An Efficient Bayesian Method for Advancing the Application of Deep Learning in Earth Science

Dan Lu
Oak Ridge National Laboratory
Oak Ridge, TN, USA
lud1@ornl.gov

Siyan Liu
University of Kansas
Lawrence, KS, USA
liusiyan@ku.edu

Daniel Ricciuto
Oak Ridge National Laboratory
Oak Ridge, TN, USA
ricciutodm@ornl.gov

***Abstract*—Despite their wide and successful applications, deep learning (DL) models are prone to overfitting for small training datasets, produce a poor predictive performance for uncertain data, and provide point estimations without any indication of the accuracy and credibility. These limitations of the deterministic DL models hinder their effective application in Earth system science where the labelled data are sparse, noisy and incomplete with large uncertainty and where the predictive uncertainty quantification is needed for scientific understanding and policy decision making. Integration of Bayesian inference into DL models adds an estimate of uncertainty and regularization in the predictions. However, traditional Bayesian methods are computationally unaffordable and inflexible for high-dimensional problems, which limits their application to DL systems that typically have millions of model parameters. In this effort, we propose an efficient and general-purpose Bayesian inference method to advance DL model optimization and uncertainty quantification, so as to facilitate the adoption of DL in Earth sciences. In a demonstration, we integrate the proposed Bayesian method with a feedforward neural network (NN) to build a fast-to-evaluate surrogate of the complex Energy Exascale Earth System Land Model for efficient modeling. The formulated Bayesian NN, using a small number of training data, produces an accurate prediction with high credibility, whereas with the same small training size, the deterministic NN cannot yield a reasonable estimation and does not provide confidence information. The proposed Bayesian method is computationally efficient and flexible, capable of integration with diverse network variants such as convolutional NNs and recurrent NNs to advance the application of DL in Earth sciences.**

***Index Terms*—Deep learning, Bayesian inference, Uncertainty quantification, Earth system science application**

I. Introduction

With the advent of better sensing technologies (e.g., remote sensing satellites and in-situ sensors in space, in the sea, and on the land) and the improvements in computational resources for running large-scale simulations of Earth system models, Earth science has witnessed a major revolution in transitioning from a data-poor field to an increasingly data-rich field. The growing availability of Earth science data offers immense potential for machine learning (ML)/deep learning (DL) to tackle geoscientific problems and advance the Earth science development and predictive capability.

ML in general and DL in particular have resulted in major success stories in a wide range of commercial applications, such as computer vision, speech recognition, and natural language translation. Attributed to the space-time nature of geoscientific data, Earth science problems share some similarities with those in the commercial applications, which is facilitating the adoption of DL in Earth sciences. For example, feedforward neural networks (FNNs) have been used in terrestrial ecosystem modeling to predict carbon fluxes at a regional scale for a long time period [1]. Convolutional neural networks (CNNs) have been used to recognize features such as tornadoes, hurricanes, and atmospheric rivers from remote sensing data. CNNs have also been applied to detect extreme weather events from climate model simulations [2]. Recurrent neural networks (RNNs) have been explored to investigate memory effects of climate and vegetation on net ecosystem carbon fluxes in global forests from the combined remote sensing, climate, and eddy-covariance flux data [3]. And RNNs have also been used in hydrological modeling to predict streamflow discharge for a variety of catchments [4].

Despite these successful studies, numerous challenges exist in application of deep neural networks (DNNs) in Earth sciences [5]. First, Earth system variables have spatiotemporal structure, are highly multivariate, follow nonlinear relationships, show non-stationary characteristics, and often involve rare but high-impact events. In addition, Earth system data are often available from different sources, at varying spatial and temporal resolutions, with varying degrees of noise, incompleteness, and uncertainty. Last but not least, large labelled geoscientific datasets with ground truth do not always exist due to the difficulties and costs in labelling the data.

On the other hand, although DNNs have the capacity to learn and model complex nonlinear relationships, they suffer from two basic limitations [6]. First, DNNs have unsatisfactorily low predictive performance when there exists uncertainty in the data. This weakness is due to the theoretical fact that DNNs generate averaged values of targets conditioned on inputs and have difficulties in properly and reliably predicting rare events. These difficulties cannot be mitigated through changing the model structure or repeating the training process. The data uncertainty will eventually be propagated to the prediction and impair predictive credibility. Second, DNNs produce frequentist inference in nature, which suggests that DNNs require substantial amounts of data to train on and are prone to overfitting. More importantly, DNNs only provide point predictions without any indication of their accuracy and credibility. These point predictions become less reliable

and accurate if the training data are sparse, or if targets are multivalued and affected by probabilistic events. To prevent overfitting, various regularization techniques have been used such as early stopping, weight decay, $L1$ or $L2$ regularizations and currently the most popular dropout. However, these techniques do not quantify the uncertainty. To evaluate the influence of data uncertainty on predictions and to improve the predictive credibility and reliability, prediction uncertainties of the DNN models should be quantified and provided. It is important to know how well the predictions generated by DNNs match the real targets and how large the risk of mismatch is.

Bayesian inference, enabling explicit representation and propagation of uncertainties, is well positioned to address the aforementioned challenges of the traditional deterministic DNNs. Application of the Bayesian inference to DNNs adds an estimate of uncertainty and regularization in the predictions making Bayesian DNNs particularly suitable to solve geoscientific problems with sparse and uncertain labelled data. The resulting Bayesian neural network (BNN) allows the model to express uncertainty via its parameters in form of probability distributions. Meanwhile, by using a prior probability distribution to integrate out the parameters, we compute the average across many models during training, which gives a regularization effect to the network, thus preventing overfitting.

