A methodological framework for improving the performance of data-driven models, a case study for daily runoff prediction in the Maumee domain, U.S.

Yao Hu[1, 2, *], Chirantan Ghosh[3, *], and Siamak Malakpour-Estalaki[1]

[1]Department of Geography and Spatial Sciences, University of Delaware, Newark, DE, USA
[2]Department of Civil and Environmental Engineering, University of Delaware, Newark, DE, USA
[3]Department of Computer and Information Sciences, University of Delaware, Newark, DE, USA
[*]These authors contributed equally to this work.

Correspondence: Yao Hu (yaohu@udel.edu)

Abstract. Geoscientific models are simplified representations of complex earth and environmental systems (EESs). Compared with physics-based numerical models, data-driven modeling has gained popularity due mainly to data proliferation in EESs and the ability to perform prediction without requiring explicit mathematical representation of complex biophysical processes. However, because of the black-box nature of data-driven models, their performance cannot be guaranteed. To address this issue, we developed a generalizable framework for improving the efficiency and effectiveness of model training and the reduction of model overfitting. This framework consists of two parts: hyperparameter selection based on Sobol global sensitivity analysis, and hyperparameter tuning using a Bayesian optimization approach. We demonstrated the framework efficacy through a case study of daily edge-of-field (EOF) runoff predictions by a tree-based data-driven model using the eXtreme Gradient Boosting (XGBoost) algorithm in the Maumee domain, U.S. This framework contributes towards improving the performance of a variety of data-driven models and can thus help promote their applications in EESs.

1 Introduction

Geoscientific models are simplified representations of complex earth and environmental systems (EESs), where predictive models can have a wide range of applications. For example, they can incorporate and advance the scientific knowledge of EESs, and assess how EESs react to changing conditions (Fleming et al., 2021; Reichstein et al., 2019). Furthermore, evidence-based decisions and policies on EESs can be made by effectively evaluating their influences using these models (Fleming et al., 2021; Prinn, 2013), which would otherwise be impossible, or too costly and time-consuming to implement in practice (Hu et al., 2015; Sohl and Claggett, 2013).

Two broad classes of models are often used to predict target environmental phenomena in EESs, physics-based numerical models, and data-driven machine learning models. Conventionally, the modeling of EESs relies heavily on physics-based models, developed based on the first principles of physics (Bergen et al., 2019), which require comprehensive understanding of the target EES and proper mathematical representations of all processes relevant to the target phenomena. As such, the long development time, insufficient representations of system components, and difficulties in access and use set barriers for the wide

application of such models. On the contrary, despite the fact that data-driven models are data-intensive and not generalizable (Ford et al., 2022), they do not require an explicit mathematical formulation of all underlying complex processes to perform predictive analysis. Thus, the development of data-driven models is often less involved. Moreover, the proliferation of data further leads to the rise of data-driven modeling in EESs (Willard et al., 2020).

For data-driven modeling, model performance relies heavily on the capability of the underlying machine learning (ML) algorithms to retrieve information from data; this capability is controlled by the complexity of ML algorithms and their associated parameters, that is, hyperparameters (Yang and Shami, 2020; Hutter et al., 2015). When the underlying ML algorithm is too simple to learn complex patterns from data, we see large biases in the training phase (i.e., underfitting; Jabbar and Khan, 2015; Koehrsen, 2018). In contrast, model overfitting occurs when the ML algorithm is overcomplicated to capture all random noises in the training data; the resulting model performs very well in training but poorly on the test (i.e., variance error). As such, to improve model performance, we need to determine appropriate ML algorithms that can balance model bias and variance error (Koehrsen, 2018).

There are various rules of thumb to choose an appropriate ML algorithm for data-driven modeling. When the model is underfitting, we can choose more complex ML algorithms (e.g., from linear regression models to tree-based regression models). However, in practice it can be more challenging to reduce overfitting; model overfitting is often associated with a long training time and poor performance in test sets. Because of the black-box nature of data-driven models, only a handful of approaches are available to deal with overfitting. One such approach is random sampling with or without replacement (Gimenez-Nadal et al., 2019) in which data points are randomly selected for training and test. This approach attempts to ensure that the data samples are uniformly distributed: both the training and test sets have data points to represent the entire domain space. Combined with this approach, other approaches such as early stopping (Yao et al., 2007), cross-validation (Fushiki, 2011), and regularization techniques (Zhu et al., 2018) are used to address overfitting by tuning hyperparameters to balance model performance in training and test sets.

Hyperparameters affect model performance through ML algorithms during model training, although they are external parameters to data-driven models. However, not all hyperparameters have the same level of impact on model performance, as they affect different aspects of ML algorithms to retrieve data patterns. For example, some hyperparameters control the algorithm complexity, while some are used to reduce overfitting as mentioned above. By tuning these hyperparameters, we want to identify optimal hyperparameter values for the ML algorithm. We can then apply the optimized ML algorithm to maximize the model performance during training.

Tuning hyperparameters manually becomes unfeasible as the number of hyperparameters associated with the ML algorithm increases. Hyperparameter optimization algorithms are developed to automatically identify the optimal hyperparameters to maximize model performance by minimizing a predefined objective function (i.e., loss function) of a data-driven model. A variety of optimization approaches are available and categorized based on the mechanisms used to search the optimal hyperparameter values: [..[1]]l) exhaustive search using grid or random search (Liashchynskyi and Liashchynskyi, 2019; Bergstra et al.,

---

[1]removed: 1

2011) and [..[2]]II) surrogate models using sequential model-based optimization (SMBO) methods (Bergstra et al., 2011). The choice of tuning approaches is affected by several factors, such as the number of hyperparameters, different ranges of their values, and the complexity of ML algorithms. In general, compared [..[3]]to the Category I approaches, the [..[4]]Category II approaches are more suitable for data-driven models with complex ML algorithms (Bergstra et al., 2011).

60   Rather than tuning all hyperparameters, it is expected to be more efficient and effective if we only need to tune a subset of them to achieve similar or better model performance. Similarly to assessing the overall impact of model parameters on model prediction for physics-based models, we can use global sensitivity analysis approaches to identify critical hyperparameters for model performance based on sensitivity scores (Sobol, 2001); hyperparameters with high sensitivity scores are considered influential, while the rest with low sensitivity scores are considered to have no or negligible influence on model performance.

65 Additionally, since fewer influential hyperparameters are involved in model training, less training time is required to achieve maximum model performance. Therefore, it is particularly useful if data-driven models need to be trained with streaming data for real-time predictions (Gomes et al., 2017).

    With the proliferation of data in EESs, we expect to have more EES applications using data-driven models. In this study, we present a new framework for data-driven modeling that combines hyperparameter selection and tuning to minimize training

70 time, reduce overfitting, and maximize overall model performance. As such, the fundamental contribution of our work is a framework, which can 1) identify a subset of hyperparameters critical for model performance through hyperparameter selection using a variance-based sensitivity analysis approach, and 2) provide optimal values for the selected hyperparameters through an optimization-based hyperparameter tuning approach. As such, we can improve the overall efficiency and effectiveness of model training, leading to better model performance. In turn, this can further promote the use of data-driven models in EESs.

75 The efficacy of the framework is evaluated using data-driven models developed to predict the magnitudes of daily surface runoff at a farm scale in the Maumee domain, U.S.

2   Method

In this study, we developed a framework to improve the performance of data-driven models by reducing their training time and overfitting; the framework comprises two modules, hyperparameter selection (HS) and hyperparameter tuning (HT; Figure

80 1). To use the framework, we first choose a machine learning algorithm and its associated hyperparameters. Then, we feed the initial hyperparameters (1) to the hyperparameter selection (HS) module to determine the influential hyperparameters (2). Once initial values are assigned to the influential hyperparameters (4), we use the hyperparameter tuning (HT) module to identify their optimal values (3), which allows the algorithm to achieve the optimal performance in training.

