Journal of Hydroinformatics

Journal of Hydroinformatics Vol 00 No 0, 1 doi: 10.2166/hydro.2025.011

Hybrid modeling for streamflow prediction in ungauged rivers with uncertainty quantification: a comparative analysis

Yichao Zeng[a], Mayank Chadha[b], Zhao Zhao[a], Sarah Miele[c], Charles J. McKnight[d], Natalie P. Memarsadeghi[d,e], Guga Gugaratshan[c], Michael D. Todd[b] and Zhen Hu[a,*]

[a] Department of Industrial and Manufacturing Systems Engineering, University of Michigan, Dearborn, MI 48128, USA
[b] Department of Structural Engineering, University of California San Diego, La Jolla, CA 92093, USA
[c] Hottinger Bruel & Kjaer Solutions LLC, Southfield, MI 48076, USA
[d] Coastal and Hydraulics Laboratory, US Army Corps of Engineers, Vicksburg, MS 39180, USA
[e] Earth System Science Interdisciplinary Center, University of Maryland, College Park, MD 20742, USA
*Corresponding author. E-mail: zhennhu@umich.edu

ZH, 0000-0003-1661-515X

ABSTRACT

Streamflow predictions are essential for flood risk prevention and long-term water resource management. Extensive research has been dedicated to forecasting river discharge for gauged rivers, where discharge data is readily available. However, accurately predicting discharge in ungauged rivers continues to pose a substantial challenge. Recent advances in machine learning (ML) offer opportunities to develop and refine predictive models for ungauged rivers. This study addresses two key limitations of existing ML models: the exclusion of sensitive physical features and overconfidence in predictions. By integrating principles of mass conservation and robust uncertainty quantification (UQ) methods, we aim to enhance the reliability of ML-based streamflow forecasts. We propose a hybrid modeling approach that leverages insights from calibrated physics-based river-routing models, considers the causal effects of carefully selected features, and integrates UQ methods to obtain probabilistic discharge predictions in ungauged rivers. Three different UQ methods within the ML models are investigated. The results of a Colorado River upper basin demonstrate that our approach outperforms existing methods (i.e., purely data-driven method), reducing the prediction error by about 40% and offering credible uncertainty bounds for risk-informed decision making.

Key words: machine learning, RAPID, streamflow prediction, uncertainty quantification

HIGHLIGHTS

- Machine learning model for streamflow prediction of ungauged rivers.
- Quantification of machine learning model prediction uncertainty.
- Metrics for quantification of the quality of uncertainty quantification.
- Comparison of different uncertainty quantification approaches.
- Demonstration of the approaches using practical case study.

1. INTRODUCTION

Streamflow prediction is vital for managing flood risks, facilitating river navigation, and ensuring sustainable water management (Lu *et al.* 2021; Zhao *et al.* 2024). In long-term water management, reliable streamflow forecasts made months in advance can greatly enhance efficiency (Chiew *et al.* 2003).

A key aspect of decision-making in this field is predicting streamflow discharge within river networks. Recent studies emphasize the need for improved discharge predictions in ungauged locations, where the lack of sensors creates forecasting challenges. River routing models, widely used in management systems, simulate hydrograph changes as water moves through networks, offering critical insights into flow dynamics. For example, the US Army Corps of Engineers (USACE) employs the Routing Application for Parallel computatIon of Discharge (RAPID) model (David *et al.* 2011) for streamflow forecasting and inland waterway management (David *et al.* 2011). Based on McCarthy's Muskingum Routing method introduced in 1938 (McCarthy 1939), RAPID has been extensively studied over the past decades (David *et al.* 2011). However, the labor-intensive

calibration process of the Muskingum method limits its efficiency and scalability (Zhao *et al.* 2023; Qin *et al.* 2024b), underscoring the need to transition to more adaptable and efficient alternatives, such as machine learning (ML) models.

Interest in ML has surged in the hydrologic community, driven by expanding hydrologic data repositories and successful applications across academic and commercial domains. Xu & Liang (2021) provide an overview for newcomers on the application and growing importance of ML in hydrology. Shamshirband *et al.* (2020) model hydrological droughts by analyzing their duration, severity, and magnitude using Support Vector Regression (SVR), Gene Expression Programming (GEP), and M5 model trees (MT), incorporating key factors such as precipitation, evapotranspiration, and runoff. Zhao *et al.* (2023) relate hydrological model parameters to river and topographic characteristics using models such as Gaussian process regression, Gaussian mixture copula, Random Forest, and XGBoost. Difi *et al.* (2024) combine ML with signal decomposition for daily streamflow forecasting, while Magni *et al.* (2023) develop a hybrid framework that improves streamflow predictions by integrating outputs from the PCR-GLOBWB global hydrological model.

Long short-term memory (LSTM), a type of recurrent neural network, is well-suited for streamflow prediction due to its ability to handle time-series data effectively (Ni *et al.* 2020; Syed *et al.* 2023; Tan *et al.* 2023; Pokharel & Roy 2024; Qin *et al.* 2024a). Rahimzad *et al.* (2021) evaluate various data-driven techniques for daily streamflow forecasting in the Kentucky River basin, finding LSTM to be the most accurate. Kratzert *et al.* (2018) propose a data-driven LSTM approach for rainfall-runoff modeling. Feng *et al.* (2020) enhance streamflow forecasts by integrating recent observations with varying schedules and types (e.g., moving average, snapshot, regularly spaced) using an LSTM model. More recently, Nearing *et al.* (2024) show that AI-based forecasting with LSTM can predict extreme riverine events in ungauged watersheds up to 5 days in advance, matching or surpassing zero-day lead-time predictions from the Copernicus Emergency Management Service Global Flood Awareness System. Data quality also plays a key role – Schutte *et al.* (2024) find that GRU and LSTM models perform reliably with commercial weather data but show moderate accuracy with freely available datasets.

While deterministic ML models have been widely adopted in hydrology for their ability to capture complex nonlinear relationships, they inherently fail to represent the uncertainties present in hydrological systems. This limitation often leads to overconfident predictions with narrow uncertainty intervals, potentially resulting in misleading conclusions and inadequate preparation for extreme events. Recognizing this, the focus has shifted toward probabilistic streamflow forecasting, which provides predictions as probability distributions or prediction intervals rather than single deterministic values. Such approaches enable stakeholders to quantify risks and make better-informed decisions under uncertainty. Consequently, uncertainty quantification (UQ) has become a critical component in advancing hydrological modeling. Several methods have been explored to address UQ in hydrology. Traditional approaches include Bayesian frameworks that integrate prior knowledge and systematically update uncertainty based on observations (Ajami *et al.* 2007). Information-theoretic and informal Bayesian methods, such as the Generalized Likelihood Uncertainty Estimation (GLUE) and limits-of-acceptability approaches, have also been reviewed extensively (Gupta & Govindaraju 2023). Techniques like first-order Taylor series expansion have been used to quantify predictive uncertainty in neural network outputs (Kasiviswanathan & Sudheer 2013), while Zhao *et al.* (2015) introduced a Bayesian joint probability model to address uncertainty in deterministic forecasts. Furthermore, Clark *et al.* (2016) emphasized the importance of explicitly accounting for hydrologic uncertainties given their implications for climate adaptation and societal resilience. In parallel, recent advances in deep learning have provided new tools for probabilistic hydrologic forecasting. Kratzert *et al.* (2019) demonstrated the effectiveness of LSTM networks for rainfall–runoff modeling, inspiring further exploration into integrating deep learning with UQ methods. Liu *et al.* (2023) propose PI3NN to enhance the trustworthiness of ML models, particularly LSTM networks, for streamflow prediction. Lu *et al.* (2021) employ Bayesian LSTM for UQ in hydrologic predictions for data-scarce rural watersheds. Techniques such as Monte Carlo dropout, deep ensembles, and Bayesian neural networks have been developed to estimate both aleatoric (data-related) and epistemic (model-related) uncertainties, thereby enhancing prediction reliability (Gal & Ghahramani 2016). Moreover, hybrid modeling approaches that integrate physics-based hydrologic models with ML frameworks have shown promise for improving predictive performance and generalization, particularly in data-scarce or ungauged basins (Nearing *et al.* 2021). Li *et al.* (2021) used stochastic variational inference in an LSTM residual error model for process-based hydrology, improving CRPS by over 10% and reliability while scaling to high-dimensional applications. Klotz *et al.* (2022) proposed an uncertainty benchmarking procedure for hydrological deep learning, showing that mixture density networks and Monte Carlo dropout provide strong baselines for accurate and nuanced predictions. In addition, Sattari *et al.* (2025) developed a probabilistic LSTM-based framework with Monte Carlo sampling and wavelet decomposition to capture

both aleatory and epistemic uncertainties in streamflow forecasting, outperforming a basic LSTM by up to 45% during extreme events such as Hurricane Harvey.

Although existing studies enhance model confidence through UQ methods, accurately predicting streamflow in ungauged rivers remains a challenge. This paper addresses probabilistic river discharge prediction in ungauged locations. The absence of discharge gauge data complicates forecasting, but by leveraging insights from calibrated physics based models such as RAPID, selecting key features, and capturing causal relationships including runoff data and neighboring gauged river discharge, it is possible to develop a hybrid model using advanced machine learning techniques. This model not only generates accurate discharge predictions for ungauged rivers but also provides uncertainty bounds. By integrating various UQ methods into ML models, it produces probabilistic forecasts essential for risk-informed decision-making. In this study, both aleatoric (data-related) and epistemic (model-related) uncertainties are explicitly quantified using probabilistic deep learning approaches, enabling a comprehensive characterization of prediction uncertainty in streamflow forecasts.

The primary contribution of this paper is a module-based hybrid modeling framework that leverages state-of-the-art ML algorithms to predict the probabilistic discharge of ungauged rivers. This is achieved through the following key components:

1. *Integrating insights from calibrated physics-based models like RAPID into hybrid model training.* This module incorporates predictions from RAPID, a first-order physics-based model grounded in conservation laws. While RAPID provides a first-order estimate, it may contain errors but serves as a causality check and effectively captures the fundamental physics of river dynamics.
2. *Incorporating carefully selected sensitive features into hybrid model training.* These features, which significantly influence river discharge, help bridge gaps in the understanding of river behavior that are not captured by first-order models like RAPID.
3. *Providing probabilistic forecasts of river discharge.* These forecasts are particularly valuable for risk-informed decision-making. UQ methods are integrated into the hybrid model to achieve this.
4. *Training the hybrid model using discharge data from gauged rivers in the target basin or neighboring rivers.* 'Neighboring rivers' refers to rivers that are geographically near the target river location and share a hydrological similarity, such as being part of the same upstream catchment or drainage basin. This enables the model to capture causal relationships between input variables and river discharge.

The framework assumes that causal relationships between input variables (e.g., runoff data, selected features), RAPID predictions, and measured discharge in gauged rivers also apply to ungauged rivers due to the neighborhood effect (i.e., similar geography, hydrology, and topology). Even in regions with few gauged rivers, discharge characteristics may vary. Training the hybrid model on multiple gauged rivers enables it to capture diverse river dynamics, including variations in peak discharge levels.