However, applying Bayesian inference to quantify the uncertainties of DNNs is very challenging because of the curse of dimensionality. For example, a typical DNN can easily have hundreds of thousands of parameters. Classical Bayesian approximation methods such as Markov chain Monte Carlo sampling and Hybrid Monte Carlo are not suitable due to their slow convergence and difficulties in scaling up to high dimensions. Recently, variational inference (VI) approaches have demonstrated successful applications in highly complex learning systems [7]. VI frames the Bayesian inference into a deterministic optimization problem that approximates the target distribution with a simpler distribution from a predefined parametric family by minimizing their Kullback–Leibler (KL) divergence. This formulation makes VI efficiently solvable by using well-developed optimization techniques and easily scalable to high-dimensional problems. However, the accuracy and computational costs of the VI methods critically depend on the choice of the parametric families and they generally do not provide consistent approximation like the sampling methods.

In this effort, we propose an efficient Bayesian inference method to advance DNN model optimization and uncertainty quantification (UQ), so as to facilitate the adoption of DL in Earth sciences. The proposed Bayesian method is developed based on the Stein variational gradient descent (SVGD) algorithm [8]. In SVGD, we start with a set of initial particles and then iteratively update the particles using adaptively constructed deterministic variable transforms that guarantee to push the particles closer to the target distribution, in the sense that the KL divergence between the distribution of the particles and the target distribution can be iteratively decreased based on the gradient descent. The SVGD algorithm is variational inference in nature, so it converges faster than sampling methods and is more suitable to solve complex and high-dimensional DNN optimization and UQ problems. But unlike the traditional VI algorithms, SVGD is nonparametric and does not limit to the specific variational distribution assumptions. This general-purpose property makes SVGD widely applicable for different kinds of DL models. For example, we can integrate SVGD into FNNs to advance ML regression analysis, into CNNs to improve image classification, and into RNNs for better time series simulation. Additionally, SVGD does not need ML experts to design the parametric variational distributions on a model-by-model basis which makes it readily accessible to domain scientists. Furthermore, SVGD leverages merits of the Monte Carlo sampling making it produce consistent approximations. In implementation, SVGD has a simple form and can be applied whenever gradient descent can be calculated. The performance of SVGD does not depend on the initial distribution, although it starts with a set of initial particles. And it can be shown that with only one particle SVGD is able to locate the maximum a posteriori and with more particles it automatically becomes a full Bayesian approach.

In this study, we integrate the SVGD algorithm into a feedforward network and apply the formulated BNN to build a fast-to-evaluate surrogate model of the Energy Exascale Earth System Land Model (ELM) for efficient modeling. The performance of the BNN is evaluated against the complex ELM simulation and compared with the deterministic NNs. In the following, we first introduce the BNN and the SVGD algorithm, then describe the application of the BNN for surrogate modeling, next we analyze the results and assess the BNN performance, and last we discuss possible applications of BNN in Earth sciences and its future development.

II. Methodology

In this section, we first describe the BNN in general, and then introduce the SVGD algorithm to implement the BNN in detail. After that, we discuss how to use the BNN to quantify the model prediction uncertainty.

A. Bayesian neural network (BNN)

We describe the BNN model based on a fully connected feedforward neural network. Given data $\mathcal{D} = \{\mathbf{x}_n, y_n\}_{n=1}^N$, which is made up of D-dimensional feature vectors $\mathbf{x}_n \in \mathbb{R}^D$ and corresponding scalar target variables $y_n \in \mathbb{R}$, a deterministic NN can be used to learn the relationship between $\mathbf{X}$ and $\mathbf{y}$ through $\mathbf{y} = f(\mathbf{X}, \mathbf{W})$ with $\mathbf{W}$ representing all parameters in the network including weights and biases. The BNN treats the parameters $\mathbf{W}$ as random variables instead of deterministic unknowns to account for epistemic uncertainty induced by lack of training data. In addition, BNNs introduce an additive error $\boldsymbol{\epsilon}$ to model aleatoric uncertainty caused by the data noise. This additive error also enables the probabilistic model to have an explicit likelihood function depending on the error distribution. Therefore, the BNN can be formulated as $\mathbf{y} = f(\mathbf{X}, \mathbf{W}) + \boldsymbol{\epsilon}$, where $f(\mathbf{X}, \mathbf{W})$ is the output of the network and $\boldsymbol{\epsilon}$ is the additive error.

We assume the data errors are Gaussian distributed with an identical variance γ^{-1} for every input sample, i.e., $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \gamma^{-1}\mathbf{I})$. Based on the pairs of input-output data $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$,the likelihood function for the BNN weights $\mathbf{W}$ and the error precision γ, can be formulated as

$$p\big(\mathbf{y}|\mathbf{X}, \mathbf{W}, \gamma\big) = \prod_{n=1}^{N} \mathcal{N}\big(y_n | f(\mathbf{x}_n, \mathbf{W}), \gamma^{-1}\big). \quad (1)$$

Given the large amount of uninterpretable parameters in the DNN, there are not many choices of $\mathbf{W}$ priors due to the limitation of computational costs and memory. Here we assume a fully factorized Gaussian prior with zero mean and Gamma-distributed precision λ on parameters $\mathbf{W}$, then we have

$$p(\mathbf{W}|\lambda) = \mathcal{N}(\mathbf{W}|\mathbf{0}, \lambda^{-1}\mathbf{I}), p(\lambda) = Gamma(\lambda|a_0, b_0). \quad (2)$$

This results in a student's t-prior for $\mathbf{W}$, which has heavy tails and more mass close to zero. We assume the additive error precision γ is also Gamma distributed, i.e., $p(\gamma) = Gamma(\gamma|a_1, b_1)$. And we set the hyperparameters $a_0 = a_1 = 1.0$ and $b_0 = b_1 = 0.1$ in this study.