    In the following sections, we will discuss the framework in detail, including the use of a global sensitivity analysis ap-

85 proach to select the hyperparameters critical for model performance and an optimization approach for hyperparameter tuning

---

[2]removed: 2

[3]removed: with the category

[4]removed: category
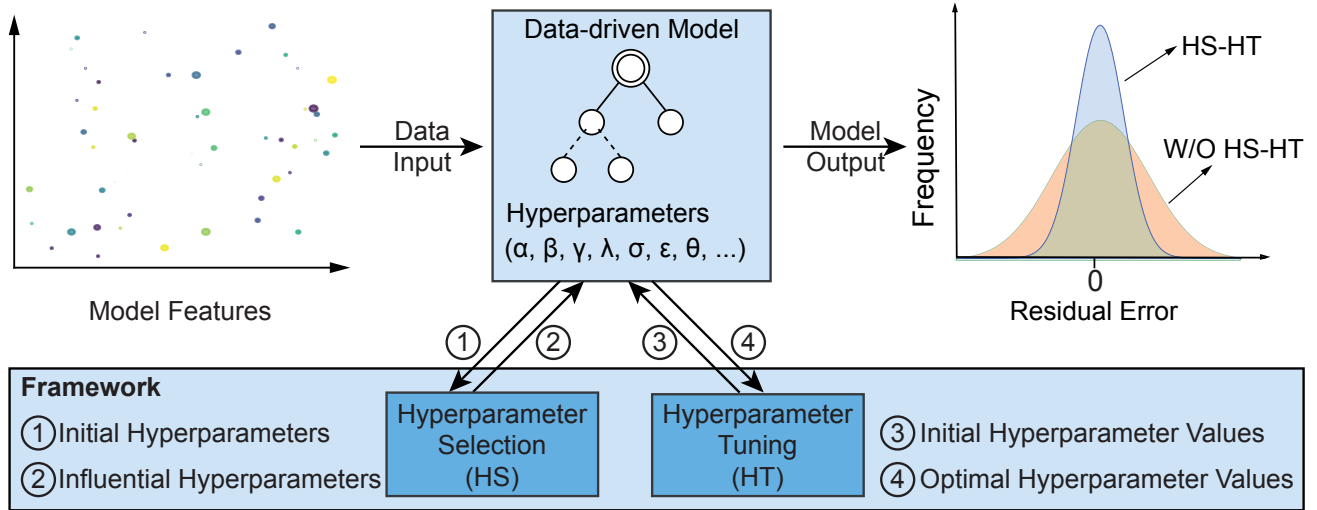
Figure 1. The methodological framework for improving the performance of data-driven models with two modules: hyperparameter selection (HS) and hyperparameter tuning (HT)

to identify the optimum of these critical hyperparameters for model training. A data-driven model using the eXtreme Gradient Boosting (XGBoost) algorithm (Chen and Guestrin, 2016) is selected to demonstrate the efficacy of the proposed framework.

2.1 Hyperparameter Selection

To understand the impact of individual hyperparameters and their interactions on the performance of a given data-driven model, we used a global sensitivity analysis (GSA) approach based on Sobol decomposition (Sobol, 2001), a variance decomposition technique. Through this GSA approach, the model output variance is decomposed into the summation of the variances from input parameters per se and their interactions. Let us assume that a data-driven model is of the form $\mathbf{Y} = \mathcal{M}(\mathbf{X}; \mathbf{H})$, where $\mathbf{X}$ is a vector of features and $\mathbf{H} = \{h_1, \ldots, h_i, \ldots, h_n\}$, a vector of hyperparameters, and h_i is the ith hyperparameter. We [..[5]]used $\mathcal{O}(Y)$ to define the scores of the objective function, $\mathcal{O}(\mathbf{Y})$ of the data-driven model, $\mathcal{M}$. By fixing the values of the features, $\mathbf{X}$ and changing the values of the hyperparameters, $\mathbf{H}$, the total variance of the score, denoted as $V(\mathcal{O}(Y))$ can be represented as the sum of the variance imposed by the individual hyperparameter, h_i and its interactions with the other hyperparameters.

$$V(\mathcal{O}(Y)) = \sum_{i=1}^{n} V_i + \sum_{i \leq j \leq n}^{n} V_{ij} + \ldots + V_{1,\ldots,n}, \tag{1}$$

where V_i is the first order contribution of h_i to $V(\mathcal{O}(Y))$, and V_{ij} denotes the variance arising from the interactions between two hyperparameters, h_i and h_j. We can then measure the influence of a hyperparameter by its contribution to $V(\mathcal{O}(Y))$ using

---

[5]removed: use

the sensitivity score of the first (S) and total order (ST) indices as follows:

$$S_i = V_i/V(\mathcal{O}(Y)) \tag{2}$$

$$ST_i = 1 - V_{\sim i}/V(\mathcal{O}(Y)), \tag{3}$$

where $V_{\sim i}$ indicates the contribution to $V(\mathcal{O}(Y))$ by all hyperparameters except h_i; S_i measures the direct contribution to $V(\mathcal{O}(Y))$ by h_i; ST_i measures the contribution by h_i and its interactions, of any order, with all other hyperparameters.

To estimate S and ST, we first generated sufficient samples of the hyperparameters that can well represent the sample space. We chose the Quasi-Monte Carlo sampling method (Owen, 2020), which uses quasi-random numbers (i.e., low discrepancy sequences) to sample points far away from the existing sample points. As such, the sample points can cover the sample space more evenly and quickly with the faster convergence rate of $O((logN)^k N^{-1})$ where N and k are the number and dimension of samples (Campolongo et al., 2011). In total, we generated m samples for n hyperparameters. We then fed the samples into the data-driven model, $\mathcal{M}$ to obtain the corresponding $\mathcal{O}(Y)$. Next, we estimated the variance components, V_i, $V_{\sim i}$, and $V(\mathcal{O}(Y))$ in Eq. (1) with $m(2n+2)$ model evaluations, and derived the scores of S_i and ST_i using Equations (2) - (3) (Saltelli, 2002). Finally, we selected the hyperparameters with high scores of the total order index as influential hyperparameters.

2.2 Hyperparameter Tuning

After hyperparameter selection, we expect to tune fewer hyperparameters through hyperparameter optimization, which involves the process to maximize or minimize the score of the objective function, $\mathcal{O}(\mathbf{Y})$ of a data-driven model, $\mathcal{M}$ over the sample space of its hyperparameters, $\mathcal{H}$. As such, we can identify the optimal values of the hyperparameters, which are then used for training data-driven models.

Rather than manually tuning these hyperparameters, we chose to use an automated optimization approach, Bayesian hyperparameter optimization (Bergstra et al., 2011). This approach creates a surrogate model of the objective function using a probabilistic model. The surrogate model avoids solely relying upon local gradient and Hessian approximations by tracking the paired values of the hyperparameters and the corresponding scores of the objective function from previous trials and proposes new hyperparameters that can improve the score based on the Bayes rules. [..[6]]As such, this optimization approach can help avoid being trapped in the local optima and also requires far less time to identify the optimum of the hyperparameters, as the objective function can converge to a better score faster.