In the next section (i.e., Section 2), we provide an overview of the proposed framework, offering a bird's eye view of its structure. Subsequently, the paper's organization is discussed, followed by the details.

2. OVERVIEW OF THE PROPOSED FRAMEWORK

This section provides an overview of the proposed framework, illustrated in Figure 1. It comprises four main modules: (1) *Calibrated Physics-Based RAPID Model*, (2) *Feature Selection*, (3) *Machine Learning Data-Driven Model*, and (4) *Uncertainty Quantification* for ML models. These modules integrate systematically to form a hybrid, comprehensive streamflow prediction system.

The process begins with calibrating the physics-based RAPID model, where river network data and runoff information are assimilated to update its parameters. Next, features from multiple sources are analyzed for their sensitivity to neighboring gauged river discharge. Only the most sensitive features are selected to reduce computational costs and enhance accuracy.

The selected features, runoff data in a reduced-dimensional space, and RAPID-based discharge predictions serve as inputs to the ML model, creating a hybrid system that captures causal relationships between inputs (e.g., runoff and selected features) and river discharge while incorporating first-order river flow physics from RAPID. Training output consists of discharge data from gauged rivers near the target ungauged river. The framework obtains probabilistic discharge predictions by integrating a UQ module into the training process.

The remainder of the paper is organized as follows: Section 3 discusses the hybrid modeling approach for streamflow prediction in ungauged rivers. Section 4 covers uncertainty quantification methods for the proposed ML models. Section 5

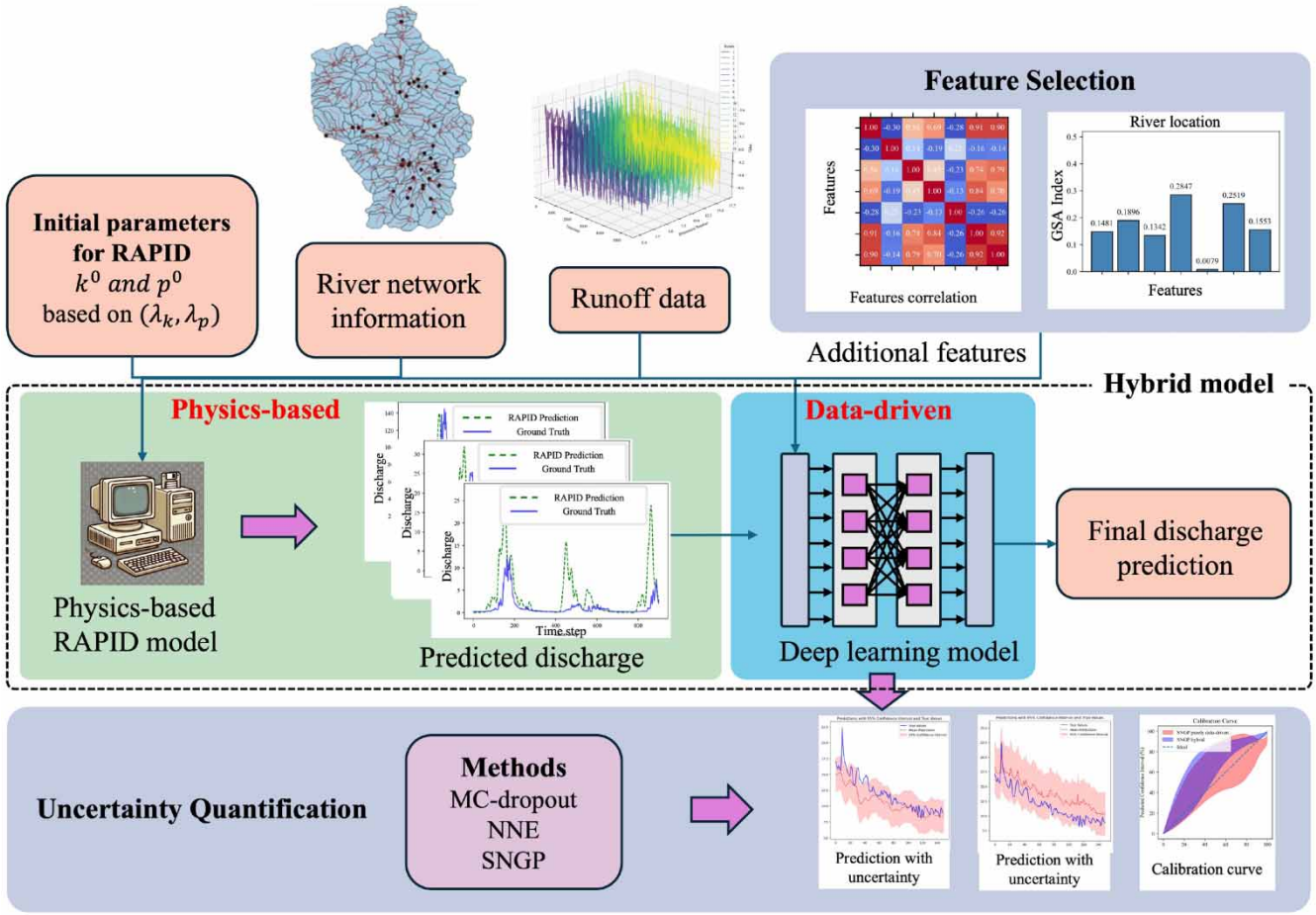

Figure 1 | Overview of the proposed framework.

presents a case study on ungauged streamflow prediction in Colorado. Section 6 outlines the model results, and Section 7 concludes with key findings and insights.

3. HYBRID MODELING FOR STREAMFLOW PREDICTION OF UNGAUGED RIVERS

Hybrid modeling integrates multiple modeling approaches to leverage their strengths and mitigate weaknesses. While numerous studies apply hybrid techniques to streamflow prediction, most do not combine physics-based and ML models. Lin *et al.* (2021) propose DIFF–FFNN–LSTM, a hybrid model integrating first-order difference (DIFF) equations, a feedforward neural network (FFNN), and LSTM for hourly streamflow prediction. Kilinc & Haznedar (2022) combine LSTM with a genetic algorithm (GA) for streamflow forecasting. Zhong *et al.* (2023) develop a model integrating physics-based approaches, specifically the variable infiltration capacity (VIC) and catchment-based macroscale floodplain (CaMa-Flood) models, with LSTM. Physics-based models can complement ML-based time series prediction by enhancing performance and incorporating physical constraints to prevent spurious predictions. These hybrid models leverage advanced ML algorithms to capture nonlinear trends and dependencies in river discharge.

This paper proposes a hybrid model integrating RAPID with either LSTM-based or Spectral Normalization Gaussian Process (SNGP) ResNet-based models. This section details the hybrid modeling framework, its components, and their collaboration to improve prediction performance.

3.1. Physics-based RAPID

RAPID is a river routing model designed to efficiently compute discharge across large, complex river networks. It is based on the Muskingum method (David *et al.* 2011), later enhanced by Cunge (1969) into the Muskingum–Cunge method, where parameters are derived from the river channel's mean physical characteristics and flow wave.

In networks with hundreds or thousands of reaches, matrices are essential for defining connectivity and calculating flow. RAPID's core is a vector–matrix implementation of the Muskingum method, as shown in Equation (1) and detailed in this section (David *et al.* 2011):

$$(\mathbf{I} - \mathbf{C}_a \cdot \mathbf{N}) \cdot \mathbf{Q}(t + \Delta t) = \mathbf{C}_a \cdot \mathbf{Q}^{\mathrm{I}}(t) + \mathbf{C}_b \cdot [\mathbf{N} \cdot \mathbf{Q}(t) + \mathbf{Q}^{\mathrm{I}}(t)] + \mathbf{C}_c \cdot \mathbf{Q}(t), \tag{1}$$

where $\mathbf{C}_a$, $\mathbf{C}_b$, and $\mathbf{C}_c$ are diagonal matrices with diagonal elements representing Muskingum method coefficients; $\mathbf{I}$ is the identity matrix, and $\mathbf{N}$ the river network matrix; $\mathbf{Q}$ is the vector of reach outflows, while $\mathbf{Q}^{\mathrm{I}}$ represents lateral inflows; t denotes time, and Δt the river routing time step.

The elements of $\mathbf{C}_a$, $\mathbf{C}_b$, and $\mathbf{C}_c$ depend on parameters k_m and p_m, where k_m is a storage constant with the dimension of time and p_m a dimensionless weighting factor representing the relative influence of inflow and outflow on volume of reach m. For any reach m, the relationship $C_{am} + C_{bm} + C_{cm} = 1$ holds. Muskingum method parameters k_m and p_m vary across river reaches.

For robust predictions, the RAPID model must be calibrated using observed data. This involves estimating the parameters k_m and p_m for all reaches ($\forall\ m$) using assumed initial values and river network information. An inverse approach is employed to optimize these parameters, minimizing a cost function that quantifies the difference between model outputs and observations.

After RAPID is calibrated, rainfall-runoff data and river properties (such as length and slope) are used to generate discharge predictions. Let $\mathbf{F}_{\mathrm{RAPID},k} \in \mathbb{R}^{N_T \times 1}$ be the RAPID predictions vector for river k, containing N_T time steps. For more details on RAPID calibration and predictions, readers are referred to (David *et al.* 2011) or Section 2 of Zhao *et al.* (2023).

In the next section, we present the methods used for feature selection in the ML models.

3.2. Feature selection for ML-based river discharge prediction

Feature selection is vital for data preprocessing and dimensionality reduction, playing a key role in ML and time-series prediction. It aims to create simpler and more interpretable models, enhance data-mining performance, and prepare clean and understandable data (Li *et al.* 2017). For the hybrid model, we gathered multiple data sources, which require preprocessing before feature selection.

- *Runoff data*: The runoff dataset comprises time series data for all rivers in the basin, each associated with a watershed. In this study, runoff data refer to surface runoff estimates simulated using the Noah with multiparameterization (Noah-MP) land surface model within the National Water Model framework (Niu *et al.* 2011). Let N_{WS} be the number of watersheds and N_T the number of time steps (observations). Define $\mathcal{S}_{\mathrm{runoff}}$ as the set of features derived from runoff, with its ith element representing the runoff time series for the ith watershed.

  Let $\mathbf{F}_{\mathrm{runoff}} \in \mathbb{R}^{N_T \times N_{\mathrm{WS}}}$ denote the runoff data matrix, where the ith column represents the runoff time series for the ith watershed. The basin of interest includes multiple watersheds and their associated rivers. Since these rivers share the same basin and interrelated dynamics, their runoff data are expected to be correlated. Therefore, rather than using the entire runoff dataset, reducing its dimensionality is advantageous. It simplifies the model, improves computational efficiency, mitigates overfitting, and enhances interpretability by identifying relevant features. To achieve this, Singular Value Decomposition (SVD) (Hu *et al.* 2017) is applied, expressing $\mathbf{F}_{\mathrm{runoff}}$ as:

$$\mathbf{F}_{\mathrm{runoff}} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T, \tag{2}$$

  where $\mathbf{U} \in \mathbb{R}^{N_T \times N_T}$ is the left singular matrix, $\boldsymbol{\Sigma} \in \mathbb{R}^{N_T \times N_{\mathrm{WS}}}$ is the diagonal matrix of singular values with σ_i as its ith diagonal element, and $\mathbf{V}^T \in \mathbb{R}^{N_{\mathrm{WS}} \times N_{\mathrm{WS}}}$ is the right singular matrix, whose columns represent the right singular vectors of $\mathbf{F}_{\mathrm{runoff}}$.