The posterior distribution for the parameters $\mathbf{W}$, γ and λ can then be obtained by Bayes' rule:

$$p(\mathbf{W}, \gamma, \lambda|\mathcal{D}) = \frac{p\big(\mathbf{y}|\mathbf{X}, \mathbf{W}, \gamma\big)p(\mathbf{W}|\lambda)p(\lambda)p(\gamma)}{p(\mathbf{y}|\mathbf{X})}, \quad (3)$$

where $p(\mathbf{y}|\mathbf{X})$ is a normalization constant. Given a new input vector $\mathbf{x}^*$, we can make a prediction for its output y^* using the predictive distribution given by

$$p(y^*|\mathbf{x}^*, \mathcal{D}) = \int p(y^*|\mathbf{x}^*, \mathbf{W}, \gamma)p(\mathbf{W}, \gamma, \lambda|\mathcal{D})d\gamma d\lambda d\mathbf{W}, \quad (4)$$

where $p(y^*|\mathbf{x}^*, \mathbf{W}, \gamma) = \mathcal{N}(y^*|f(\mathbf{x}^*, \mathbf{W}), \gamma^{-1})$. It can be seen that the Bayesian prediction explicitly represent and propagate the data and model uncertainties into the prediction uncertainty. Moreover, Eq. 4 indicates that the Bayesian prediction is a weighted sum of what the model would predict under any possible configuration of the parameters, weighted by the parameter posterior $p(\mathbf{W}, \gamma, \lambda|\mathcal{D})$. By accounting for all possible explanations of the data when making predictions, BNN avoids overfitting that occurs when the model learns a specific configuration of parameters given training data and does not generalize well to unseen data. However, the exact computation of $p(\mathbf{W}, \gamma, \lambda|\mathcal{D})$ and $p(y^*|\mathbf{x}^*, \mathcal{D})$ is intractable in most geoscientific problems and some approximation approaches need to be used practically. In this work, we adopt a recently proposed nonparametric variational inference method called Stein variational gradient descent (SVGD) [8] to approximate the Bayesian inference in DNN models.

B. Stein variational gradient descent (SVGD) algorithm

The SVGD algorithm leverages the optimization property of the VI approaches and the assumption-free property of the sampling techniques, so it is efficient, general-purpose, and widely applicable for high-dimensional Bayesian inference tasks.

In a general definition, we are interested in Bayesian inference of the uncertain parameters $\boldsymbol{\theta}$, i.e., to calculate its posterior distribution $p(\boldsymbol{\theta}|\mathcal{D})$ dependent on the pairs of observation (training) data $\mathcal{D} = \{\mathbf{x}_n, y_n\}_{n=1}^{N}$. Hereinafter we drop the conditioning on $\mathcal{D}$ and represent the posterior $p(\boldsymbol{\theta}|\mathcal{D})$ as $p(\boldsymbol{\theta})$ for convenience. VI approximates the target posterior distribution $p(\boldsymbol{\theta})$ with a simple and tractable variational distribution $q(\boldsymbol{\theta})$ from a restricted family by minimizing the two distributions' KL divergence. SVGD is a nonparametric VI algorithm; it defines the variational distribution from a set of particles instead of using a parametric form from a specific family. Thus, SVGD addresses the challenges of the traditional VI methods that simple approximation sets are too restrictive to resemble the target posterior and more advanced choices cast more difficulties on the subsequent optimization tasks, making SVGD generally applicable to a wide range of problems.

The key idea of SVGD is to start from a set of particles $\{\boldsymbol{\theta}_i^0\}_{i=1}^{S}$ generated from an initial tractable distribution (e.g., the prior distribution) and then use a set of sequential deterministic transforms to iteratively push these particles closely to simulate the shape of the target posterior distribution:

$$\boldsymbol{\theta}_i^{t+1} \leftarrow \mathcal{T}_{t+1}(\boldsymbol{\theta}_i^t), \text{where } \mathcal{T}_{t+1}(\boldsymbol{\theta}_i^t) = \boldsymbol{\theta}_i^t + \delta_t \phi(\boldsymbol{\theta}_i^t). \quad (5)$$

The transform $\mathcal{T}_{t+1}$ at iteration $t+1$ is constructed based on the most recent particles $\{\boldsymbol{\theta}_i^t\}_{i=1}^{S}$ at iteration t being perturbed by a velocity field ϕ with a magnitude controlled by a step size δ_t whose value is assumed to be small. The velocity field ϕ is chosen to maximally decrease the KL divergence between the distribution of the particles and the target distribution.

The SVGD algorithm is summarized in Algorithm 1 where the velocity field ϕ is derived based on Stein's identity and kernelized Stein discrepancy. More details refers to [8].

Algorithm 1: The SVGD algorithm

input : A target distribution $p(\boldsymbol{\theta})$; A set of initial particles $\{\boldsymbol{\theta}_i^0\}_{i=1}^{S}$; a kernel function k, and a step-size scheme δ_t

output: A set of particles $\{\boldsymbol{\theta}_i\}_{i=1}^{S}$ that approximates the target distribution.