To describe the Bayesian optimization approach in more detail, let us assume that we have evaluated the objective function, $\mathcal{O}(\mathbf{Y})$ for n sets of hyperparameters, $\{h^{(1)}, \ldots, h^{(n)}\}$. Based on pairs of a set of hyperparameters and the corresponding score $(h, \mathcal{O}(y))$ from n evaluations, we applied a sequential Model-Based Optimization (SMBO) method, a Tree-structured Parzen Estimator (TPE) (Bergstra et al., 2011) to develop a surrogate model for $\mathcal{O}(\mathbf{Y})$. The TPE defines the conditional probability,

---

[6]removed: This automated approach

$p(h|\mathcal{O}(y))$ using two densities:

$$p(h|\mathcal{O}(y)) = \begin{cases} l(h) & \text{if } \mathcal{O}(y) < \mathcal{O}(y^*) \\ g(h) & \text{if } \mathcal{O}(y) \geq \mathcal{O}(y^*), \end{cases} \tag{4}$$

where $l(h)$ and $g(h)$ can be modeled using different probability density or mass functions for continuous or discrete hyperparameters. For example, $l(h)$ and $g(h)$ can be a uniform, a Gaussian, or a log-uniform distribution for continuous hyperparameters. To define $l(h)$, we use part of n sets of hyperparameters, $\{h^{(i)}\}$ that result in the score, $\mathcal{O}(y)$ less than the threshold, $\mathcal{O}(y^*)$ which is chosen to be some quantile γ of all scores, $p(\mathcal{O}(y) < \mathcal{O}(y^*)) = \gamma$; $g(h)$ is defined using the remaining hyperparameters (Bergstra et al., 2011).

The following step is to decide the next hyperparameter values that possibly give a better score, $\mathcal{O}(y)$ given the corresponding uncertainty measured by the surrogate model (Frazier, 2018). To do so, a selection function is defined based on the Expected Improvement (EI):

$$EI_{\mathcal{O}(y^*)}(h) := \int_{-\infty}^{\mathcal{O}(y^*)} (\mathcal{O}(y^*) - \mathcal{O}(y)) p(\mathcal{O}(y)|h) d\mathcal{O}(y) \tag{5}$$

$$= \int_{-\infty}^{\mathcal{O}(y^*)} (\mathcal{O}(y^*) - \mathcal{O}(y)) \frac{p(h|\mathcal{O}(y)) p(\mathcal{O}(y))}{p(h)} d\mathcal{O}(y), \tag{6}$$

where $p(\mathcal{O}(y)|h)$ is parameterized as $p(h|\mathcal{O}(y))p(\mathcal{O}(y))$. It is set to zero when $\mathcal{O}(y) < \mathcal{O}(y*)$ in order to neglect all hyperparameters that yield no improvements in the score. Through maximizing EI, a better set of hyperparameters is identified. Together with the corresponding score, they are used to update the TPE, $l(h)$, and $g(h)$ for the maximization of EI. The iterative process continues until the maximum allowed number of iterations is reached.

2.3 Case Study

The Maumee River Watershed (Figure 2) is the largest watershed in the Great Lakes region, covering more than 17,000 square kilometers in Ohio, Indiana and Michigan (U.S. Geological Survey, 2014). The watershed receives more than 800mm of annual precipitation on average, with most of its rainfall from March to July and snowfall from December to March; The coldest and warmest months are February and July, respectively (NOAA National Centers for Environmental Information, 2012). The soil in the watershed is mainly composed of glacial till, which is a mixture of clay, silt, sand and gravel deposited by glaciers; this type of soil is highly fertile, but prone to erosion if not managed properly (USDA, 2013). Owing to these excellent geophysical and humid continental climate conditions, over 70% of the watershed is dedicated to agriculture, growing row crops such as corn, soybeans, and wheat (Kalcic et al., 2016). As such, fertilizer applied for crop growth in the watershed contributes over 77% of Total Phosphorus (TP) entering the Western Basin of Lake Erie through the Maumee River (Kast et al., 2021; Maccoux et al., 2016).

Agricultural runoff is the main source of non-point source pollution in the Maumee domain. The high nutrient load carried by edge-of-field (EOF) runoff from agricultural fields in the watershed has had detrimental effects on aquatic ecosystems,
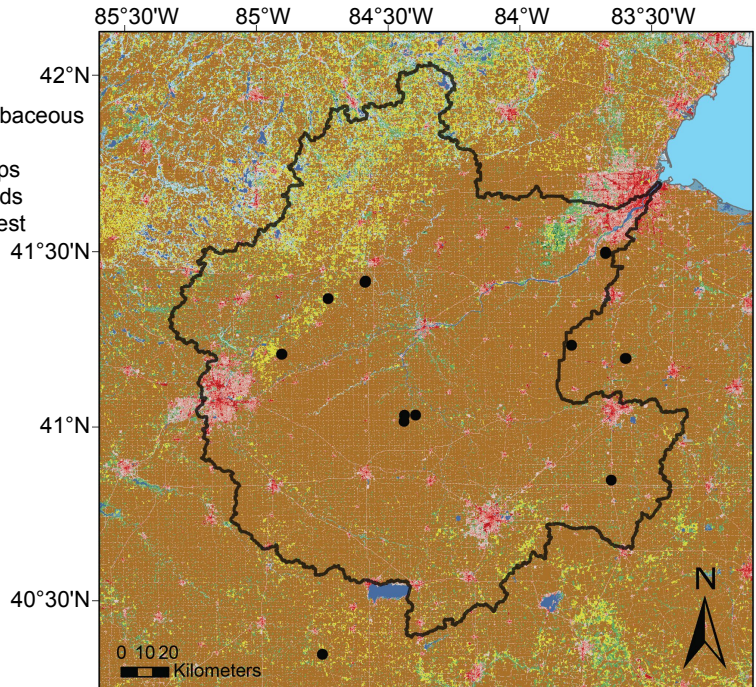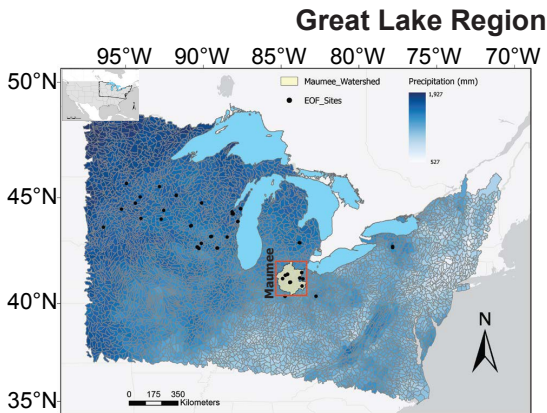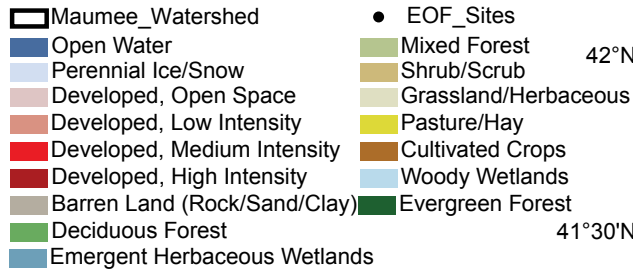
Figure 2. Study area of the Maumee domain in the Great Lakes region, U.S. EOF Sites (black dots) denote the locations where the observational data of daily edge-of-field (EOF) runoff is available over multiple years.

such as harmful algal blooms and hypoxia in Lake Erie (Scavia et al., 2019; Stackpoole et al., 2019). The occurrence and magnitude of EOF runoff can be influenced by many factors, but mainly driven by precipitation and snowmelt (Ford et al., 2022; Hu et al., 2021; Hamlin et al., 2020). An early warning system to forecast runoff risk can assist agriculture producers in the timing of fertilizer application to retain more nutrients in the land; this also reduces nutrient transport carried by runoff to nearby water bodies. To design such an early warning system, in the previous study (Hu et al., 2021), we developed a hybrid model to predict the magnitude of daily EOF runoff for all EOF sites in the domain (Figure 2); the model combines National Oceanic and Atmospheric Administration's (NOAA) National Water Model (NWM) with a data-driven model based on the eXtreme Gradient Boosting algorithm (XGBoost; Chen and Guestrin, 2016). In this study, we will demonstrate the efficiency and effectiveness to train XGBoost models using the proposed framework (Figure 1).