  The reduced dimensionality N_{WS_d} is chosen such that $N_{\mathrm{WS}_d} < N_{\mathrm{WS}}$, using the cumulative energy criterion to retain most of the data variance. It is computed as:

$$\mathcal{E}(N_{\mathrm{WS}_d}) = \frac{\sum_{i=1}^{N_{\mathrm{WS}_d}} \sigma_i^2}{\sum_{i=1}^{\min(N_T, N_{WS})} \sigma_i^2}. \tag{3}$$

Here, $\mathcal{E}(N_{\mathrm{WS}_d})$ denotes the variance captured by N_{WS_d} features. To retain the desired variance, N_{WS_d} is the smallest integer satisfying $\mathcal{E}(N_{\mathrm{WS}_d}) > \tau$, where τ (typically 0.90–0.95) is the variance retention threshold. Once determined, we select the first N_{WS_d} columns of $\mathbf{U}$, the top N_{WS_d} singular values in $\mathbf{\Sigma}$, and the first N_{WS_d} rows of $\mathbf{V}^T$. Let $\mathbf{U}_d \in \mathbb{R}^{N_T \times N_{\mathrm{WS}_d}}$ be the truncated left singular matrix, $\mathbf{\Sigma}_d \in \mathbb{R}^{N_{\mathrm{WS}_d} \times N_{\mathrm{WS}_d}}$ the diagonal matrix of the largest N_{WS_d} singular values, and $\mathbf{V}_d^T \in \mathbb{R}^{N_{\mathrm{WS}_d} \times N_{\mathrm{WS}_d}}$ the truncated right singular matrix. The reduced runoff matrix $\mathbf{F}_{\mathrm{runoff-SVD}}$ is given by:

$$\mathbf{F}_{\mathrm{runoff-SVD}} = \mathbf{U}_d \mathbf{\Sigma}_d = \mathbf{F}_{\mathrm{runoff}} \mathbf{V}_d. \tag{4}$$

The matrix $\mathbf{F}_{\mathrm{runoff-SVD}} \in \mathbb{R}^{N_T \times N_{\mathrm{WS}_d}}$ is a low-dimensional representation of $\mathbf{F}_{\mathrm{runoff}}$. Let $\mathcal{S}_{\mathrm{runoff-SVD}}$ denote the reduced feature space corresponding to the runoff data, which now contains N_{WS_d} features.

- *Dynamic environmental variables (DEV)*: A significant portion of the input consists of time-variant features that exhibit substantial short-term and seasonal fluctuations over time. They are highly responsive to daily environmental changes, such as evapotranspiration and temperature. A detailed description of these features in the proposed hybrid model is provided in Section 5. DEV can be grouped into separate feature sets. For instance, $\mathcal{S}_{\mathrm{evap}}$ represents a set of features evapotranspiration-derived features (containing N_{evap} variables), while $\mathcal{S}_{\mathrm{temp}}$ represents a set of temperature-derived features (containing N_{temp} variables). Since these features originate from different sources, they must share the same number of time stamps.

  Let $\mathcal{S}_{\mathrm{DEV}}$ be the combined set of DEV features, representing the union of all sources: $\mathcal{S}_{\mathrm{DEV}} = \mathcal{S}_{\mathrm{evap}} \cup \cdots \cup \mathcal{S}_{\mathrm{temp}}$. Define $\mathbf{F}_{\mathrm{DEV},k} \in \mathbb{R}^{N_T \times N_{\mathrm{DEV}}}$ as the data matrix of dynamic environmental variables for watershed k and its associated river reach. Here, $N_{\mathrm{DEV}} = N_{\mathrm{evap}} + \cdots + N_{\mathrm{temp}}$ is the total number of DEV features, and N_T the number of time stamps. Section 5 provides details on the variables considered as DEVs.
- *Quasi-static environmental variables (QSEV)*: Unlike DEV, some features remain stable over short-term records but evolve over decades due to environmental or geological processes. In our case study, these features are treated as constant over the period of interest and will be detailed in Section 5.

  QSEV features form a set where each element represents a feature that changes minimally over time. Examples include soil properties and river slope. Let $\mathcal{S}_{\mathrm{soil}}$ (containing N_{soil} soil-related features) and $\mathcal{S}_{\mathrm{slope}}$ (containing N_{slope} slope-related features) denote features extracted from watershed and river data. The combined QSEV set is $\mathcal{S}_{\mathrm{QSEV}}$, containing $N_{\mathrm{QSEV}} = N_{\mathrm{soil}} + \cdots + N_{\mathrm{slope}}$ features, and is the union of features obtained from all the sources: $\mathcal{S}_{\mathrm{QSEV}} = \mathcal{S}_{\mathrm{soil}} \cup \cdots \cup \mathcal{S}_{\mathrm{slope}}$.

  Since each feature has a unique scalar value over time, we represent the data in matrix form as $\mathbf{F}_{\mathrm{QSEV},k} \in \mathbb{R}^{N_T \times N_{\mathrm{QSEV}}}$ for watershed k and its associated river reach. The ith column denotes a vector of N_T rows, where each row holds the same scalar value for the ith feature.

3.2.1. Feature correlation analysis

Feature correlation analysis is essential for feature selection, assessing statistical relationships within a dataset. It helps identify linear dependencies, detect multicollinearity, and understand feature interactions. A widely used method is Pearson's correlation, which quantifies the linear relationship between two continuous variables. After performing correlation analysis using the Pearson correlation coefficient, we can cluster potential features. Features within the same cluster can be retained or discarded based on their importance in the feature selection process. The following section introduces two methods for selecting these features.

3.2.2. Feature selection methods

In this section, we introduce two feature selection methods used to identify the most sensitive combination of features for streamflow prediction.

(a) Gradient-based feature selection. Gradient-based feature importance is a model-based selection technique that evaluates input feature significance by analyzing the gradients of the model's output with respect to its inputs. A foundational approach, *vanilla gradients*, computes straightforward gradients to assess how changes in each feature impact the output (Wang *et al.* 2024).

This method measures how sensitive the model's output is to each input feature. Features with higher gradient magnitudes are deemed more important, as small changes in these features cause larger output variations. Tailored to the specific model,

it captures how the trained model utilizes each feature and provides detailed, sample-level importance scores, which can be aggregated for overall feature importance.

Suppose we establish a neural network model g that takes the jth sample of input vector $\mathbf{f}^{(j)} = [f_1^{(j)}, f_2^{(j)}, \ldots, f_{N_{\text{total}}}^{(j)}]$, where N_{total} indicates the total number of features in the input vector. The gradient of the output $g(\mathbf{f}^{(j)})$ with respect to an input feature $f_i^{(j)}$ is given by the partial derivative:

$$w_i^{(j)} = \frac{\partial g(\mathbf{f}^{(j)})}{\partial f_i^{(j)}}. \tag{5}$$

To assess the overall importance of each feature across all samples, we compute the average absolute gradient magnitude. For a dataset with N_{samples} samples, this is computed as Wang *et al.* (2024):

$$\bar{G}_i = \frac{1}{N_{\text{samples}}} \sum_{j=1}^{N_{\text{samples}}} \left| \frac{\partial g(\mathbf{f}^{(j)})}{\partial f_i^{(j)}} \right| = \frac{1}{N_{\text{samples}}} \sum_{j=1}^{N_{\text{samples}}} |w_i^{(j)}|, \tag{6}$$

where the quantity $\bar{G}_i$ represents the importance of the ith input feature, $|w_i^{(j)}|$ represents the absolute gradient magnitude of the function g with respect to the ith input feature for the jth sample.

This method depends on the trained and validated ML model, making it time-consuming as models must be trained before feature importance can be assessed. Therefore, an alternative, model-independent approach that efficiently selects relevant features without extensive processing is valuable. The next section introduces variance decomposition-based feature selection as such an alternative.

(b) Variance decomposition-based feature selection. Another feature selection approach is the Sobol' index, a variance-based sensitivity analysis method widely used in global sensitivity analysis (GSA) to quantify the contribution of input variables to output variance (Li & Mahadevan 2016). The Sobol' index comprises a set of variance-based sensitivity indices that partition and quantify the variance of an output as a function of contributions from various input variables and their interactions. It helps identify which input variables are the most influential in driving the variability of the output.

The Sobol' indices are typically divided into two types: the First-Order Sobol' Index and the Total Sobol' Index. The First-Order Sobol' Index measures the main effect of an individual input variable $\mathbf{f}_i = [f_1^{(i)}, f_2^{(i)}, \ldots, f_{N_{\text{samples}}}^{(i)}]$ (a vector containing all the samples of ith feature) on the corresponding output vector $\mathbf{y}$, representing the fraction of the output variance attributable to $\mathbf{f}_i$ alone. It can be expressed as:

$$S_i = \frac{Var_{\mathbf{f}_i}[\mathbb{E}(\mathbf{y} \mid \mathbf{f}_i)]}{Var(\mathbf{y})}, \tag{7}$$

where $Var_{\mathbf{f}_i}[\mathbb{E}(\mathbf{y} \mid \mathbf{f}_i)]$ is the variance of the expected value of $\mathbf{y}$ given $\mathbf{f}_i$, and $Var(\mathbf{y})$ is the total variance of $\mathbf{y}$.

The Total Sobol' Index quantifies the overall effect of the ith feature vector $\mathbf{f}_i$, including its main effect and interactions with other variables. It is given by:

$$S_{Ti} = 1 - \frac{Var_{\sim \mathbf{f}_i}[\mathbb{E}(\mathbf{y} \mid \mathbf{f}_{\sim i})]}{Var(\mathbf{y})}. \tag{8}$$

Here, $Var_{\sim \mathbf{f}_i}[\mathbb{E}(\mathbf{y} \mid \mathbf{f}_{\sim i})]$ is the variance of $\mathbf{y}$ when the input $\mathbf{f}_i$ is fixed, focusing on all other variables $\mathbf{f}_{\sim i}$. A higher Sobol' index S_i or S_{Ti} signifies greater feature contribution to output variability, emphasizing its importance. This article utilizes the method proposed by Li & Mahadevan (2016). It is a model-free approach that proposes a direct estimation of sensitivity indices using just model inputs and outputs and does not require a model that connects the input to the output.

Remark 1: On a practical note, when performing gradient-based feature selection, all three feature sources – the reduced runoff data matrix $\mathbf{F}_{\text{runoff-SVD}}$, dynamic environmental variables $\mathbf{F}_{\text{DEV},k}$, and quasi-static environmental variables

$\mathbf{F}_{QSEV,k}$ – are used as model inputs. However, variance decomposition-based feature selection considers only $\mathbf{F}_{runoff-SVD}$ and $\mathbf{F}_{DEV,k}$ to identify globally sensitive features, while quasi-static variables in $\mathbf{F}_{QSEV,k}$ remain unchanged, as they represent constant time series. For hybrid model training, data from multiple rivers are used, with their respective matrices vertically concatenated.