for iteration t **do**

$$\boldsymbol{\theta}_i^{t+1} \leftarrow \boldsymbol{\theta}_i^t + \delta_t \phi(\boldsymbol{\theta}_i^t) \text{ where}$$

$$\phi(\boldsymbol{\theta}_i^t) = \frac{1}{S}\sum_{j=1}^{S}\big[k(\boldsymbol{\theta}_j^t, \boldsymbol{\theta}_i^t)\nabla_{\boldsymbol{\theta}_j^t}\log p(\boldsymbol{\theta}_j^t) + \nabla_{\boldsymbol{\theta}_j^t} k(\boldsymbol{\theta}_j^t, \boldsymbol{\theta}_i^t) \quad (6)$$

end

Algorithm 1 simulates gradient dynamics at the particle level, starting with a set of arbitrary particles, the velocity field ϕ transports these particles to match the target distribution $p(\boldsymbol{\theta})$ with user-defined step size δ_t. The two terms of ϕ in Eq. 6 play critical roles in the transportation. The first term drives the particles towards the high probability regions

of $p(\boldsymbol{\theta})$ by following a gradient ascent direction, which is calculated as the weighted sum of the gradients $\nabla_{\boldsymbol{\theta}_j} \log p(\boldsymbol{\theta}_j)$ of all the particles $\{j = 1, \cdots, S\}$ weighted by the kernel function $k(\boldsymbol{\theta}_j, \boldsymbol{\theta}_i)$. The second term acts as a repulsive force that prevents all the particles falling together into local modes of $p(\boldsymbol{\theta})$. Here we consider the radial basis kernel function $k(\boldsymbol{\theta}_j, \boldsymbol{\theta}_i) = \exp\big(-\frac{1}{h} \parallel \boldsymbol{\theta}_j - \boldsymbol{\theta}_i \parallel^2 \big)$. After calculating the derivation of the kernel, the second term in Eq. 6 reduces to $\sum_j \frac{2}{h}(\boldsymbol{\theta}_i - \boldsymbol{\theta}_j) k(\boldsymbol{\theta}_j, \boldsymbol{\theta}_i)$, which drives $\boldsymbol{\theta}_i$ away from its neighboring points $\boldsymbol{\theta}_j$ that have large $\nabla_{\boldsymbol{\theta}_j} k(\boldsymbol{\theta}_j, \boldsymbol{\theta}_i)$. When the bandwidth $h \to 0$, the second term of ϕ vanishes and the Eq. 6 reduces to a set of independent chains of typical gradient ascent for maximizing $\log p(\boldsymbol{\theta})$ (i.e., maximum a posteriori, MAP) and all the particles would fall into the local modes. When only one particle ($S = 1$) is used, Algorithm 1 reduces to a single chain of the gradient ascent for MAP for any defined kernels that satisfy $\nabla_{\boldsymbol{\theta}} k(\boldsymbol{\theta}, \boldsymbol{\theta}) = 0$. This suggests that SVGD can perform well even with a very small number of particles, since the gradient ascent for MAP has been shown successful in practice. This property of SVGD makes it more efficient than sampling methods that typically require many points for convergence.

The major computational complexity in evaluating Eq. 6 lies on the calculation of the gradient $\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{\theta})$ for all the particles. For a large dataset $\{\mathcal{D}_n\}_{n=1}^N$, the likelihood evaluation involves a multiplication of N Gaussian functions as shown in Eq. 1. The other computational cost in assessing Eq. 6 comes form the calculation of the kernel matrix $k(\boldsymbol{\theta}_j, \boldsymbol{\theta}_i)$ whose complexity is $\mathcal{O}(S^2)$, i.e., the larger the number of particles S is, the higher the computing cost. In section III, we investigate the influence of particle size on the BNN's prediction performance and find that the prediction accuracy is rather insensitive to the number of particles in the problem, and in this case a relatively small number of particles can be used to reduce the computational cost without sacrificing the prediction accuracy.

In application of the SVGD algorithm to the BNNs defined in section II-A, we specify the parameter $\boldsymbol{\theta} = \{\mathbf{W}, \gamma, \lambda\}$. And based on Eq. 3, we can write $\log p(\boldsymbol{\theta}) = \log p\big(\mathbf{y}|\mathbf{W}, \mathbf{X}, \gamma\big) + \log p(\mathbf{W}|\lambda) + \log p(\lambda) + \log p(\gamma)$ without consideration of the normalization constant. We use the radial basis kernel function with the bandwidth $h = \text{median}^2/\log S$, where the median is defined as the median of the pairwise distance between the current set of particles. According to [8], this setup of h balances the contribution of each particle from its own gradient and the influence from the other particles. We initialize the particles using the prior distribution and use Adam algorithm [9] for optimization with step size $\delta = 0.01$. In learning, we investigate the predictive performance of the training data and testing data, and terminate the training process when the performance has minor change or when the number of iterations hits a given value.

C. Prediction and predictive uncertainty of the BNN

BNNs predict the quantity y^* at a new input $\mathbf{x}^*$ using the predictive distribution $p(y^*|\mathbf{x}^*, \mathcal{D})$ which is a high-dimensional integral defined in Eq. 4. We can use sampling methods to approximate this intractable integral based on the samples of the NN parameters $\mathbf{W}$ generated in above section II-B. In particular, we use the samples to approximate the first and second statistical moments $\mathbb{E}\big[p(y^*|\mathbf{x}^*, \mathcal{D})$ and $\text{Var}\big[p(y^*|\mathbf{x}^*, \mathcal{D})$ of the predictive distribution. That is,

$$\mathbb{E}\big[p(y^*|\mathbf{x}^*, \mathcal{D}) \approx \frac{1}{S} \sum_{i=1}^{S} f(\mathbf{x}^*, \mathbf{W}^i), \tag{7}$$

$$\text{Var}\big[p(y^*|\mathbf{x}^*, \mathcal{D}) \approx \frac{1}{S} \sum_{i=1}^{S} \Big((\gamma^i)^{-1} \mathbf{1} + f^2(\mathbf{x}^*, \mathbf{W}^i) \Big) - \frac{1}{S} \sum_{i=1}^{S} f(\mathbf{x}^*, \mathbf{W}^i) \Big)^2, \tag{8}$$

where $\{(\gamma^i)^{-1}\}_{i=1}^S$ and $\{\mathbf{W}^i\}_{i=1}^S$ are samples from the SVGD calculation.