2.3.1 Data preparation

In this study, we used two types of datasets to train XGBoost models preceded by different approaches as illustrated in the framework (Figure 1) and evaluated their performance of daily EOF runoff prediction at each EOF site in the study area. These

two datasets include 1) observations of daily EOF runoff at the EOF sites within the Maumee domain (Figure 2). We obtained this dataset from the conservation partners in the previous study (Hu et al., 2021); 2) daily values of the influential NWM model outputs. Based on the previous work on hybrid modeling using directed information for causal inference (Hu et al., 2021), we first calculated daily values of 72 NWM model outputs on the 1 km x 1 km grids where EOF sites are located and then identified seven influential outputs for the Maumee domain (Tables S1 and S2 in Supporting Information). As we did not separate the winter season (i.e., from November to April) from the rest of the year when selecting influential variables, some of the selected influential outputs represented the driving forces of daily EOF runoff during the winter season, such as snow melt and soil temperature (Table S2 in Supporting Information).

2.3.2 Implementation

eXtreme Gradient Boosting (XGBoost) is a tree-based ensemble machine learning algorithm, which is mainly designed for its overall high convergence speed through optimal use of memory resources, as well as good predictability through ensemble learning that leverages the combined predictive power of multiple tree models (Chen and Guestrin, 2016). Using the gradient boosting technique, XGBoost incorporates a new tree model (i.e., weak learner) into the tree ensemble models obtained from previous iterations in a repetitive manner. At the tth iteration, an objective function, J is defined as:

$$J^{(t)} = \sum_{i=1}^{n} L(y_i, \hat{y_i}) + \sum_{m=1}^{t} R(f_m), \tag{7}$$

where n is the number of samples and L is the training loss function; $\hat{y_i}$ is the prediction from the tree ensemble models $\mathcal{F}$; $\hat{y_i} = \sum_{i=1}^{m} f_m(x_i)$, $f_m \in F$ and m is the number of tree models; R is the regularization function used to penalize the complexity of the tree ensemble models to reduce model overfitting. All these functions are characterized by a set of hyperparameters, e.g., learning rate (LR) and maximum tree depth (MD). Through optimizing J, an XGBoost model can be obtained with locally optimal hyperparameter values, which gives the best predictive performance at the ith iteration. The process iterates for a defined number of repetitions to train the XGBoost model that can balance model bias and variance error.

The XGBoost algorithm has been demonstrated to be effective for a wide range of regression and classification problems, such as overfitting and imbalanced datasets (Dong et al., 2020). In this study, we used XGBoost models to predict the magnitudes of daily EOF runoff in the Maumee domain (Figure 2). We considered nine hyperparameters associated with the XGBoost algorithm (Figure 3). Daily EOF measurements within the watershed (Hu, 2022) were used for hyperparameter selection, and the score $\mathcal{O}(y)$ of the objective function was measured by the mean absolute error (MAE; Section 2.3.3). We then modified the Python SALib package (Herman and Usher, 2017), which was developed for Sobol-based global sensitivity analysis only with model features and parameters; such modification allows the calculation of sensitivity scores for hyperparameters (i.e., the HS approach). Given the number and range of the hyperparameter values, and the complexity of the XGBoost model, we generated 4,000 samples of hyperparameters (Figure 3(a)) to calculate the S and ST values for all nine hyperparameters and selected the influential ones given their ST values.

After the influential hyperparameters were identified, the next step was to search the optimal values for these hyperparameters through hyperparameter tuning (i.e., the HT approach). To do so, we first randomly selected 70% of the EOF data sets within

the domain. Based on the selected data, we then used the Bayesian optimization (BO) approach implemented via the Python Hyperopt library (Bergstra et al., 2013) to identify the optimal hyperparameter values. Given these optimal values, we trained and evaluated XGBoost models in predicting the magnitude of daily EOF runoff at the EOF sites in the domain (Hu et al., 2021). Additionally, to evaluate the impact of hyperparameter selection on model performance, we trained and evaluated the performance of XGBoost models without hyperparameter selection, that is, using only hyperparameter tuning on the initial set of hyperparameters. To further mitigate the impact of the imbalanced runoff data, besides the effective XGBoost algorithm, we used the Stratified K-Fold cross-validation (Kohavi, 1995) across different scenarios to ensure the training and test datasets follow a similar distribution and defined a loss function (e.g., R-Squared) that penalizes more the missing predictions of [..[7]]large runoff events, that is, the minority class in this study.

2.3.3 Evaluation Metrics

In this study, we used mean absolute error (MAE) to measure the score of the objective function, $\mathcal{O}(\mathbf{Y})$, and R-Squared(R^2) to measure the level of agreement between the predictions from the XGBoost models and observations of daily EOF runoff.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \; where \; MAE \in [0, +\infty], \tag{8}$$

$$R^2 = \frac{\left(\sum_{i=1}^{n}\hat{y}_i y_i - n\bar{\hat{Y}}\bar{Y}\right)^2}{\left(\sum_{i=1}^{n}\hat{y_i}^2 - n\bar{\hat{Y}}^2\right)\left(\sum_{i=1}^{n}y_i^2 - n\bar{Y}^2\right)} \; where \; R^2 \in [-\infty, 1], \tag{9}$$

where n is the sample size; $y_i \; and \; \hat{y}_i$ are the observed and predicted value of daily EOF runoff for a specific EOF site, respectively; $\bar{Y} \; and \; \bar{\hat{Y}}$ are the mean value of $y_i \; and \; \hat{y}_i$, respectively. The MAE value equal to zero is the perfect score, whereas the R^2 value closer to one is considered to be the perfect agreement between predictions and observations.

3 Results

The ability to represent the search space of all nine hyperparameters by the selected samples is critical to estimating their influence on the model performance through the sensitivity analysis approach. In our case, we have selected 4,000 samples in total. As shown by the histogram plots on the diagonal in Figure 3(a), the entire range of values for each hyperparameter is well represented by uniformly distributed intervals of values. Additionally, the well-scattered sample points on the off-diagonal plots indicate no correlation among each other, confirming the independence between these hyperparameters and the appropriate samples to use for hyperparameter selection.

Through hyperparameter selection, the influence of hyperparameters is ranked by their contributions to the variance of the objective function, characterized by the sensitivity score of the total order index, ST (Figure 3(b)). The higher the score, the more influential is the hyperparameter. Among the nine hyperparameters for the XGBoost model, the subsample ratio of the training data, SS is the most influential hyperparameter with the highest sensitivity score for both S and ST, followed