3.3. ML models for time-series river discharge prediction of ungauged rivers

The primary idea is to obtain all the reduced features and input data that need to be trained against the gauged river discharge time series. The premise of developing this hybrid ML model, trained using gauged rivers and then applying it to predict discharge for ungauged rivers, is based on the expectation that the model learns the causal relationship between the input features and the discharge data of neighboring rivers (with gauges). The model is expected to learn reasonably generic river dynamics, and this learned knowledge can then be used to predict the discharge of ungauged rivers, provided the necessary input data is available for the ungauged river of interest. In this paper, we primarily employ two types of deep learning models: an LSTM-based model and an SNGP ResNet-based model.

3.3.1. LSTM-based model

LSTM is a specialized recurrent neural network (RNN) architecture designed to effectively manage and utilize long-range dependencies in sequential data. Introduced by Hochreiter & Schmidhuber (1997), LSTM addresses the limitations of traditional RNNs, particularly the issues of vanishing and exploding gradients, which hindered RNNs from learning long-term dependencies effectively.

The LSTM model used in this paper begins with an input layer that processes time-series data. This is followed by three stacked LSTM layers to capture temporal dependencies across various time scales. To mitigate overfitting, dropout layers are applied after each LSTM layer. A reshape layer then formats the outputs, which are flattened before reaching the final densely connected output layer, tailored for regression tasks. This architecture allows flexibility in forecasting methods by adapting the final layer based on specific prediction needs. Section 5.2.1 provides more detailed information about the LSTM model used in this study.

3.3.2. SNGP Resnet model

While LSTM models excel at handling time-series data, they are limited in their ability to effectively quantify uncertainty, a limitation explored in the next section. To address this, integrating the SNGP with ResNet has proven beneficial (Liu *et al.* 2020), offering enhanced robustness, improved uncertainty estimation, and better generalization. To mitigate the high memory and inference costs of Bayesian neural networks and deep ensembles in real-time, industrial-scale applications, Liu *et al.* (2020) proposed SNGP, which enables high-quality uncertainty estimation using a single deep neural network (DNN). Before discussing the model, we first introduce the residual-based DNN used in our study: ResNet. Residual Networks (ResNets) mark a key advancement in deep learning, allowing the training of much deeper networks. Introduced by He *et al.* (2016), ResNets address the degradation problem, where deeper networks suffer performance deterioration due to training challenges rather than overfitting.

ResNet is based on the idea that each layer in a deep network learns a residual mapping instead of a direct input-output mapping. Mathematically, if $H(\mathbf{f})$ is the desired mapping for a set of layers and $\mathbf{f}$ is the input, ResNet models the residual function as $F(\mathbf{f}) = H(\mathbf{f}) - \mathbf{f}$. The final output is $H(\mathbf{f}) = F(\mathbf{f}) + \mathbf{f}$, implemented via skip connections that perform identity mapping, adding their outputs to the stacked layers.

To incorporate both aleatoric (data-related) and epistemic (model-related) uncertainty in ResNet and improve its robustness, the SNGP technique can be integrated. This involves two key components: Spectral Normalization (SN) and a Gaussian Process (GP) layer. The structure of the SNGP model is shown in Figure 2, with details of both components provided below.

Spectral normalization. SN is used in neural network training to control the Lipschitz constant of network layers. By constraining the spectral norm (the largest singular value) of weight matrices, SN limits how much a layer's transformation expands distances between input points. It normalizes each layer's weight matrix $\mathbf{W}$ by its spectral norm λ, estimated using the power iteration method – an efficient technique for finding a matrix's dominant singular value

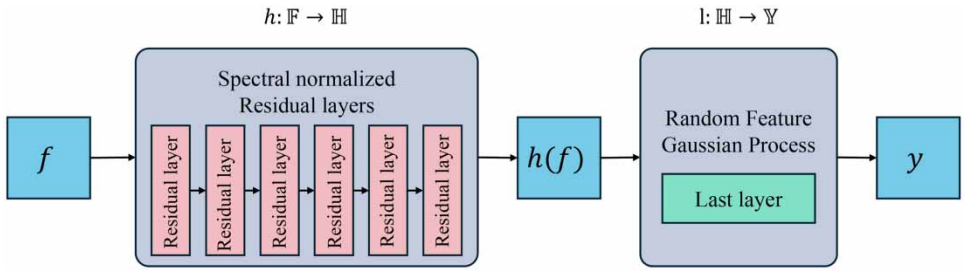

Figure 2 | The diagram of SNGP Resnet-based model structure.

(Bartlett *et al.* 2018). The normalization adjusts weights as Liu *et al.* (2020):

$$\mathbf{W}_l = \begin{cases} \frac{c \cdot \mathbf{W}_l}{\sqrt{\lambda}} & \text{if } c < \lambda \\ \mathbf{W}_l & \text{otherwise} \end{cases} \tag{9}$$

Here, $\mathbf{W}_l$ is the weight matrix of layer l, λ is the estimated spectral norm, and c is a hyperparameter that adjusts the effective upper bound of the spectral norm, ensuring $|\mathbf{W}_l|_2 \leq c$. This constraint helps maintain the desired Lipschitz continuity, ensuring that the network does not distort input distances excessively.

In ResNet, SN is applied to each residual block's weight matrix W to maintain a controlled output range. This ensures that the addition of the residual block's output to its input (i.e., $\mathbf{f} + F(\mathbf{f})$) does not lead to disproportionate scaling of the input features, thereby preserving the geometric structure in the hidden space. During training, SN is enforced at each step before forward and backward passes, stabilizing training and preventing exploding gradients in deep networks. By controlling the Lipschitz constant, it enhances generalization, mitigates overfitting, and improves robustness to input perturbations and adversarial attacks by reducing sensitivity to small input changes.

GP layer. A GP layer is incorporated into the SNGP model to enhance its distance-aware learning capabilities by replacing the standard dense output layer of ResNet with a GP layer that leverages a Laplace approximation and Random Fourier Features (RFF). This substitution helps manage computational complexity while preserving end-to-end trainability.

A GP is defined over the hidden representations $\mathbf{h} = h(\mathbf{f})$ of the input $\mathbf{f}$, where h is a mapping learned by preceding residual neural network layers. The GP output layer models $\boldsymbol{y}$ as drawn from a multivariate normal distribution ($\mathcal{MVN}$):

$$g(\mathbf{h}) \sim \mathcal{MVN}(\mathbf{0}, \mathbf{K}) \tag{10}$$

where $\mathbf{K}$ is the covariance matrix, built using a kernel function, which encodes the similarity between pairs of data points in terms of their hidden representations. Each element K_{sp} of the matrix represents the covariance between the outputs for data points s and p. In this case, $K_{sp} = \exp(-|\mathbf{h}_s - \mathbf{h}_p|^2/2)$.

To approximate the GP kernel efficiently, RFF is employed (Liu *et al.* 2020):

$$\mathbf{K} \approx \Phi\Phi^T, \tag{11}$$

where Φ is constructed from the hidden layer outputs as:

$$\Phi_{D_l} = \sqrt{\frac{2}{D_l}} \cos(\mathbf{W}\mathbf{h} + b_l). \tag{12}$$

Here, RFF samples random 'frequencies' $\mathbf{W}$ and random phases b_l. $\mathbf{W}$ consists of weights sampled from $N(0, 1)$, and b_l is sampled from a uniform distribution $U(0, 2\pi)$. D_l represents the dimensionality of $\mathbf{h}$.

Posterior inference in Gaussian Processes is computationally expensive due to the inversion of the covariance matrix $\mathbf{K}$. The Laplace approximation simplifies this by approximating the posterior around the mode (MAP estimate) of the likelihood:

$$\beta \sim \mathcal{MVN}(\hat{\beta}, \Sigma), \tag{13}$$

where $\hat{\beta}$ is the MAP estimate, and Σ is the inverse of the Hessian matrix $\mathbf{H}$ of the log posterior likelihood evaluated at $\hat{\beta}$. The posterior mean β is updated using stochastic gradient descent (Liu *et al.* 2020):

$$\beta \leftarrow \beta - \eta \nabla_\beta \left(\log p(\mathbf{h} \mid \beta) + \frac{1}{2} \|\beta\|^2 \right), \tag{14}$$

where, η is the learning rate of the GP layer.

In summary, SNGP enhances deep learning models like ResNets by integrating spectral normalization and a GP layer. Spectral normalization regulates the spectral norm of layer weights, preserving input space geometry and stabilizing training. The GP layer, leveraging a Laplace approximation and RFF, efficiently approximates the kernel matrix and models final predictions, enabling end-to-end training and improved uncertainty estimation. This setup enhances robustness, reliability, and interpretability, making SNGP ideal for critical applications requiring predictive uncertainty assessment.

3.4. Hybrid model architecture

The proposed hybrid model integrates a data-driven machine learning approach with the physics-based RAPID model, combining the former's ability to capture complex patterns with the latter's detailed physical insights into river dynamics. The next two sections introduce the purely data-driven and hybrid models.

Consider N_{gauged} gauged rivers near the ungauged river of interest. Various input data (discussed in Section 3.2) are used, depending on the feature extraction approach (see Remark 1), yielding the reduced data matrix $\mathbf{F}_{\text{reduced}} \in \mathbb{R}^{(N_T \times N_{\text{gauged}}) \times N_{\text{reduced}}}$. Let $\mathbf{F}_{\text{reduced},k} \in \mathbb{R}^{N_T \times N_{\text{reduced}}}$ represent the reduced input data for river reach k, and $\mathbf{y}_{\text{obs},k} \in \mathbb{R}^{N_T \times 1}$ the observed discharge data for gauged river k.

3.4.1. Purely data-driven model

The LSTM-based model that is purely data-driven, is trained using the *reduced-feature matrix* as input and the *observed discharge of all gauged rivers* as output. A single model is trained for all gauged rivers to effectively capture both shared and river-specific flow dynamics.

The organization of training data is crucial and is discussed here. At time step t, the model is trained on the past N_{W} time steps (including t), represented as $\mathbf{F}_{\text{reduced},k}([N_{\text{W}} - 1 : t], :) \in \mathbb{R}^{N_{\text{W}} \times N_{\text{reduced}}}$. The combined training data for all gauged rivers is given by $\mathbf{X}_{\text{train}} \in \mathbb{R}^{((N_T - N_W) \times N_{\text{gauged}}) \times N_{\text{W}} \times N_{\text{reduced}}}$, with a sample size of $((N_T - N_W) \times N_{\text{gauged}})$ and an input size per sample of $(N_{\text{W}} \times N_{\text{reduced}})$. The corresponding output is $\mathbf{Y}_{\text{train}} \in \mathbb{R}^{((N_T - N_W) \times N_{\text{gauged}}) \times 1}$. For instance, the input at time step t for river k is $\mathbf{X}_{\text{train}}(t, k, :, :) = \mathbf{X}_{\text{train},t,k} = \mathbf{F}_{\text{reduced},k}([N_{\text{W}} - 1 : t], :) \in \mathbb{R}^{N_{\text{W}} \times N_{\text{reduced}}}$, with the corresponding output $\mathbf{Y}_{\text{train}}(t, k, 1) = \mathbf{Y}_{\text{train},t,k} = \mathbf{y}_{\text{obs},k}(t + 1) \in \mathbb{R}^1$.

