and Deep Learning. *Geophysical Research Letters*, *47*(13), e2020GL088229. https://doi.org/10.1029/2020GL088229

Jung, M., Reichstein, M., & Bondeau, A. (2009). Towards global empirical upscaling of FLUXNET eddy covariance observations: validation of a model tree ensemble approach using a biosphere model, 13.

Kidston, J., Brümmer, C., Black, T. A., Morgenstern, K., Nesic, Z., McCaughey, J. H., & Barr, A. G. (2010). Energy Balance Closure Using Eddy Covariance Above Two Different Land Surfaces and Implications for CO2 Flux Measurements. *Boundary-Layer Meteorology*, *136*(2), 193–218. https://doi.org/10.1007/s10546-010-9507-y

Knoben, W. J. M., Freer, J. E., & Woods, R. A. (2019). Technical note: Inherent benchmark or not? Comparing Nash–Sutcliffe and Kling–Gupta efficiency scores. *Hydrology and Earth System Sciences*, *23*(10), 4323–4331. https://doi.org/10.5194/hess-23-4323-2019

Kratzert, F., Klotz, D., Brenner, C., Schulz, K., & Herrnegger, M. (2018). Rainfall-Runoff modelling using Long-Short-Term-Memory (LSTM) networks. *Hydrology and Earth System Sciences Discussions*, 1–26. https://doi.org/10.5194/hess-2018-247

Kreyenberg, P. J., Bauser, H. H., & Roth, K. (2019). Velocity Field Estimation on Density-Driven Solute Transport With a Convolutional Neural Network. *Water Resources Research*, *55*(8), 7275–7293. https://doi.org/10.1029/2019WR024833

Lathière, J., Hauglustaine, D. A., & Friend, A. D. (2006). Impact of climate variability and land use changes on global biogenic volatile organic compound emissions. *Atmos. Chem. Phys.*, 19.

Li, L., Wang, Y.-P., Yu, Q., Pak, B., Eamus, D., Yan, J., et al. (2012). Improving the responses of the Australian community land surface model (CABLE) to seasonal drought. *Journal of Geophysical Research: Biogeosciences*, *117*(G4). https://doi.org/10.1029/2012JG002038

Liu, Y., & Wu, L. (2016). Geological Disaster Recognition on Optical Remote Sensing Images Using Deep Learning. *Procedia Computer Science*, *91*, 566–575. https://doi.org/10.1016/j.procs.2016.07.144

Matott, L. S. (2017). OSTRICH: an Optimization Software Tool, Documentation and User's Guide, Version 17.12.19. University at Buffalo Center for Computational Research. Retrieved from www.eng.buffalo.edu/~lsmatott/Ostrich/OstrichMain.html

Moshe, Z., Metzger, A., Elidan, G., Kratzert, F., Nevo, S., & El-Yaniv, R. (2020). HydroNets: Leveraging River Structure for Hydrologic Modeling. Retrieved from https://arxiv.org/abs/2007.00595v1

Musselman, K. N., Clark, M. P., Liu, C., Ikeda, K., & Rasmussen, R. (2017). Slower snowmelt in a warmer world. *Nature Climate Change*, *7*(3), 214–219. https://doi.org/10.1038/nclimate3225

Nearing, G. S., Kratzert, F., Sampson, A. K., Pelissier, C. S., Klotz, D., Frame, J. M., et al. (2020). What Role Does Hydrological Science Play in the Age of Machine Learning? *Water Resources Research*, e2020WR028091. https://doi.org/10.1029/2020WR028091

Niu, G.-Y., Yang, Z.-L., Mitchell, K. E., Chen, F., Ek, M. B., Barlage, M., et al. (2011). The community Noah land surface model with multiparameterization options (Noah-MP): 1. Model description and evaluation with local-scale measurements. *Journal of Geophysical Research: Atmospheres*, *116*(D12). https://doi.org/10.1029/2010JD015139

Ott, J., Pritchard, M., Best, N., Linstead, E., Curcic, M., & Baldi, P. (2020). A Fortran-Keras Deep Learning Bridge for Scientific Computing. *ArXiv:2004.10652 [Cs]*. Retrieved from http://arxiv.org/abs/2004.10652

Pan, B., Hsu, K., AghaKouchak, A., & Sorooshian, S. (2019). Improving Precipitation Estimation Using Convolutional Neural Network. *Water Resources Research*, *55*(3), 2301–2321. https://doi.org/10.1029/2018WR024090

Pastorello, G., Trotta, C., Canfora, E., Chu, H., Christianson, D., Cheah, Y.-W., et al. (2020). The FLUXNET2015 dataset and the ONEFlux processing pipeline for eddy covariance data. *Scientific Data*, *7*(1), 225. https://doi.org/10.1038/s41597-020-0534-3

Ramadhan, A., Marshall, J., Souza, A., Wagner, G. L., Ponnapati, M., & Rackauckas, C. (2020). Capturing missing physics in climate model parameterizations using neural differential equations. *ArXiv:2010.12559 [Physics]*. Retrieved from http://arxiv.org/abs/2010.12559

Rasp, S., Pritchard, M. S., & Gentine, P. (2018). Deep learning to represent subgrid processes in climate models. *Proceedings of the National Academy of Sciences*, *115*(39), 9684–9689. https://doi.org/10.1073/pnas.1810286115

Renner, M., Brenner, C., Mallick, K., Wizemann, H.-D., Conte, L., Trebs, I., et al. (2019). Using phase lags to evaluate model biases in simulating the diurnal cycle of evapotranspiration: a case study in

Luxembourg. *Hydrology and Earth System Sciences*, *23*(1), 515–535. https://doi.org/10.5194/hess-23-515-2019

Renner, M., Kleidon, A., Clark, M., Nijssen, B., Heidkamp, M., Best, M., & Abramowitz, G. (n.d.). How well can land-surface models represent the diurnal cycle of turbulent heat fluxes? *Journal of Hydrometeorology*, 1–56. https://doi.org/10.1175/JHM-D-20-0034.1

Shen, C. (2018). A Transdisciplinary Review of Deep Learning Research and Its Relevance for Water Resources Scientists. *Water Resources Research*, *54*(11), 8558–8593. https://doi.org/10.1029/2018WR022643

Tolson, B. A., & Shoemaker, C. A. (2007). Dynamically dimensioned search algorithm for computationally efficient watershed model calibration. *Water Resources Research*, *43*(1). https://doi.org/10.1029/2005WR004723

Tramontana, G., Jung, M., Schwalm, C. R., Ichii, K., Camps-Valls, G., Ráduly, B., et al. (2016). Predicting carbon dioxide and energy fluxes across global FLUXNET sites with regression algorithms. *Biogeosciences*, *13*(14), 4291–4313. https://doi.org/10.5194/bg-13-4291-2016

Wilson, K., Goldstein, A., Falge, E., Aubinet, M., Baldocchi, D., Berbigier, P., et al. (2002). Energy balance closure at FLUXNET sites. *Agricultural and Forest Meteorology*, *113*(1), 223–243. https://doi.org/10.1016/S0168-1923(02)00109-0

Zhao, W. L., Gentine, P., Reichstein, M., Zhang, Y., Zhou, S., Wen, Y., et al. (2019). Physics-Constrained Machine Learning of Evapotranspiration. *Geophysical Research Letters*, *46*(24), 14496–14507. https://doi.org/10.1029/2019GL085291