---

[7]removed: non-zero
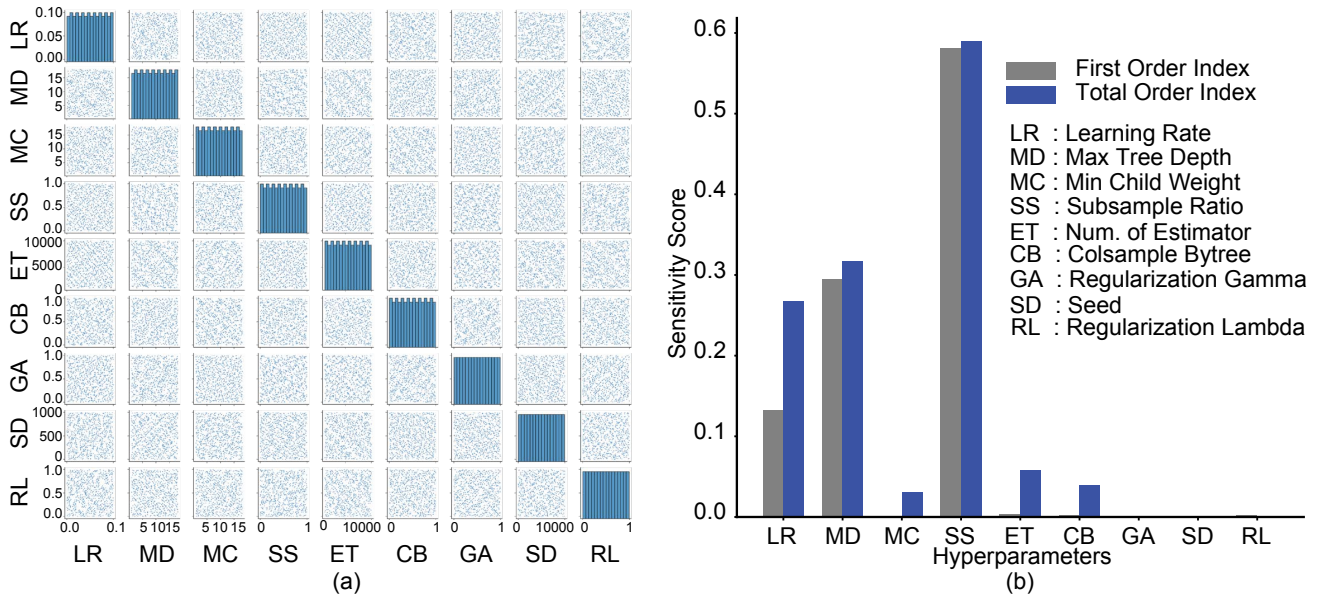
Figure 3. (a) Samples of nine hyperparameters associated with the eXtreme Gradient Boosting (XGBoost) algorithm for the global sensitivity analysis. The histogram plots on the diagonal describe the actual sample distribution for a given hyperparameter. Each bar in the histogram plot measures the sample size in the corresponding interval. (b) Comparison of the sensitivity scores of the nine hyperparameters for both first (grey) and total (blue) order Sobol indices.

by the maximum tree depth, MD, and the learning rate LR (Figure 3(b)). We noticed small differences in sensitivity score between S and ST for the first two most influential hyperparameters, SS and MD, indicating that the contributions to their ST scores are made by the variation of the hyperparameters per se. In contrast, for the hyperparameter, LR, a large portion of its high ST score is contributed through the interaction between LR and the other hyperparameters. As a result, instead of tuning all hyperparameters, we only need to search for the optimal values of these three influential hyperparameters through hyperparameter tuning.

We trained XGBoost models for the prediction of daily EOF runoff events in the study domain (Figure 2) with a fixed number of iterations (i.e., 8,000 in our case). We noticed that the training of XGBoost models preceded by our proposed framework, the combination of hyperparameter selection and tuning (i.e., the HS-HT approach), took the least time (0.7s; Figure 4(c)) compared [..[8]]to the training of models preceded by either the HS or HT approach (1.4s and 61.6s; Figure 4(a) and (b)). Similar to the case with only the HS approach, the performance of the XGBoost model steadily improved during training when the HS-HT approach was used, i.e., R^2 increased from 0.52 to 0.68. In contrast, when only the HT approach is used, the model quickly achieved almost perfect performance (i.e., close to $R^2 = 1$) but gained small improvements over the rest of the training.

---

[8]removed: with
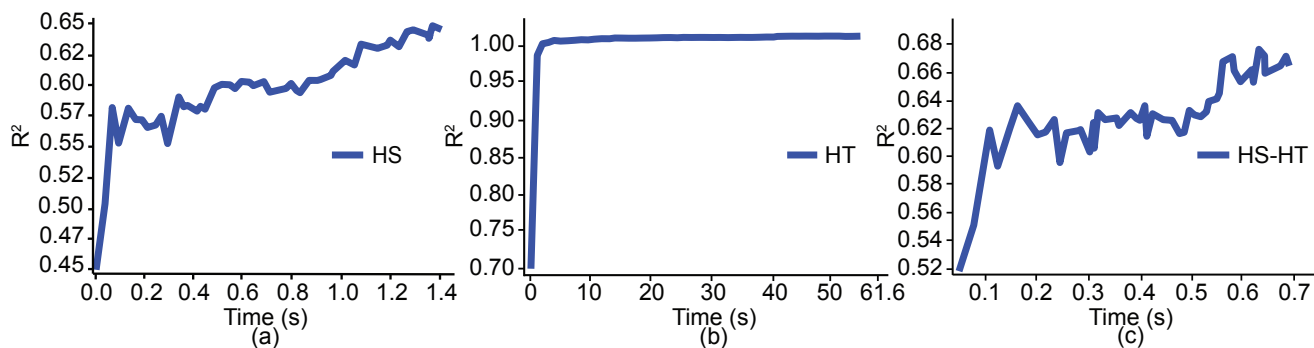
Figure 4. Comparison of the training time for the XGBoost models using hyperparameter selection (HS), hyperparameter tuning (HT) and both of them (HS-HT) as proposed by the framework with respect to their performance measured by R^2 over 8,000 iterations.

In this regard, the training process after the HT approach was not as effective compared [..[9]]to that using the HS-HT approach, although the former achieved better training performance.

However, better training performance cannot guarantee a better test performance due to the risk of overfitting. For the Maumee domain (Figure 2), the XGBoost model achieved an almost perfect agreement with the observations ($R^2 = 1.0$) when preceded by the HT approach, while having a relatively poor performance in test with $R^2 = 0.31$ (Figure 5(a)), representing a 69.0% reduction in performance. On the contrary, the XGBoost model performed worse in training when preceded by the proposed framework (i.e., the HS-HT approach) with $R^2 = 0.67$, but produced a better test performance ($R^2 = 0.40$), resulting in a smaller performance reduction, 40.3%.

Similarly, we also evaluated the overfitting of the resulting XGBoost models by directly measuring the gaps between the model performances in training at different numbers of iterations and their test performances (Figure 5(b)). Please note that once XGBoost models are trained, their test performances become irrelevant to the number of iterations during the training process and thus stay constant (Figure 5(b)). We noticed that the model preceded by the HT approach was more prone to overfitting, since the gaps measured by the differences in R^2 values were always larger than the gaps for the case with the proposed framework. As such, it demonstrated that the proposed framework can help reduce model overfitting.

As shown by Figure 6(a), using the proposed framework (i.e., the HS-HT approach), the resulting XGBoost model achieved a better agreement with the observations ($R^2 = 0.40$) than the corresponding XGBoost models preceded by other approaches ($R^2 = 0.36$ for the HS approach, $R^2 = 0.31$ for the HT approach). We noticed the relative difference in model performance for the HS and HS-HT approaches (in terms of R^2 value) is smaller than the difference for the HT and HS-HT approaches. In this regard, the XGBoost model had the worst performance if only the HT approach was used to search for the optimal values for all hyperparameters. This is further demonstrated by the comparison of residual errors between the observations and predictions [..[10]]of then XGBoost models preceded by different approaches (Figure 6(b)): for the HS-HT approach, residual