The number of training samples is reduced from N_T due to the sliding window approach with size N_W, yielding $(N_T - N_W + 1)$ samples. Since the output is the discharge at the next time step, the sample size is further reduced by one to $(N_T - N_W)$.

The LSTM model $\mathcal{G}$ learns the mapping from the input sequence $\mathbf{X}_{\text{train}}$ to the output sequence $\mathbf{Y}_{\text{train}}$:

$$\hat{\mathbf{Y}}_{t,k} = \mathcal{G}(\mathbf{X}_{\text{train},t,k}), \tag{15}$$

where $\hat{\mathbf{Y}}_{t,k} \in \mathbb{R}^1$ is the predicted output at time t for river k. LSTM model in this case is used as an operator that maps time-sequence inputs of river discharge drivers to the discharge prediction at t.

3.4.2. Data-driven model augmented with RAPID prediction

Unlike the purely data-driven model, the hybrid model incorporates RAPID predictions $\mathbf{F}_{\text{RAPID},k}$ alongside features $\mathbf{F}_{\text{reduced},k}$ for each gauged river k. The appended feature matrix is defined as: $\mathbf{F}_{\text{total},k} = [\mathbf{F}_{\text{reduced},k}, \mathbf{F}_{\text{RAPID},k}] \in \mathbb{R}^{N_T \times (N_{\text{reduced}} + 1)}$. The training process remains unchanged except that $\mathbf{F}_{\text{reduced},k}$ is replaced with $\mathbf{F}_{\text{total},k}$.

4. UNCERTAINTY QUANTIFICATION OF RIVER DISCHARGE PREDICTION OF UNGAUGED RIVERS

4.1. Methods for UQ of ML models

Streamflow prediction using ML techniques relies on historical data and can suffer from significant extrapolation errors under changing environmental conditions. Thus, UQ is crucial for enhancing model robustness and reliability. This section presents three UQ methods for LSTM-based and SNGP ResNet-based models: MC-dropout for LSTM, Neural Network Ensemble (NNE) for LSTM, and SNGP for ResNet.

4.1.1. MC-dropout based on LSTM

MC-dropout leverages dropout layers during both training and inference to approximate Bayesian inference. By enabling dropout at inference, the model performs multiple stochastic forward passes, capturing the distribution of possible predictions rather than a single deterministic output. Gal & Ghahramani (2016) established a theoretical framework interpreting dropout training in deep neural networks as approximate Bayesian inference in deep Gaussian processes, providing a scalable and computationally efficient means to quantify model (epistemic) uncertainty. This approach requires only a modest increase in computation compared to standard inference and is highly versatile, as it can be applied to any neural network with dropout layers. While MC-dropout addresses epistemic uncertainty, aleatory uncertainty can also be quantified when the model samples from a predictive distribution, for example, by employing a pinball loss function. Together, these capabilities make MC-dropout a widely used and practical method for uncertainty quantification in hydrological deep learning models.

In MC-dropout, each weight matrix $\mathbf{W}_l$ in a neural network layer l is multiplied elementwise with a random binary mask. This mask is sampled from a Bernoulli distribution with a dropout probability p_l:

$$z \sim Bernoulli\ (p_l). \tag{16}$$

If $z = 0$, the corresponding weight connection is 'dropped' (set to zero). This creates a sparse network for each forward pass, with randomly selected neurons deactivated.

MC-dropout approximates Bayesian inference by introducing stochasticity through dropout during training and testing. This allows the model to generate diverse outputs for the same input via multiple forward passes, effectively creating an ensemble of predictions. Mathematically, each pass is a sample from a distribution $q(\boldsymbol{\theta}; \lambda)$ over the model weights, where $\boldsymbol{\theta}$ are the weights and λ is the dropout probability. It approximates the posterior of the model weights conditioned on the input $\mathbf{X}$ and output $\mathbf{Y}$, denoted as $p(\boldsymbol{\theta} \mid \mathbf{X}, \mathbf{Y})$, using a variational distribution $q(\boldsymbol{\theta}; \lambda)$ that factorizes over the layers:

$$q(\boldsymbol{\theta}; \lambda) = \prod_{l=1}^{L} q_{\mathcal{M}}(\mathbf{W}_l). \tag{17}$$

Here, L is the number of layers in the network, $q_{\mathcal{M}}(\mathbf{W}_l)$ represents a Gaussian mixture model for each layer's weights, which approximates the posterior distribution as Nemani *et al.* (2023):

$$q_{\mathcal{M}}(\mathbf{W}_l) = p_l \mathcal{N}(M_l, \sigma_l^2 \mathbf{I}) + (1 - p_l)\mathcal{N}(0, \sigma_l^2 \mathbf{I}). \tag{18}$$

This represents a Bernoulli mixture of two Gaussians with the same standard deviation, σ_l, but different means: one Gaussian has a mean of M_l, and the other has a mean of zero.

During the testing phase, the model performs multiple forward passes (Monte Carlo sampling) with active dropout. This generates a set of predictions $\{\hat{\mathbf{y}}^{(1)}, \hat{\mathbf{y}}^{(2)}, \ldots, \hat{\mathbf{y}}^{(N_{\text{passes}})}\}$ for a given input $\mathbf{X}_{\text{test}}$. The mean prediction and uncertainty can then be estimated as Nemani *et al.* (2023):

$$\begin{aligned} \hat{\boldsymbol{\mu}} &\approx \frac{1}{N_{\text{passes}}} \sum_{n=1}^{N_{\text{passes}}} \hat{\mathbf{y}}^{(n)}, \\ \hat{\boldsymbol{\sigma}}^2 &\approx \frac{1}{N_{\text{passes}}} \sum_{n=1}^{N_{\text{passes}}} (\hat{\mathbf{y}}^{(n)} - \hat{\boldsymbol{\mu}}) \circ (\hat{\mathbf{y}}^{(n)} - \hat{\boldsymbol{\mu}}). \end{aligned} \tag{19}$$

Here, $\circ$ is the Hadamard product, or element-wise multiplication operator.

4.1.2. Neural network ensemble based on LSTM

NNE estimates uncertainty by combining predictions from multiple independently trained neural networks. Leveraging model diversity, it enhances robustness and reliability in streamflow predictions. A key advantage of NNE is its ability to reduce overfitting by averaging multiple models, which improves generalization to unseen data. Additionally, the spread of predictions naturally quantifies uncertainty. NNE is highly flexible and applicable to any neural network architecture, making it a versatile solution for various machine learning problems (Nemani *et al.* 2023).

NNE captures both aleatoric and epistemic uncertainty. To estimate epistemic uncertainty (related to the model), multiple neural networks $\{\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_M\}$ are trained independently with different random initializations. Aleatoric uncertainty, representing inherent data noise, is modeled as a variance term associated with each prediction. For input $\mathbf{X}$, each model $\mathcal{M}_m$ outputs a mean prediction $\hat{\boldsymbol{\mu}}_m(\mathbf{X})$ and an aleatoric uncertainty $\hat{\boldsymbol{\sigma}}_m^2(\mathbf{X})$. This is achieved by incorporating a final Gaussian layer into $\mathcal{M}_m$, such that the predicted discharge $\hat{\mathbf{y}}_m$ is given as:

$$\hat{\mathbf{y}}_m \sim \mathcal{N}(\hat{\boldsymbol{\mu}}_m(\mathbf{X}), \hat{\boldsymbol{\sigma}}_m^2(\mathbf{X})). \tag{20}$$

To estimate epistemic uncertainty, we use an ensemble of neural networks, $\{\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_M\}$, capturing uncertainty in model parameters due to limited or incomplete training data. For input $\mathbf{X}$, the ensemble mean prediction, $\boldsymbol{\mu}(\mathbf{X})$, is obtained by averaging the predictions from all networks as:

$$\hat{\boldsymbol{\mu}}(\mathbf{X}) = \frac{1}{M}\sum_{m=1}^{M} \hat{\boldsymbol{\mu}}_m(\mathbf{X}). \tag{21}$$

The total predictive uncertainty (which includes both aleatoric and epistemic components) is given by Nemani *et al.* (2023):

$$\hat{\boldsymbol{\sigma}}^2(\mathbf{X}) = \overbrace{\frac{1}{M}\sum_{m=1}^{M} \hat{\boldsymbol{\sigma}}_m^2(\mathbf{X})}^{\text{Aleatory uncertainty}} + \overbrace{\frac{1}{M}\sum_{m=1}^{M} (\hat{\boldsymbol{\mu}}_m(\mathbf{X}) - \boldsymbol{\mu}(\mathbf{X})) \circ (\hat{\boldsymbol{\mu}}_m(\mathbf{X}) - \boldsymbol{\mu}(\mathbf{X}))}^{\text{Epistemic uncertainty}}. \tag{22}$$

The structure of the NNE LSTM-based model is illustrated in Figure 3. Each ensemble member is trained independently with different random initializations to capture epistemic uncertainty. No explicit selection or weighting of ensemble members

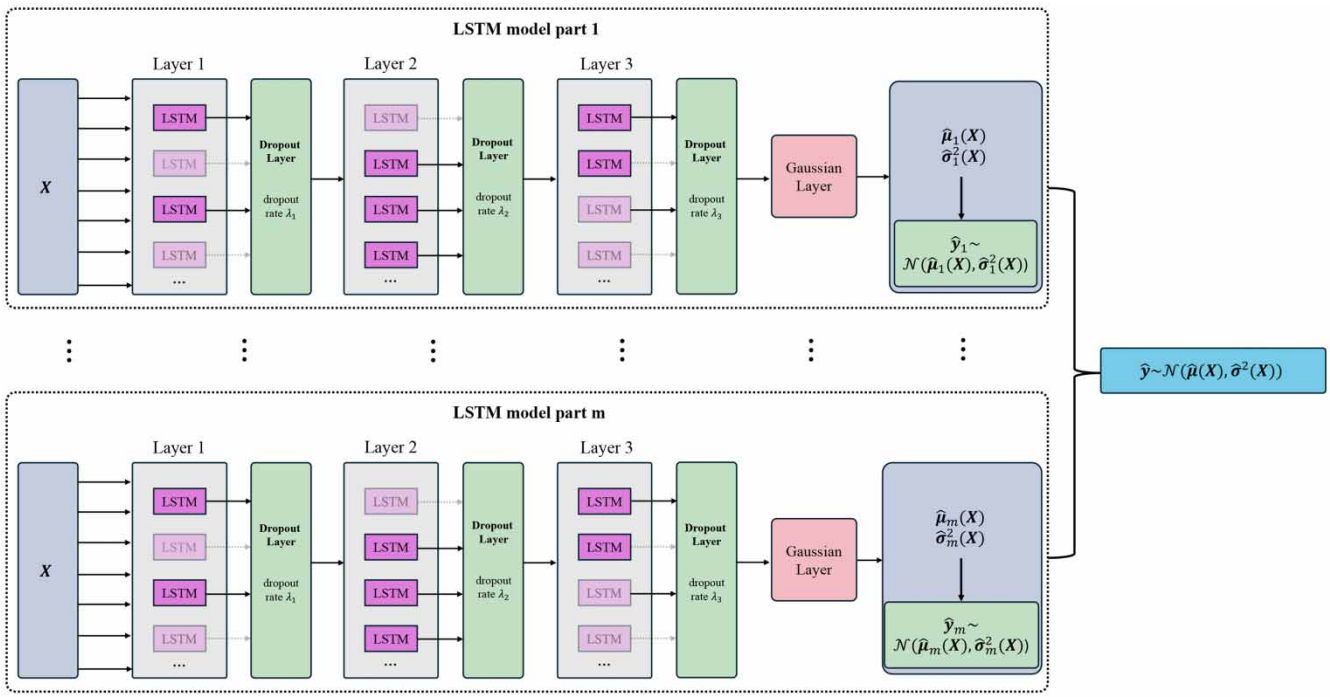

Figure 3 | The structure of the NNE LSTM-based model.

based on probabilistic criteria is performed after training. Instead, diversity in predictions is achieved through randomization in training, and each member's reliability is verified using validation performance to ensure consistency and avoid unstable predictions. This architecture adheres to the principles of the NNE and incorporates LSTM layers throughout the entire model.