The term γ^{-1} in Eq. 8 measures aleatoric uncertainty caused by the data noise. Here we consider homoscedastic aleatoric uncertainty by assuming identical data noise for every input sample x. The remaining terms in Eq. 8 measures the epistemic uncertainty which captures our ignorance about the models most suitable to explain the existing limited data. The epistemic uncertainty represents the model uncertainty and it can be reduced with the increase of training data and a better choice of NN model. These data and model uncertainty are propagated to the prediction uncertainty. Eq. 8 suggests that BNN explicitly quantifies the prediction uncertainty and considers the influence of uncertain data on the predictions. Moreover, Eq. 7 indicates that BNN's prediction is an average across many models during training. This gives a regularization effect to the network thus avoiding overfitting. Hence, BNNs are particularly suitable for geoscientific problems.

III. Application

In this study, we use the BNN to build a fast-to-evaluate surrogate model of the complex and computationally expensive Energy Exascale Earth System Land Model (ELM) for efficient modeling. We assess the surrogate accuracy by comparing the BNN prediction with the ELM simulation for some testing data. Additionally, we analyze the BNN's prediction performance and computational efficiency and robustness in comparison with the deterministic NN.

A. Description of the terrestrial ecosystem model

The ELM is a complex land-surface model that simulates terrestrial water, energy, and biogeochemical processes and is an important tool for improving our understanding of ecosystem responses to climate change. The ELM simulates several carbon states and flux variables, and here we are interested in simulating annual sum of net ecosystem exchange of carbon dioxide (NEE), annual average total leaf area index (TLAI), and latent heat fluxes (LH) at the Missouri Ozark flux tower (MOFLUX) site from 2006 to 2010. Therefore in total, there are 15 model output variables of interest. A Python scripting framework is used to perform site-level ensemble simulations

TABLE I
MODEL PARAMETERS AND THEIR RANGES TO GENERATE TRAINING AND TESTING DATA.

Parameter	Minimum	Maximum	Units
$rootb_par$	0.5	4.0	$none$
$slatop$	0.01	0.05	m^2/gC
$flnr$	0.1	0.4	$none$
$frootcn$	25	60	gC/gN
$froot_leaf$	0.3	1.5	$none$
br_mr	1.5×10^{-6}	4.0×10^{-6}	$gC/m^2/s/gN$
$crit_dayl$	35000	45000	$seconds$
$crit_onset_gdd$	600	1000	m^2/gC

for all given sets of parameter values. Because accurate carbon cycle predictions require multi-century spin-up and transient simulations in addition to simulating the five years of interest, one forward model simulation takes about 12 hours on a single processor. This computational complexity limits the application of advanced model-data integration analysis such as the iterative global parameter optimization to improve the model performance. To reduce the computational cost, we use NNs to build surrogate models of the ELM based on pairs of ELM input and output samples and then use the cheap-to-evaluate NN model to improve the computational efficiency in the model-data integration tasks. The ELM has a large number of parameters. Recently, 65 ELM parameters were analyzed in a global sensitivity analysis for 96 sites [10], and the study indicated that at the MOFLUX site, eight model parameters were responsible for more than 80% of the variation in five quantities of interest including the NEE, TLAI and LH. These eight parameters are listed in Table I with their physically reasonable ranges. In this defined parameter space, we randomly draw 1000 samples and run the ELM to get corresponding samples of the 15 output variables. Based on the 1000 pairs of model input-output samples, we use NNs to construct the surrogate model and evaluate its performance.

B. BNN produces accurate predictions with UQ

In this section, we discuss the NN prediction performance for a fixed network structure, i.e., a single hidden layer with 50 nodes. To make a fair comparison between the deterministic NN and BNN, all the settings in the NN and BNN trainings are the same and the same random seed is used to avoid the impact of randomness. To investigate the influence of training data size on the NN's performance, we consider a collection of training sets that are a subset of the 1000 samples, i.e., $[10, 30, 50, 100, 300, 500, 700, 900]$, and always reserve the last 100 of these 1000 samples as the testing data. We use the coefficient of determination, R^2 score, to quantify the NN's performance in building a surrogate model of the ELM. The R^2 score measures how well the actual ELM simulation can be replicated by the NN-based surrogates. The closer the R^2 value is to 1.0, the more accurate the surrogate estimation. The score value can be negative which occurs when the surrogate simulation is even worse than the estimation based on the training data mean. So, a negative R^2 score indicates an unreasonable surrogate prediction.

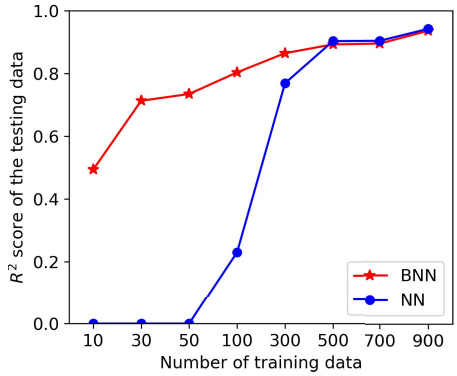

Fig. 1. R^2 scores of the testing data based on the BNN and the deterministic NN for different training sizes.

Fig. 1 compares the overall performance of the BNN and the deterministic NN in prediction of the 15 quantities at all the testing data. The figure indicates that BNN performs much better than the NN especially for a small training data. For example, when the training size is smaller than 50, the R^2 of the deterministic NN has negative values (Note: in Fig. 1 we display the negative values as zeros for the purpose of visualization), suggesting that the NN is incapable of a reasonable and realistic prediction based on a small training size. In contrast, the BNN, even with 10 training data, can produce a fairly accurate prediction having the R^2 score of 0.5. And as the training size increases, the prediction accuracy of the BNN gradually improves and has a R^2 score about 0.9 for 500 training data. When the number of training data exceeds 500, the deterministic NN and the BNN have the similar performance.