---

[9]removed: with

[10]removed: by
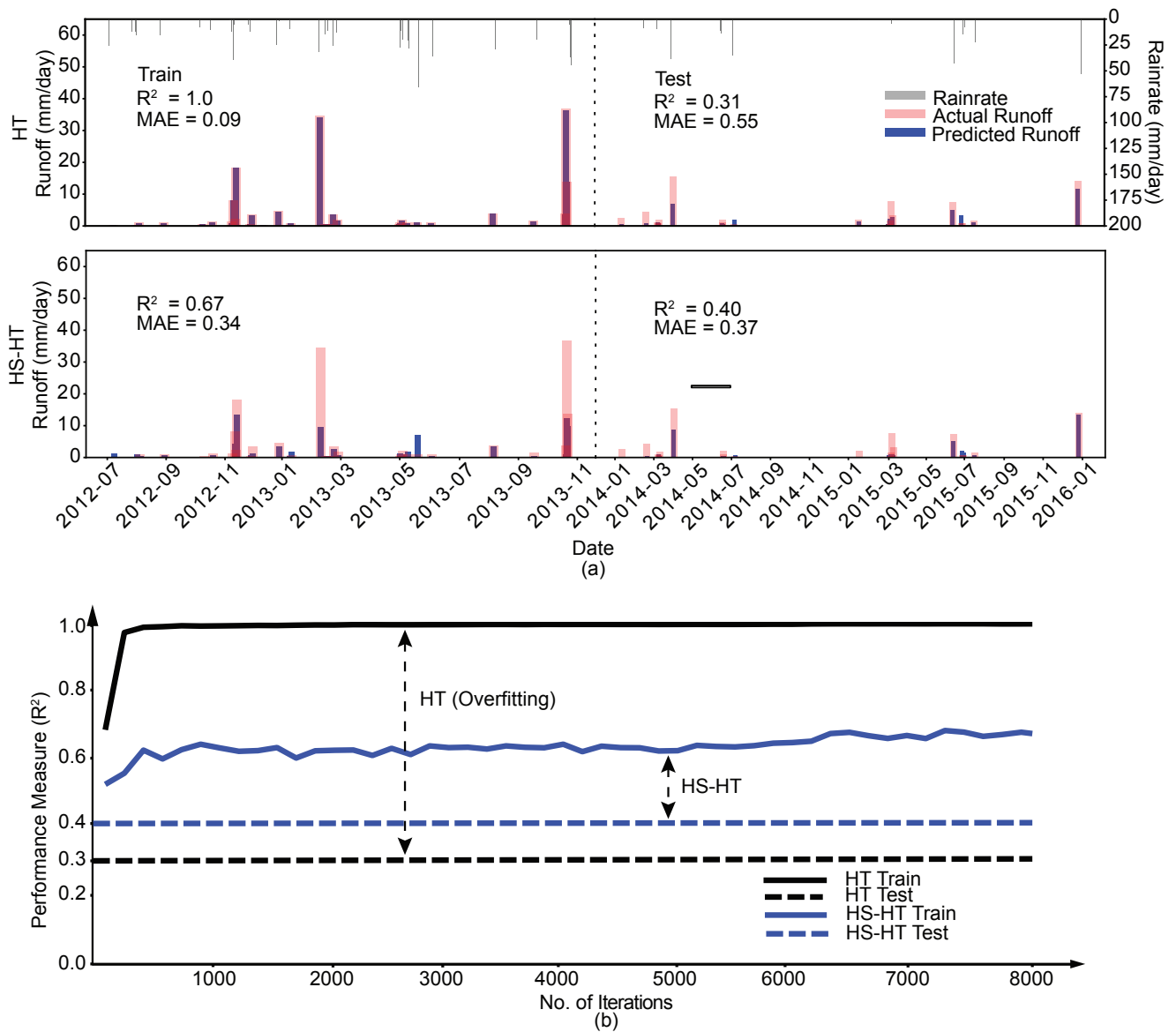
Figure 5. (a) Comparison of the performance of the XGBoost models in prediction of daily EOF runoff for the Maumee domain with respect to the observed runoff preceded by the HT approach and the proposed framework (i.e., the HS-HT approach) for a given period. The upper x-axis shows the rainfall intensity [mm/d] over the training (07/2012 - 12/2013) and test (01/2014 - 01/2016) period, respectively. (b) Comparison of the performance of the XGBoost models preceded by the two approaches, HS-HT and HT with respect to the number of iterations. The double-headed arrows indicate the differences in R^2 values between the training and test for each approach.
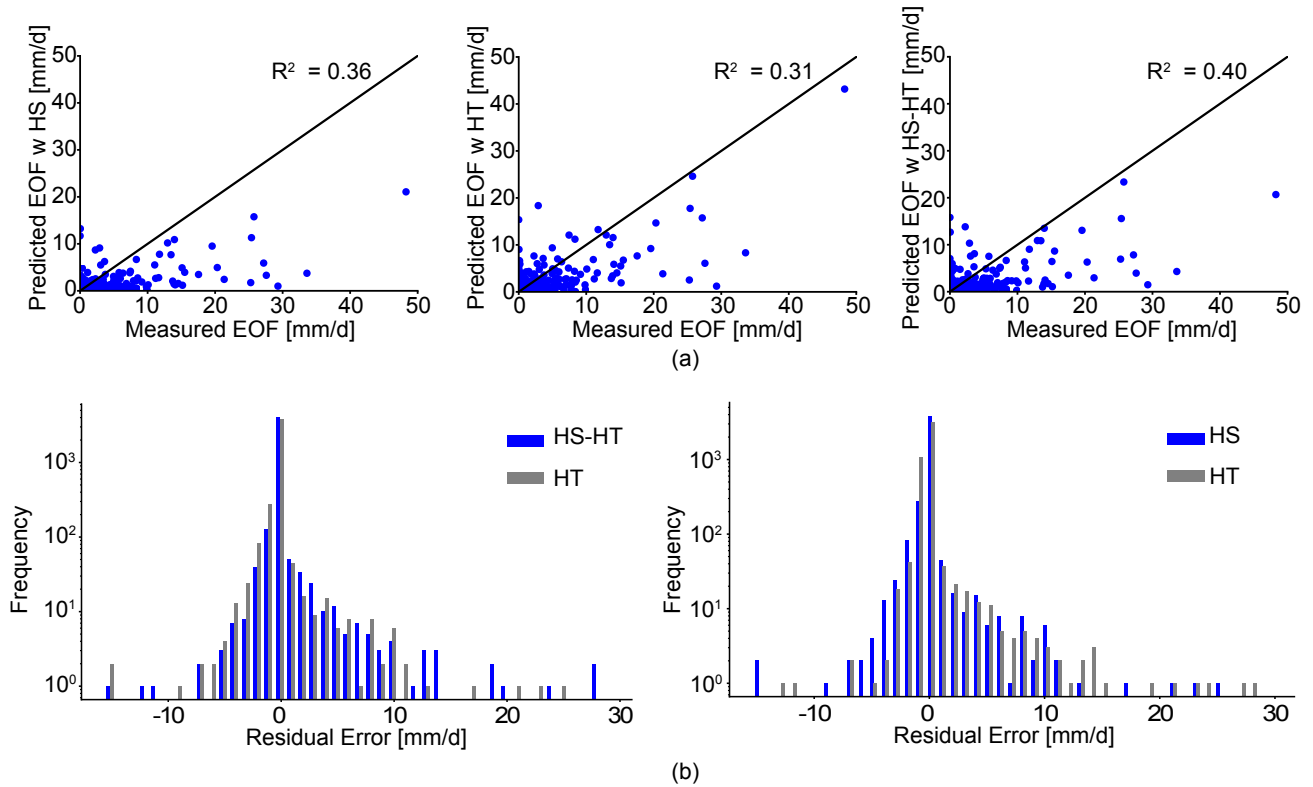
Figure 6. (a) Comparison of the observations with the predictions of daily EOF runoff for the Maumee domain by the XGBoost models trained using the HS, HT, and HS-HT approaches, respectively. (b) Comparison of residual errors between the observations and the predictions of daily EOF runoff for the Maumee domain by the XGBoost models trained using the HS-HT, HS, and HT approaches, respectively.

errors are more concentrated around the zero value compared [..[11]]to the wider scatter of errors as the result of using only the HS or HT approach, respectively. As such, the XGBoost models can often achieve better test performance when preceded by the proposed framework.

4 Discussion

In this section, we will discuss the effects of the proposed framework using hyperparameter selection and tuning on model training and the overall performance of XGBoost models. Through the discussion below, we aim to demonstrate that the results gained from this study are generally applicable to other data-driven models.