4.1.3. SNGP based on Resnet

SNGP combines Gaussian processes with neural networks by normalizing spectral norms of weight matrices. This retains neural network flexibility while incorporating Gaussian process-based uncertainty quantification. A key advantage of SNGP is its improved calibration, ensuring probability estimates better reflect true prediction confidence. Spectral normalization is computationally efficient and scales well, making SNGP suitable for large-scale applications. A detailed description is given in Section 3.3.2, and the model structure is shown in Figure 2.

4.2. Evaluation method for river discharge prediction uncertainty

Validating the quality of the UQ methods requires appropriate performance evaluation. A common method for evaluating the quality of predictive uncertainty is to create a calibration curve, a technique that has been widely used in the past. For example, Zadrozny & Elkan (2002) demonstrated how to achieve accurate probability estimates for multiclass problems by combining calibrated binary estimates. The calibration curve applies to both classification and regression tasks and is constructed through three key steps (Nemani *et al.* 2023):

- *Determine the confidence level*: We first determine the confidence levels to assess. To model prediction uncertainty, we assume the probabilistic output at the ith test step, $\hat{y}_i$, follows a Gaussian distribution with the probability density function:

$$\hat{y}_i \sim \mathcal{N}(\hat{\mu}_i, \hat{\sigma}_i), \tag{23}$$

  where, for a given input $X_{\text{test},i}$ (denoting the ith row of the test input data matrix $\mathbf{X}_{\text{test}}$), $\hat{\mu}_i$ is the predicted mean (the ith element of $\hat{\boldsymbol{\mu}}(\mathbf{X}_{\text{test}})$), and $\hat{\sigma}_i$ is the standard deviation in the prediction of $\hat{y}_i$ (the ith component of $\hat{\boldsymbol{\sigma}}(\mathbf{X}_{\text{test}})$).

  For a specified confidence level $c \in [0, 1]$, we can determine a two-sided $100c$ confidence interval for the Gaussian random variable $\hat{y}_i$ as follows Nemani *et al.* (2023):

$$CI_{i,c} = \left[\hat{\mu}_i - z_{\frac{1+c}{2}}\hat{\sigma}_i, \hat{\mu}_i + z_{\frac{1+c}{2}}\hat{\sigma}_i\right], \tag{24}$$

  Here, $z_{(\frac{1+c}{2})}$ represents the $(\frac{1+c}{2})$th quantile of the standard normal distribution, given by $z_{(\frac{1+c}{2})} = \Phi^{-1}(\frac{1+c}{2})$, where $\Phi(\cdot)$ denotes the cumulative distribution function of the standard normal distribution.
- *Calculate the observed confidence*: To assess the reliability of the model's uncertainty estimates, the observed confidence level is computed by comparing predictions to actual outcomes. This involves checking how often the predicted confidence intervals contain the true values. For example, at a 95% confidence level, the proportion of intervals including the true values should ideally approach 95%. This involves checking how often the predicted confidence intervals contain the true values. For example, at a nominal 95% confidence level, the proportion of intervals including the true values should ideally approach 95%, indicating good calibration. As discussed in Khosravi *et al.* (2011), this concept is formalized through the Prediction Interval Coverage Probability (PICP) metric, which measures the fraction of true observations that fall within the predicted intervals. A well-calibrated model should exhibit a PICP close to the nominal confidence level, providing a direct indication of the model's uncertainty reliability.
- *Plot observed vs. expected confidence*: Finally, the results are visualized by plotting observed confidence against expected confidence levels. In a well-calibrated model, this plot aligns with the diagonal from bottom-left to top-right, indicating observed confidence matches expected confidence. Deviations from this line reveal potential calibration issues. This evaluation using calibration curves follows the same principle outlined by Renard *et al.* (2010), which recommends assessing predictive reliability through consistency between nominal and empirical coverage.

Beyond the calibration curve, several metrics assess uncertainty quantification performance. Root mean square error (RMSE) measures the difference between predicted and actual values and is used here to evaluate point prediction accuracy only. To assess the quality of predictive intervals and probabilistic forecasts, we use metrics such as the calibration curve and Negative

Log-Likelihood (NLL). The calibration curve evaluates how well the predicted probabilities or intervals align with the actual observations, ensuring proper coverage. NLL quantifies how well the predicted distribution matches the observed data (Nemani *et al.* 2023). A detailed description of NLL is provided in Section 5.3. Together, these metrics comprehensively evaluate a model's ability to produce accurate point predictions and reliable uncertainty estimates.

5. CASE STUDY

The United States is divided into seven main catchment zones. Building on Zhao *et al.* (2023), this study focuses on part of the upper Colorado River basin, which includes 431 rivers. This basin features a range of discharge rates, including rivers regulated by dams. However, discharge data is available at only six locations, identified by USGS IDs: 700739316, 700749972, 700777836, 700782048, 700811946, and 700825284, highlighted in Figure 4. Our objective is to use discharge data from select gauged rivers to train predictive models. The remaining gauged rivers are treated as ungauged for testing and validation, with predicted discharge rates compared against observed data to assess model accuracy. The six river locations are split into training and test sets, with five randomly selected for training and one for test. This simulates using gauged river data to predict discharge rates for ungauged rivers.

This case study requires two main datasets. The first is the input data, which includes all river-related features: time-series data from Climate Engine ($\mathbf{F}_{\mathrm{DEV},k}$), runoff data after dimension reduction ($\mathbf{F}_{\mathrm{runoff-SVD}}$), constant features such as soil properties, elevation, and slope ($\mathbf{F}_{\mathrm{QSEV},k}$), and RAPID discharge predictions ($\mathbf{F}_{\mathrm{RAPID},k}$). The second dataset is the model's output, the gauged discharge data for selected river locations, $\mathbf{y}_{\mathrm{obs},k}$. The next section details these datasets.

5.1. Raw data extraction and preprocessing

The dataset includes gauged river discharge data and river network details of the Colorado Basin for training and calibrating the RAPID model. Selective time-series features (DEV) were obtained from ClimateEngine.org (https://app.climateengine.org/climateEngine), which offers an application and API that integrates climate and remote sensing data for environmental analysis. Eight features were selected from the GridMET dataset (https://www.climatologylab.org/gridmet.html), which provides daily high-resolution (~ 4 km, 1/24th degree) surface meteorological data for the contiguous United States since 1979. These features include Mean Temperature, Precipitation, Hargreaves Potential Evapotranspiration, Specific Humidity, Wind Speed, Downward Shortwave Radiation, Vapor Pressure Deficit, and Burning Index, recorded as daily time-series data.

The slowly changing environmental features (QSEV) are gathered from various data sources. The first component to consider is soil data. Soil plays a vital role in streamflow prediction as it regulates water storage and release within a watershed. It influences the proportion of precipitation that infiltrates the ground versus becoming runoff, thereby affecting streamflow. The soil's moisture content determines whether water will infiltrate into the soil or flow across the surface during rainfall,

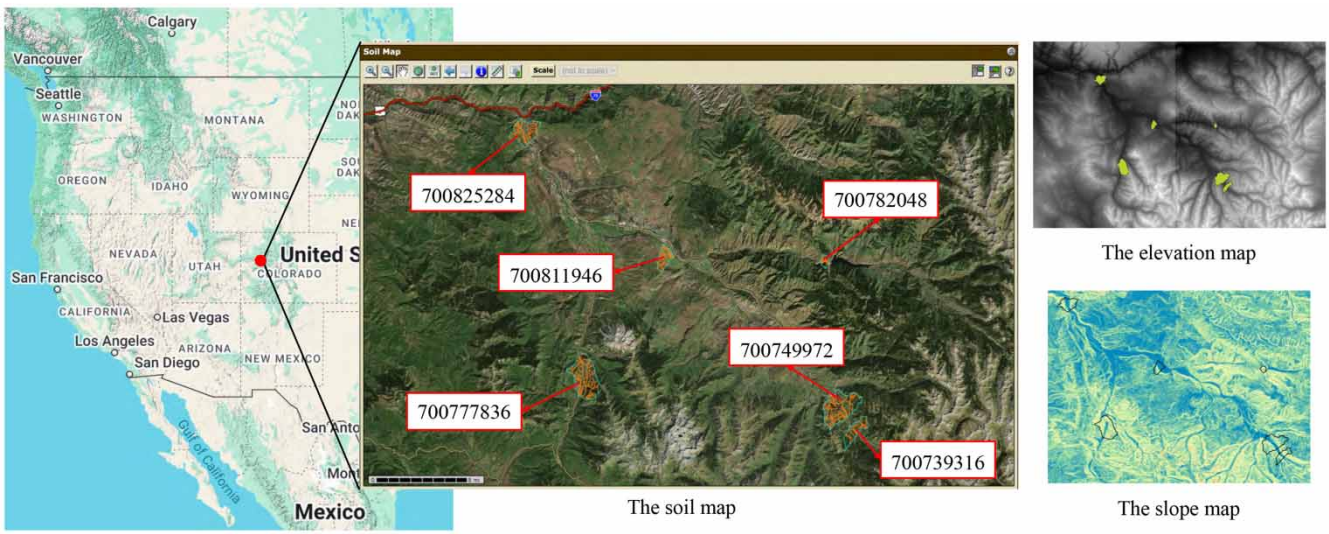

Figure 4 | The map of the six gauged river locations of interest.

with dry soils promoting infiltration and wet soils increasing surface runoff. The soil data is sourced from the Web Soil Survey (WSS) (https://websoilsurvey.sc.egov.usda.gov/app/), which offers comprehensive soil data and information produced by the National Cooperative Soil Survey. This resource includes various features, such as hydrologic soil groups, among others. Figure 4 displays the soil map for the gauged river locations, highlighting the soil features pertinent to this case study. The raw data for the rest of the environmental features such as elevation and slope are sourced from the USGS online data portals, including The USGS Earth Explorer (https://earthexplorer.usgs.gov/), NHDPlus HR (https://www.usgs.gov/national-hydrography/nhdplus-high-resolution), and The National Map Apps (https://apps.nationalmap.gov/downloader/). By uploading the relevant shapefile, users can access Landsat satellite imagery, radar data, and more. We utilize the seamless map provided by the 3D Elevation Program (3DEP) with a resolution of 1/3 arc second. The seamless 3DEP data are processed from diverse sources to maintain a consistent coordinate system and unit of vertical measurement. This data is distributed using geographic coordinates in decimal degrees, adhering to the North American Datum of 1983 (NAD 83). All elevation values are measured in meters and, for the contiguous United States, are referenced to the North American Vertical Datum of 1988 (NAVD 88). Two map sections covering the six river locations can be accessed through links: https://www.sciencebase.gov/catalog/item/61b19ba7d34eb8f531255a1b and https://www.sciencebase.gov/catalog/item/620de4aed34e6c7e83ba9fdc. Figure 4 also illustrates the elevation map and the corresponding slope map derived from it. The image data can be processed using geographic information system (GIS) software to extract key statistics such as the mean, median, minimum, maximum, and standard deviation values for the selected river locations. Given that much of the Colorado region is mountainous with highly variable terrain and slopes, the mean and standard deviation of elevation and slope for each river location are primarily chosen as the features for this case study.