Fig. 2 evaluates the prediction errors of the BNN and the deterministic NN in predicting the five years annual NEE, TLAI, and LH at the 100 testing data. The prediction error is calculated as the absolute difference between the network model and the ELM simulations averaged over all the testing data, and the error values in each heat map are standardized because of the different magnitudes of the three output quantities. The deterministic NN uses the same standardization of the BNN for a fair comparison. The figure indicates that for all the three quantities, the prediction errors of the BNN are significantly smaller than those of the deterministic NN, especially for a small training set. Also, BNN performs equally well for the three kinds of quantities, while the deterministic NN performs particularly poor in predicting the LH suggesting an unstable training. The equally-good performance among different variables is important when we are interested in simulating a large number of quantities with various types from different locations at different times.

Fig. 3 analyzes the prediction performance of BNN and the deterministic NN for predicting NEE, TLAI, and LH of year 2010 at one testing sample in detail. ELM simulations of the three quantities are listed as the reference in red lines. Once again, the figure indicates that BNN gives a more accurate

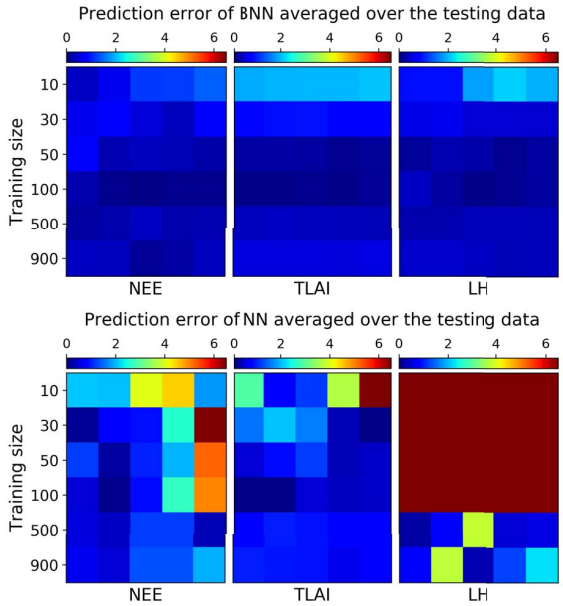

Fig. 2. Prediction error of the BNN and the deterministic NN in predicting NEE, TLAI, and LH at all testing points. The errors are standardized and the deterministic NN uses the same standardization of the BNN.

prediction than the deterministic NN. More importantly, compared to the point estimation of the deterministic NN, BNN quantifies prediction uncertainty and the uncertainty bound (calculated as two standard deviations according to Eq. 8) encloses the reference for all the training cases. Additionally, with the increase of training size, BNN produces a more accurate and precise prediction with the predicted values getting closer to the reference and the predictive uncertainty decreasing while still enclosing the reference.

One reason for the superior prediction performance of BNN to the deterministic NN is due to the fact that BNN incorporates naturally effects of regularization and thus prevents overfitting. This makes the well-trained BNN model also predict the unseen testing data pretty well. As shown in Fig. 4, while the deterministic NN without applying any regularization techniques overfits greatly on the training data and produces a poor prediction in the testing data, BNN obtains the similar superior prediction performance on both training and testing data. Moreover, with a small number of epochs, BNN starts producing fairly good predictions. The shorter the training process is, the lower the training costs. For example, at the beginning of the training with a few epochs, BNN already shows high R^2 scores on both training and testing curves, and the R^2 values quickly climb up above 0.8 after 400 epochs. Whereas, the deterministic NN has negative R^2 scores before 200 epochs of the training data and before 300 epochs of the testing data. Till 2000 epochs, the NN starts having similar training performance with the BNN, but the prediction accuracy of the NN in the testing data is still significantly low with the R^2 score below 0.5.

In this section, we demonstrate that BNN, with a small number of training data, can produce an accurate prediction with uncertainty information. Additionally, BNN naturally prevents overfitting and makes consistent and stable estimations. These nice properties of BNN enables it to solve various difficult Earth science problems such as the efficient and effective surrogate modeling of the complex Earth system models in this work.

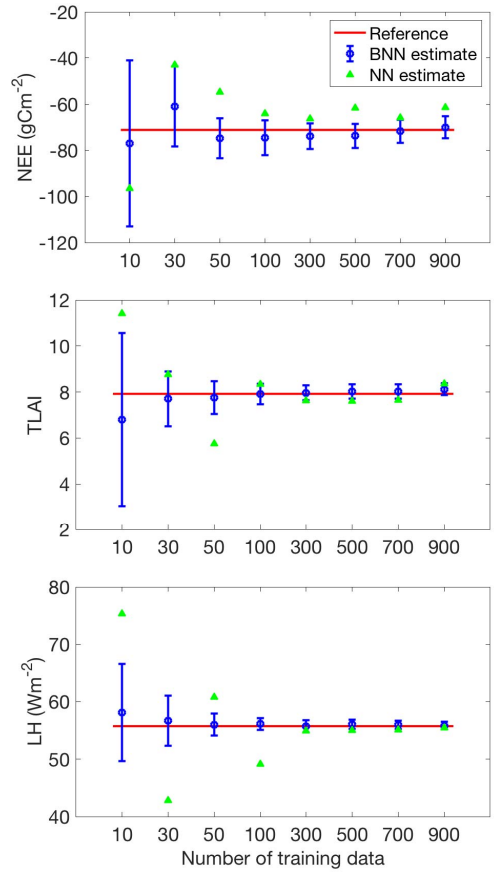

Fig. 3. Prediction performance of the BNN and the deterministic NN in predicting NEE, TLAI, and LH at one testing sample.