---

[11]removed: with

4.1 Influence of hyperparameters

In this study, we conducted the Sobol-based global sensitivity analysis (i.e., the HS approach) to identify the influential hyperparameters of XGBoost models. We identified three influential hyperparameters for the XGBoost model based on their sensitivity scores of the total order index (i.e., ST) and their relative differences from the first order index (i.e., S). Among them, the maximum tree depth, MD and the learning rate, LR are often considered important hyperparameters for XGBoost models, since LR is associated with model convergence and MD controls the depth of the tree model. As the depth of the tree increases, the tree model contains more inner layers, enabling it to better learn complex, nonlinear patterns from the data. However, tree models with greater depth are also more prone to overfitting.

For the learning rate (LR), a higher learning rate often leads to faster training, but the resulting tree models are more likely to reach sub-optimal solutions. In contrast, models with a low learning rate converge slowly, but are likely to have good performance with optimal hyperparameter values. Additionally, around half of its influence measured by ST is the result of interactions with other hyperparameters (Figure 3). For such a hyperparameter, we need to investigate if other hyperparameters with low ST scores should also be considered influential due to their interactions with the target hyperparameter. In our case, we decided not to consider others, mainly because the S score of the learning rate is already much higher than the ST score of the next one in the rank, i.e., the number of tree estimators, ET.

Although these two hyperparameters are considered influential in the current study, the most influential hyperparameter is the sub-sample ratio (SS) of the training data, which determines the sample size used to grow a new tree model in each boosting iteration. This is possibly due to the imbalanced data of the target variable, the daily EOF runoff, which is often zero-inflated with sparsely distributed runoff events over a long time horizon. The number of non-zero EOF runoffs in the training set, determined by the sub-sample ratio, can affect the model performance. With more zero values included in the data set, fewer non-zero EOF runoffs are available to support model training, and vice versa. As such, the sub-sample ratio appears to be the most critical hyperparameter for the performance of the XGBoost model in the study. Similar to the sensitivity analysis of physics-based models, analysis results depend on the characteristics of the target variable (e.g., the daily EOF runoff in our case). As such, for applications involving data-driven models, we can first rely on our experience to select the hyperparameters and then refine the list of influential hyperparameters using the proposed HS approach.

4.2 Algorithm complexity and model training

Data-driven models perform differently in training with and without hyperparameter selection. In general, models with more hyperparameters are more capable of learning complex, nonlinear relationships from data. In our case study, XGBoost models were initially set up with nine hyperparameters (Figure 3) to account for the complexity of daily EOF runoff prediction (Hu et al., 2021). This explains why the XGBoost model without hyperparameter selection can often achieve very good training performance (Figure 4). However, fast convergence to good training performance indicates that data patterns can be too easy for the model to learn. Furthermore, after the initial significant improvement, the performance of the XGBoost model levels off

for the majority of the training time. In this regard, the whole training is not effective, using additional training time on almost negligible improvements.

After hyperparameter selection, three out of nine hyperparameters are considered influential to the prediction of daily EOF runoff, which allows model training with a less complex XGBoost algorithm for the search of optimal model parameter values. For this reason, given the same number of iterations for training, it is thus more efficient to train the model after hyperparameter selection in terms of training time (Figure 4). Meanwhile, guaranteed by the HS approach, the removal of non-influential hyperparameters will have no or limited impact on model performance in the EOF runoff prediction. Training can also be more effective, as demonstrated by the steady improvement of the XGBoost model during the training period. As such, through hyperparameter selection, the resulting XGBoost model, equipped with fewer but influential hyperparameters, can be trained more efficiently and effectively to predict the target variable, e.g., the daily EOF runoff over the Maumee domain.

Meanwhile, XGBoost models also perform differently in training with and without hyperparameter tuning. When training an XGBoost model without the HT approach, we assign values to the hyperparameters by trial and error. The resulting XGBoost algorithm is likely not to be optimal and thus can take longer time to search for the optimal values for the model parameters compared [..[12]]to the case using hyperparameter optimization; this is demonstrated by the faster convergence to better performance when training is preceded by the HS-HT approach compared [..[13]]to that by the HS approach alone (Figure 4(a) and (c)). Nevertheless, model training preceded by the HS approach is still more effective compared [..[14]]to that using the HT approach alone (Figure 4(a) and (b)). This can be because the XGBoost algorithm with more hyperparameters (without hyperparameter selection) can more easily learn the pattern from data, resulting in no improvement in training for the majority of the training time. As such, the combination of the HS and HT approach as proposed by the framework can most effectively improve search efficiency.

4.3 Model overfitting and performance

The complexity of the underlying machine learning algorithm can be characterized by the number of hyperparameters and their values, which are critical to the model performance. High algorithm complexity can often result in overfitted models, as demonstrated by the large model performance gap in training and test (Figure 5(a) and (b)). Through the identification of influential hyperparameters, the HS approach helps reduce the algorithm complexity by using an appropriate number of hyperparameters that can balance the prediction errors and variance in the data set. As a result, reduction of algorithm complexity through the removal of non-influential hyperparameters can effectively reduce model overfitting without compromising model performance, which is further guaranteed by the use of the HT approach, searching optimal values for these influential hyperparameters.

---

[12]removed: with

[13]removed: with

[14]removed: with

4.4 Limitation and Outlook

The framework is designed to reduce model training time and improve model performance, which is done through the identification of influential hyperparameters and their optimal values. Please note that the specific results for hyperparameter selection and tuning are data- and domain-specific, and the impact of data size, quality and location is not yet fully explored in this study. Additionally, previous work (Hu et al., 2018) has demonstrated the importance of feature selection for model performance in terms of model training time and overfitting. Thus, it is worth investigating the performance of data-driven models when the framework is combined with feature selection.

5 Conclusions

In this paper, we developed a framework composed of hyperparameter selection and tuning, which can effectively improve the performance of data-driven models by reducing both model training time and model overfitting. We demonstrated the framework efficacy using a case study of daily EOF runoff prediction by XGBoost models in the Maumee domain, U.S. Through the use of Sobol-based global sensitivity analysis, hyperparameter selection enables the reduction in complexity of the XGBoost algorithm without compromising its performance in model training. This further allows hyperparameter tuning using a Bayesian optimization approach to be more effective in searching the optimal values only for the influential hyperparameters. The resulting optimized XGBoost algorithm can effectively reduce model overfitting and improve the overall performance of XGBoost models in the prediction of daily EOF runoff. This framework can thus serve as a useful tool for the application of data-driven models in EESs.

Code and data availability. Input data and codes to reproduce the study can be found here: https://doi.org/10.5281/zenodo.7026695

.

Author contributions. The conceptualization and methodology of the research was developed by YH. The coding scripts that configured the training and test data, trained the XGBoost Models, and produced figures were written by CG and YH. The analysis and interpretation of the results were carried out by CG and YH. SME produced the map of the case study site. The original draft of the paper was written by CG and YH, with edits, suggestions, and revisions provided by SME.

Competing interests. The contact author has declared that neither they nor their co-authors have any competing interests.

Acknowledgements. This work was supported by the Great Lakes Restoration Initiative through the U.S. Environmental Protection Agency and National Oceanic and Atmospheric Administration. An award is granted to Cooperative Institute for Great Lakes Research (CIGLR) through the NOAA Cooperative Agreement with the University of Michigan (NA17OAR4320152). We also thank Aihui Ma for editing the figures and the following agencies for providing us with daily EOF measurements, including USGS, USDA-ARS, Discovery Farms Minnesota, and Discovery Farms Wisconsin.

360

References

Bergen, K. J., Johnson, P. A., de Hoop, M. V., and Beroza, G. C.: Machine learning for data-driven discovery in solid Earth geoscience, Science, 363, eaau0323, 2019.