One of the input features not yet mentioned is the runoff dataset. This dataset is considered large as it contains information on 431 rivers within the area of interest, often leading to redundancy. To address this, the SVD technique is applied to the runoff data, $\mathbf{F}_{\text{runoff}}$, reducing its dimensionality from 431 to 18. The resulting reduced data matrix, $\mathbf{F}_{\text{runoff}-\text{SVD}}$, effectively captures the non-redundant information contained in $\mathbf{F}_{\text{runoff}}$.

Table 1 provides an overview of all features used in this case study. The abbreviation of the feature names will be used in the following sections. After data preprocessing, all the data is compiled and utilized for the subsequent feature selection process.

5.2. Hybrid model setup

The hybrid model integrates the physics-based RAPID model with ML using selected features. The following sections detail two ML models: an LSTM-based model (Section 5.2.1) and a ResNet-SNGP model (Section 5.2.2).

Table 1 | An overview of features in this case study

Full term	Dimension	Feature type	Abbreviation
Reduced runoff data	18	Time-series	M^3
RAPID predictions	1	Time-series	RP
Soil data	10	Constant (QSEV)	SD
Mean temperature	1	Time-series (DEV)	MT
Hargreaves potential evapotranspiration	1	Time-series (DEV)	HE
Wind speed	1	Time-series (DEV)	WS
Precipitation	1	Time-series (DEV)	PR
Vapor pressure deficit	1	Time-series (DEV)	VP
Specific humidity	1	Time-series (DEV)	SH
Downward shortwave radiation	1	Time-series (DEV)	SR
Elevation difference	1	Constant (QSEV)	ED
Elevation max and min	1	Constant (QSEV)	EM
Slope mean	1	Constant (QSEV)	SM
Slope standard deviation	1	Constant (QSEV)	SS

5.2.1. LSTM-based model setup

LSTM-based models are utilized for both streamflow prediction and uncertainty quantification. Establishing a solid foundational structure and selecting appropriate hyperparameters for the LSTM model can significantly enhance its performance. The structure of the LSTM model is discussed in Section 3.3.1, and the following text introduces the hyperparameters used in the case study.

Hyperparameters were fine-tuned using a grid search strategy across a predefined search space, including 1–4 layers, 50–200 units per layer, dropout rates from 0.0 to 0.3, and learning rates ranging from 0.0001 to 0.001. The optimal input window size was selected from 10 to 100 days. Each configuration was assessed based on validation performance and calibration criteria to ensure both accuracy and robust uncertainty quantification. The optimal LSTM model consists of three layers, each with 100 units, providing a moderately complex architecture capable of capturing intricate data patterns while ensuring computational efficiency. The choice of 100 units per layer balances the model's capacity to learn detailed features with its operational efficiency, supporting scalability and responsiveness during training and inference. A batch size of 32 is employed based on experience. The model is trained for 200 epochs, a duration deemed sufficient for convergence on effective parameters without overfitting, especially with the application of regularization techniques like dropout. A dropout rate of 0.1 is applied in most cases (except in MC-dropout), meaning 10% of neurons are randomly excluded during each training iteration, which helps prevent reliance on individual neurons and enhances network resilience. The learning rate is set to 0.0001, a relatively low value that ensures stable and gradual convergence by making small adjustments to the weights, minimizing the risk of overshooting the optimal loss values during training. Finally, the Adam optimizer is employed for its efficiency in handling sparse gradients and its adaptive learning rate capabilities.

The optimal input window size is determined to be 35 days before the prediction date, with the output being the next day's streamflow discharge. This was determined through a grid search over 10 to 100 days, considering practical factors such as seasonal variations. The 35-day window was selected for its balance between predictive accuracy, computational efficiency, and reduced overfitting risk.

5.2.2. SNGP setup

As previously discussed, the SNGP model enhances robustness and uncertainty estimation in deep learning by integrating spectral normalization and Gaussian processes within a residual network. For the SNGP model used in this paper, it consists of six spectrum norm layers with each layer having 32 units. The spectrum norm bound used is 0.9. The learning rate is set to 0.0001 and a dropout rate of 0.1 is used to avoid overfitting. The model is trained using 1,000 epochs and the best model is saved using callback.

5.3. Uncertainty quantification methods

Building on the benchmark models from Nemani *et al.* (2023), the UQ models adopt similar architectures with modifications such as LSTM cells and varying hyperparameters. This section details three approaches: MC-dropout (LSTM-based), Neural Network Ensemble (LSTM-based), and SNGP (ResNet-based), outlining their structures and hyperparameters. To ensure the reliability of predictive intervals and guide decisions on sampling size, we evaluate probabilistic performance using calibration curves, NLL, and PICP. These metrics comprehensively assess how well the predicted uncertainty captures the true observations, supporting robust decision-making and ensuring the generated intervals are informative and reliable.

5.3.1. MC-dropout based on LSTM

We develop an MC-dropout model following the methodology in Section 4.1.1. This model comprises three LSTM blocks, each equipped with a dropout layer that remains active during the prediction phase. The dropout rate for these layers varies randomly within the range of [0.05, 0.15]. This approach ensures that while an individual LSTM model might produce predictions with varying levels of confidence, the ensemble of models collectively generates reliable forecasts with a well-defined uncertainty range. The loss function utilized is Mean Squared Error (MSE) loss. The Elbow Method is employed to determine the optimal ensemble size for our case study. The Elbow Method determines the optimal ensemble size by testing values from 2 to 16, identifying 6 as optimal. The model is iterated 15 times to capture run-to-run variation.

5.3.2. Neural network ensemble based on LSTM

As discussed in Section 4.1.2, each individual model in the ensemble incorporates a Gaussian layer as its final layer. For instance, in the mth model at the ith time step during the training process, this Gaussian layer outputs a predicted mean,

$\hat{\mu}_{m,i}$ (i.e., the ith element of $\hat{\boldsymbol{\mu}}_m(\mathbf{X}_{\text{train}}; \boldsymbol{\theta})$), and variance $\hat{\sigma}^2_{m,i}$ (i.e., the ith element of $\hat{\boldsymbol{\sigma}}^2_m(\mathbf{X}_{\text{train}}; \boldsymbol{\theta})$). Here, $\boldsymbol{\theta}$ represents the parameter vector for the Neural Network model. These parameters are optimized by minimizing the Negative Log-Likelihood (NLL) loss, which is expressed as:

$$\text{NLL Loss for model } m : \mathcal{L}_m(\boldsymbol{\theta}) = \sum_{i=1}^{(N_T - N_W)} \left[\frac{\log \hat{\sigma}^2_{m,i}}{2} + \frac{(Y_{\text{train},i} - \hat{\mu}_{m,i})}{2\hat{\sigma}^2_{m,i}} \right]. \tag{25}$$

The Elbow Method is utilized to determine the optimal ensemble size for this case study. By varying the ensemble size from 2 to 16, it was observed that an ensemble size of 4 is optimal for the NNE model. Subsequently, this model is iterated 15 times to account for run-to-run variation.

5.3.3. SNGP based on Resnet

As discussed in Section 5.2.2, the SNGP model inherently quantifies uncertainty. It outputs the predicted next-day discharge and its associated standard deviation. To capture run-to-run variation, the model is iterated 15 times.

6. RESULTS AND DISCUSSION

6.1. Feature selection analysis results

As discussed in Section 3.2.1, the Pearson correlation coefficient is used to identify linear dependencies between time-series streamflow features. The Pearson correlation coefficients indicate a clear linear relationship between different features at six river locations of interest. The analysis reveals that certain features exhibit strong linear correlations over time. For instance, some features share similar seasonality patterns over extended periods, resulting in high linear relationships. Consequently, these features are grouped, and only a subset from each group is selected as potential features. One distinct group of features that exhibits a high degree of correlation includes *mean temperature*, *vapor pressure deficit*, and *Hargreaves potential evapotranspiration*. After that, variance decomposition-based feature selection was applied to all six river locations, analyzing all time-series features. Figure 5 presents the results, showing variations in feature importance across sites. Overall, *downward shortwave radiation* (SR) and *Hargreaves potential evapotranspiration* (HE) emerged as the most significant features. Similar results were found using gradient-based feature selection, as shown in Table 2, where M^3_{10} represents the 10th element in the *reduced runoff data*. Table 3 summarizes model validation performance with different feature combinations. Notably, including more features does not always improve performance. The best-performing model uses only *runoff data* (M^3), *RAPID predictions* (RP), *soil data* (SD), *evapotranspiration* (HE), and SR. For efficiency, subsequent research focuses on this feature set.

One river in the training set has a dam, leading to controlled flow patterns distinct from natural rivers. This discrepancy affects model predictions. To evaluate its impact, we removed this human-controlled river (river 4) from the training set. The results showed worsened performance, indicating its inclusion benefits prediction accuracy. Therefore, the human-controlled river remains in the training set.

6.2. Model performance and the results of uncertainty quantification analysis

6.2.1. MC-dropout

Figure 6 presents a detailed prediction comparison of a selected region over different time periods of prediction with and without RAPID using MC-dropout. In this figure, the discharge values for both actual and predicted data are shown over time. The predictions include the mean values bounded by confidence intervals, which represent both the predicted values and their associated uncertainties. To generate Figure 6, we focus specifically on the peaks of the time-series data, as accurately predicting peak discharge is critical in practice. During the intervals from days 0 to 300, 400 to 600, 1,150 to 1,350, 2,950 to 3,200, and 4,000 to 4,300, the hybrid model clearly outperforms the purely data-driven model. However, for the peak occurring between days 1,500 and 1,700, both models overpredict discharge values before and after the peak. This discrepancy is primarily attributed to the relatively smaller size of this peak compared to others, which is closely linked to unexpected environmental conditions during that period. Overall, the hybrid model, incorporating MC-dropout with a 95% confidence interval, successfully captures most true values and demonstrates superior prediction performance compared to the purely data-driven model.