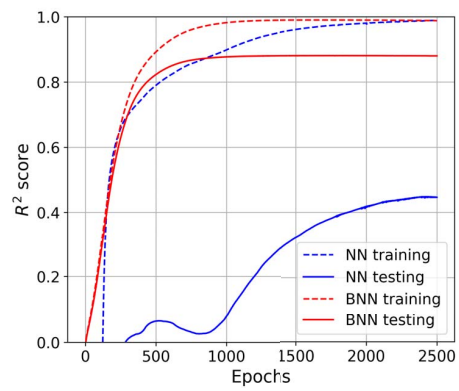

Fig. 4. R^2 scores in predicting the training and testing data based on the deterministic NN and the BNN.

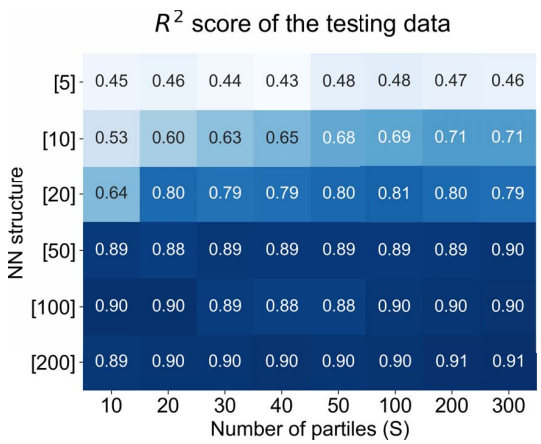

Fig. 5. R^2 score of the testing data produced by the BNNs with different NN structures and particle sizes.

Fig. 6. Computing time used in training the BNNs with different NN structures and particle sizes running on a single CPU processor.

C. BNN is computationally efficient and effective

In this section, we discuss the computational efficiency and robustness of BNN by investigating its prediction performance for a series of network structures, all having a single hidden layer with nodes of $5, 10, 20, 50, 100,$ and 200, respectively; and for a sequence of particle sizes where $S \in [10, 20, 30, 40, 50, 100, 200, 300]$. The BNN considered here is trained using 500 samples and used to predict the 100 samples.

Fig. 5 lists the predictive R^2 scores of BNN for different network structures and particle sizes. The figure indicates that the predictive performance of BNN is relatively sensitive to the NN structure but insensitive to the particle size. After finding a suitable network structure, here the number of nodes greater than 50, the BNN using 10 particles obtains the similar performance with the application of 300 particles. More particles mean higher computational costs. As shown in Fig. 6, the time of training the same BNN structure (for example, with 50 nodes) using 300 particles is about 30 times higher than the training time using 10 particles, but the prediction accuracy of the two trainings is about the same. This suggests that after finding a suitable network structure, the SVGD-based BNN can use a small number of particles for training to save the computational cost. Note that, searching a proper NN structure is not an issue specifically to BNN, in fact, it affects all the NN's performance. Thus, before using BNN for learning and inference, it is better to perform model selection analysis and network hyperparameter optimization to produce a good NN to work on.

BNN explicitly quantifies the data and model uncertainties and propagates these uncertainties to predictions to provide the knowledge about how confident we are in the NN model estimation. Thus, if the Bayesian inference correctly quantifies the model uncertainty, it is expected to see a decrease in the prediction uncertainty of BNN for a proper model that produces an increasing prediction accuracy. This is reasonable that the more correct labels the model predicts, the more certain we are about the suitability of the model for the problem. Fig. 7 shows the predictive standard deviations produced by BNN for different NN structures and particle sizes. A clear trend along the columns in the heat map can be observed that with an increasing complexity of NN structures, the prediction uncertainty decreases with decreasing standard deviation values. This trend is opposite to Fig. 5 where the prediction accuracy increases with an increasing NN structure complexity. The correlating pattern between Fig. 5 and Fig. 7 indicates that our BNN is able to correctly quantify the model uncertainty by demonstrating the negative correlation between prediction accuracy and model uncertainty, which is observable for all the particle cases. Additionally, the figures suggest that a good choice of NN model can improve the prediction accuracy and reduce the model uncertainty. Thus, in practice when the predictive uncertainty is large and more training data is not available, we may consider finding a more suitable ML model to explain the existing limited data so as to improve the model prediction performance.

In this section, we demonstrate that BNN can efficiently produce accurate and credible predictions using low computational costs. On the other hand, although it seems that BNN requires more computational time than the deterministic NN for training, it is not easy and fair to draw the conclusion that BNN is more computationally expensive. First, BNN may need a few iterations to obtain a high prediction accuracy while the deterministic NN, after a very long iteration, may still not get a reasonable prediction (Fig. 4). Second, even BNN uses 10 particles to train a fairly complex network on a single CPU, the training time is less than 2 minutes, and we can always use parallelization and GPUs to speed up the BNN training. More importantly, comparing to the benefits of using BNN such as the predictive uncertainty quantification and the required small number of labelled data for training, the affordable computational time in training BNN is a minor issue.

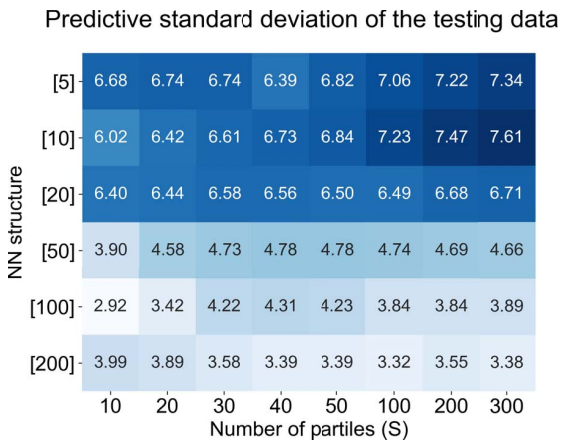

Fig. 7. Predictive standard deviation of the testing data produced by the BNNs with different NN structures and particle sizes.