Bergstra, J., Bardenet, R., Bengio, Y., and Kégl, B.: Algorithms for hyper-parameter optimization, Advances in neural information processing systems, 24, 2011.

Bergstra, J., Yamins, D., and Cox, D.: Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures, in: International conference on machine learning, pp. 115–123, PMLR, 2013.

Campolongo, F., Saltelli, A., and Cariboni, J.: From screening to quantitative sensitivity analysis. A unified approach, Computer Physics Communications, 182, 978–988, 2011.

Chen, T. and Guestrin, C.: Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, pp. 785–794, 2016.

Dong, W., Huang, Y., Lehane, B., and Ma, G.: XGBoost algorithm-based prediction of concrete electrical resistivity for structural health monitoring, Automation in Construction, 114, 103 155, 2020.

Fleming, S. W., Watson, J. R., Ellenson, A., Cannon, A. J., and Vesselinov, V. C.: Machine learning in Earth and environmental science requires education and research policy reforms, Nature Geoscience, 14, 878–880, 2021.

Ford, C. M., Hu, Y., Ghosh, C., Fry, L. M., Malakpour-Estalaki, S., Mason, L., Fitzpatrick, L., Mazrooei, A., and Goering, D. C.: Generalization of Runoff Risk Prediction at Field Scales to a Continental-Scale Region Using Cluster Analysis and Hybrid Modeling, Geophysical Research Letters, 49, e2022GL100 667, 2022.

Frazier, P. I.: Bayesian optimization, in: Recent advances in optimization and modeling of contemporary problems, pp. 255–278, Informs, 2018.

Fushiki, T.: Estimation of prediction error by using K-fold cross-validation, Statistics and Computing, 21, 137–146, 2011.

Gimenez-Nadal, J. I., Molina, J. A., and Velilla, J.: Modelling commuting time in the US: Bootstrapping techniques to avoid overfitting, Papers in Regional Science, 98, 1667–1684, 2019.

Gomes, H. M., Barddal, J. P., Enembreck, F., and Bifet, A.: A survey on ensemble learning for data stream classification, ACM Computing Surveys (CSUR), 50, 1–36, 2017.

Hamlin, Q., Kendall, A., Martin, S., Whitenack, H., Roush, J., Hannah, B., and Hyndman, D.: Quantifying landscape nutrient inputs with spatially explicit nutrient source estimate maps, Journal of Geophysical Research: Biogeosciences, 125, e2019JG005 134, 2020.

Herman, J. and Usher, W.: SALib: an open-source Python library for sensitivity analysis, Journal of Open Source Software, 2, 97, 2017.

Hu, Y.: Edge of field runoff for the Great Lakes Region, https://doi.org/10.4211/hs.9460830270ec4d8b9d9c4260cca2114d, data retrieved from Hydroshare, 2022.

Hu, Y., Garcia-Cabrejo, O., Cai, X., Valocchi, A. J., and DuPont, B.: Global sensitivity analysis for large-scale socio-hydrological models using Hadoop, Environmental Modelling & Software, 73, 231–243, 2015.

Hu, Y., Scavia, D., and Kerkez, B.: Are all data useful? Inferring causality to predict flows across sewer and drainage systems using directed information and boosted regression trees, Water research, 145, 697–706, 2018.

Hu, Y., Fitzpatrick, L., Fry, L. M., Mason, L., Read, L. K., and Goering, D. C.: Edge-of-field runoff prediction by a hybrid modeling approach using causal inference, Environmental Research Communications, 3, 075 003, 2021.

Hutter, F., Lücke, J., and Schmidt-Thieme, L.: Beyond manual tuning of hyperparameters, KI-Künstliche Intelligenz, 29, 329–337, 2015.

Jabbar, H. and Khan, R. Z.: Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study), Computer Science, Communication and Instrumentation Devices, 70, 2015.

Kalcic, M. M., Kirchhoff, C., Bosch, N., Muenich, R. L., Murray, M., Griffith Gardner, J., and Scavia, D.: Engaging stakeholders to define feasible and desirable agricultural conservation in western Lake Erie watersheds, Environmental Science & Technology, 50, 8135–8145, 2016.

Kast, J. B., Apostel, A. M., Kalcic, M. M., Muenich, R. L., Dagnew, A., Long, C. M., Evenson, G., and Martin, J. F.: Source contribution to phosphorus loads from the Maumee River watershed to Lake Erie, Journal of Environmental Management, 279, 111 803, 2021.

Koehrsen, W.: Overfitting vs. underfitting: A complete example, Towards Data Science, 2018.

Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection, in: Proceedings of the 14th International Joint Conference on Artificial Intelligence, pp. 1137–1145, Montreal, Canada, 1995.

Liashchynskyi, P. and Liashchynskyi, P.: Grid search, random search, genetic algorithm: a big comparison for NAS, arXiv preprint arXiv:1912.06059, 2019.

Maccoux, M. J., Dove, A., Backus, S. M., and Dolan, D. M.: Total and soluble reactive phosphorus loadings to Lake Erie: A detailed accounting by year, basin, country, and tributary, Journal of Great Lakes Research, 42, 1151–1165, 2016.

NOAA National Centers for Environmental Information: Monthly National Climate Report for Annual 2011, retrieved on February 5, 2023 from https://www.ncei.noaa.gov/access/monitoring/monthly-report/national/201113., 2012.

Owen, A. B.: On dropping the first Sobol'point, arXiv preprint arXiv:2008.08051, 2020.

Prinn, R. G.: Development and application of earth system models, Proceedings of the National Academy of Sciences, 110, 3673–3680, 2013.

Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N., et al.: Deep learning and process understanding for data-driven Earth system science, Nature, 566, 195–204, 2019.

Saltelli, A.: Making best use of model evaluations to compute sensitivity indices, Computer physics communications, 145, 280–297, 2002.

Scavia, D., Bocaniov, S. A., Dagnew, A., Hu, Y., Kerkez, B., Long, C. M., Muenich, R. L., Read, J., Vaccaro, L., and Wang, Y.-C.: Detroit River phosphorus loads: Anatomy of a binational watershed, Journal of Great Lakes Research, 45, 1150–1161, 2019.

Sobol, I. M.: Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates, Mathematics and computers in simulation, 55, 271–280, 2001.

Sohl, T. L. and Claggett, P. R.: Clarity versus complexity: Land-use modeling as a practical tool for decision-makers, Journal of Environmental Management, 129, 235–243, 2013.

Stackpoole, S. M., Stets, E. G., and Sprague, L. A.: Variable impacts of contemporary versus legacy agricultural phosphorus on US river water quality, Proceedings of the National Academy of Sciences, 116, 20 562–20 567, 2019.

U.S. Geological Survey: National Land Cover Database (NLCD) 2011 Land Cover Conterminous United States, https://doi.org/10.5066/P97S2IID, u.S. Geological Survey data release, 2014.

USDA, N.: Natural Resources Conservation Service, Soil Survey Staff, Web Soil Survey, 2013.

Willard, J., Jia, X., Xu, S., Steinbach, M., and Kumar, V.: Integrating physics-based modeling with machine learning: A survey, arXiv preprint arXiv:2003.04919, 1, 1–34, 2020.

Yang, L. and Shami, A.: On hyperparameter optimization of machine learning algorithms: Theory and practice, Neurocomputing, 415, 295–316, 2020.

Yao, Y., Rosasco, L., and Caponnetto, A.: On early stopping in gradient descent learning, Constructive Approximation, 26, 289–315, 2007.

Zhu, D., Cai, C., Yang, T., and Zhou, X.: A machine learning approach for air quality prediction: Model regularization and optimization, Big data and cognitive computing, 2, 5, 2018.