IV. Discussion and Conclusion

Deep learning has a great potential application in Earth system science. For example, CNN enables automated characterization of objects and detection of events; FNN infers critical Earth system variables that are difficult to monitor directly; RNN is capable of predicting long-term trends of the states of the Earth system; and we can also use network models to understand how different physical processes are related to each other and discover cause-effect relationships in the Earth system. However, Earth system objects and variables are complex, multivariate, highly nonlinear, and non-stationary; Earth system science data are noisy, incomplete, sparsely labelled, and filled with uncertainty; and meanwhile Earth system model predictions require not only accuracy but also quantification of confidence and credibility. These unique properties of geoscientific problems challenge the traditional DL methods that usually require a large number of labelled data for training, whose performance are sensitive to the uncertain data, and which are unable to quantify the uncertainty.

In this study, we integrate Bayesian inference into NN model training and prediction. The formulated BNN explicitly represents the data and model uncertainties and propagates these uncertainties to the prediction uncertainty in a formal but flexible way. Additionally, the BNN naturally includes a regularization effect thus preventing overfitting. These two nice properties of BNN makes it particularly suitable for Earth science problems. To solve the computational complexity and rigidness of the traditional Bayesian methods, we adopt the efficient and general-purpose Stein variational gradient descent (SVGD) algorithm for Bayesian inference. In application of BNN for a surrogate modeling in the Energy Exascale Earth System Land Model (ELM), the BNN only requires a few dozen training data (i.e., expensive ELM simulations) to make an accurate prediction with high credibility, whereas the traditional deterministic NN requires hundreds of training data to achieve a similar estimation but without any confidence information. BNN quantifies both data and model uncertainties. With an increase of training data, the model uncertainty (thus the total prediction uncertainty) decreases. Moreover, with a selection of a suitable NN model structure, the prediction accuracy increases, and the predictive uncertainty, specifically the model uncertainty, decreases. Thus, practically when the predictive uncertainty is large and more training data are not available due to resource constraints, we may consider finding a more suitable ML model to reduce the model uncertainty thus improve model prediction accuracy and credibility.

The SVGD algorithm is general-purpose and efficiently scalable for high-dimensional Bayesian learning, so it can be easily adopted by geoscientists for advancing the ML/DL application in Earth sciences for more complex problems than the simple demonstration in this work. On the other hand, this study's successful application of BNN in surrogate modeling of the ELM promises the potential of applying the method in building surrogates of other Earth system model components or even the fully integrated Earth system model. Furthermore, the SVGD algorithm can be integrated into different variants of network models such as the FNNs, CNNs, and RNNs for diverse geoscientific applications. For example, we may design Bayesian CNNs to improve the automated characterization of objects and detection of extreme events, and design Bayesian RNNs to train on historical data and then to predict the future system behavior with uncertainty.

BNN accounting for data and model uncertainty along with integration of knowledge by priors and constraints is critical for solving Earth science problems. However, traditionally BNN priors are defined in parameter space, making it hard to encode physical prior knowledge and constraints expressed in function space. In the future, we will formulate a prior that incorporates functional constraints about what the model outputs can or cannot be in regions of the input space to further improve the BNN performance and advance its application in Earth sciences.

Acknowledgment

Primary support for this work was provided by the Scientific Discovery through Advanced Computing (SciDAC) program, funded by the U.S. Department of Energy (DOE), Office of Advanced Scientific Computing Research (ASCR) and Office of Biological and Environmental Research (BER). Additional support was provided by BER's Terrestrial Ecosystem Science Scientific Focus Area (TES-SFA) project and Oak Ridge National Laboratory (ORNL) AI initiative project. The authors are supported by ORNL, which is supported by the DOE under contract DE-AC05-00OR22725.

References

[1] D. Lu, and D. Ricciuto, "Efficient surrogate modeling methods for large-scale Earth system models based on machine-learning techniques", Geosci. Model Dev., 12, 1791-1807, 2019.

[2] Y. Liu, E. Racah, J. Correa, A. Khosrowshahi, D. Lavers, K. Kunkel, M. Wehner, and W. Collins, "Application of deep convolutional neural networks for detecting extreme weather in climate datasets," Int'l Conf. on Advances in Big Data Analytics, 2016.

[3] S. Besnard et al., "Memory effects of climate and vegetation affecting net ecosystem CO2 fluxes in global forests", PLoS ONE 14(2): e0211510, 2019.

[4] F. Kratzert, D. Klotz, C. Brenner, K. Schulz, and M. Herrnegger, "Rainfall–runoff modelling using Long Short-Term Memory (LSTM) networks," Hydrol. Earth Syst. Sci., 22, 6005-6022, 2018.
[5] M. Reichstein, G. Camps-Valls, B. Stevens, M. Jung, J. Denzler, N. Carvalhais, and Prabhat, "Deep learning and process understanding for data-driven Earth system science", Nature, 566, 195-204, 2019.
[6] A. Khosravi, S. Nahavandi, D. Creighton, and A. F. Atiya, "Comprehensive review of neural network-based prediction intervals and new advances", IEEE transactions on neural networks, 22(9), 2011.
[7] Y. Gal, and Z. Ghahramani, "Dropout as a Bayesian approximation: representing model uncertainty in deep learning", Proceedings of the 33 rd International Conference on Machine Learning, New York, NY, USA, 2016.
[8] Q. Liu, and D. Wang, "Stein variational gradient descent: a general purpose bayesian inference algorithm", In Advances In Neural Information Processing Systems, 2370-2378, 2016.
[9] D. P. Kingma, and J. Ba, "Adam: A method for stochastic optimization", arXiv preprint arXiv:1412.6980, 2014.
[10] D. Ricciuto, K. Sargsyan, and P. Thornton, "The impact of parametric uncertainties on biogeochemistry in the E3SM land model". Journal of Advances in Modeling Earth Systems, 10, 297-319, 2018.