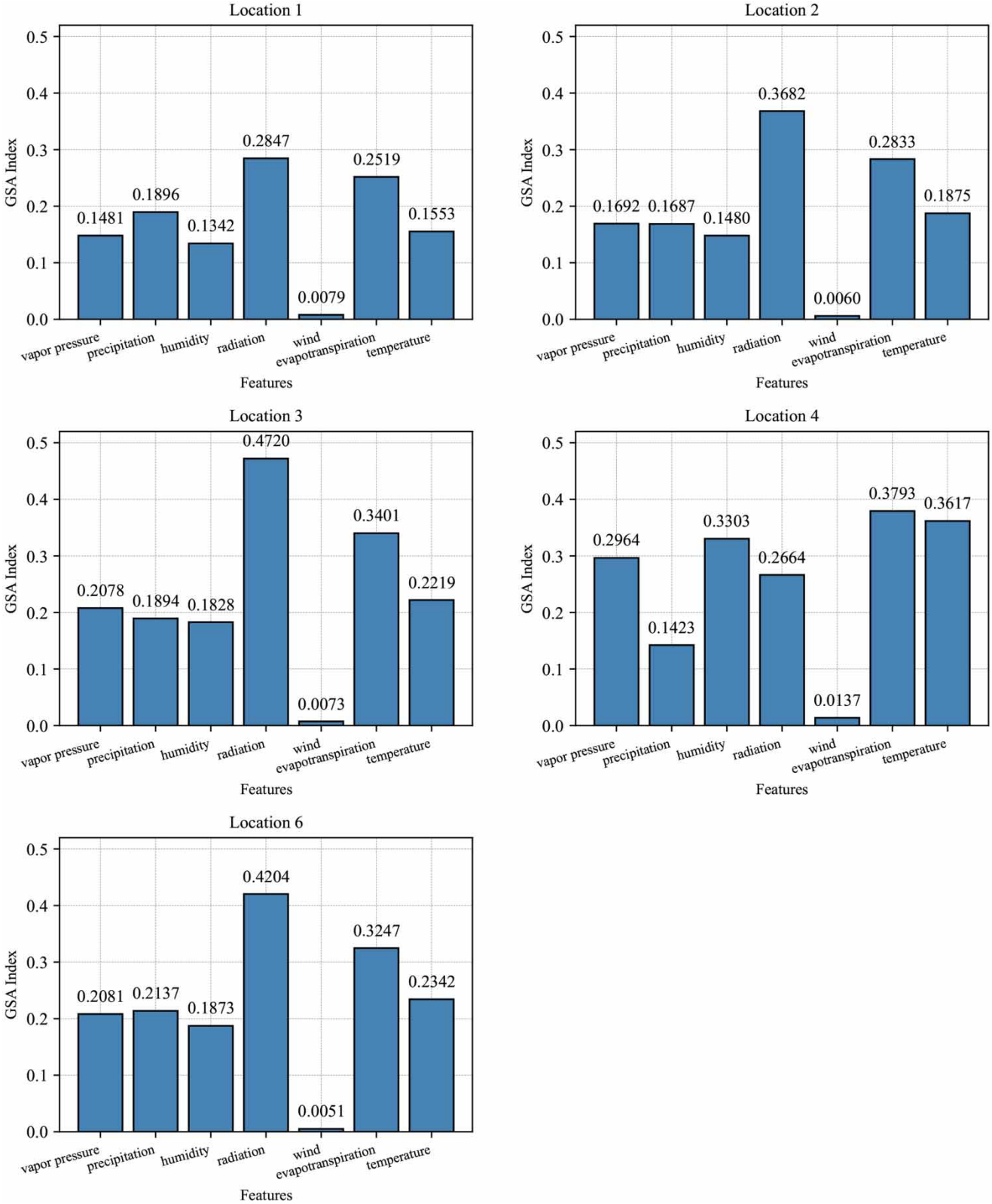

Figure 5 | Variance decomposition-based feature selection.

Table 2 | Top 5 features using gradient-based feature selection

#	Feature name
1	Specific humidity
2	Hargreaves potential evapotranspiration
3	Downward shortwave radiation
4	Reduced runoff M_{10}^{3}
5	RAPID

Table 3 | LSTM-based model validation performance using different input feature combinations

Features combination	Features	*MSE*	*RMSE*
M^3, RP, SD, MT, HE, WS, PR, VP, SH, SR, ED, EM, SM, SS	40	62.43	7.90
M^3, RP, SD, MT, HE, WS, PR, VP, SH, SR, SM, SS	38	**53.94**	**7.34**
M^3, RP, SD, MT, HE, WS, PR, VP, SH, SR	36	90.97	9.54
M^3, RP, SD, HE, SR, SM, SS	33	55.59	7.46
M^3, RP, SD, HE, SR, ED, EM	33	65.64	8.10
M^3, RP, SD, HE, SR	31	**49.13**	**7.01**
M^3, RP, SD	29	162.57	12.75

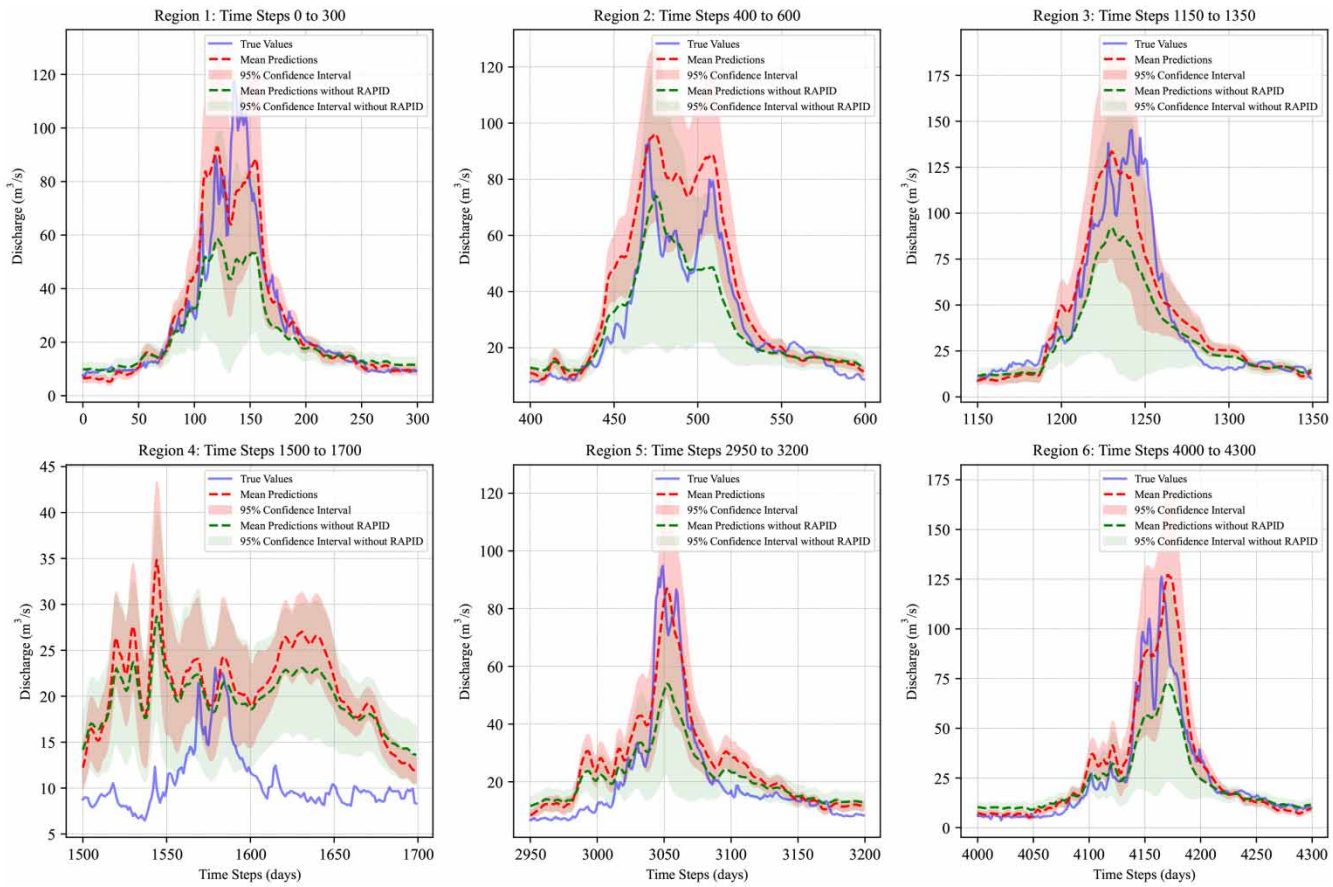

Figure 6 | Divided time-series prediction plot with 95% confidence interval and true values (MC-dropout) between purely data-driven and hybrid model.

Figure 7(a) presents the violin plot that compares the absolute errors between the models' predictions and observed discharge values using MC-dropout. This violin plot shows the distribution of absolute errors for both models, which is crucial for assessing model performance in terms of error minimization. By comparing the height and spread of the violins for each model, we can discern which model tends to have lower prediction errors, indicating better performance. The plot highlights areas where errors are most concentrated, showing the commonality of error sizes. A taller, narrower violin suggests more consistent error sizes, whereas a wider base indicates greater variability. This plot reveals that the purely data-driven model has a wider error distribution, suggesting less reliability and potentially higher average errors, including more high-value errors. Overall, these findings demonstrate that the hybrid model using MC-dropout outperforms the purely data-driven model, underscoring the benefits of using a hybrid approach for prediction.

6.2.2. Neural network ensemble

Figure 8 illustrates the comparisons between model predictions with and without RAPID using NNE of different time periods. Similar to MC-dropout methods, the discharge values for both actual and predicted data are depicted over time, showcasing how well each model captures the dataset's dynamics over a 5,000-day period. The prediction values include the mean predictions and the previously discussed confidence intervals, which represent both the predicted values and their associated uncertainties. Six peaks have been selected from the entire time period shown in Figure 8. It is evident that from days 0 to 300, 400 to 600, 1,150 to 1,350, 2,950 to 3,200, and 4,000 to 4,300, the hybrid model surpasses the purely data-driven model in performance. It is important to highlight that both models effectively capture the peak between days 1,500 to 1,700, a detail that the MC-dropout model overlooks. Overall, the hybrid model using NNE with a 95% confidence interval successfully captures most true values and delivers superior prediction performance compared to the purely data-driven model. Additionally, the performance of both NNE models surpasses that of the MC-dropout models.

Figure 7(b) presents the violin plot that illustrates the absolute errors between the models' predictions and observed discharge values using NNE. This plot reveals that the purely data-driven model has a wider error distribution, suggesting less reliability and potentially higher average errors, including more high-value errors. Overall, these findings demonstrate that the hybrid model using NNE outperforms the purely data-driven model, underscoring the benefits of using a hybrid approach for prediction. This conclusion is consistent with the results observed from the MC-dropout models.

6.2.3. Spectral normalization Gaussian process

Figure 9 illustrates detailed comparisons between model predictions with and without RAPID using SNGP for different time periods. It is evident that from days 0 to 300, 400 to 600, 1,150 to 1,350, 2,950 to 3,200, and 4,000 to 4,300, the hybrid model surpasses the purely data-driven model in performance. For the peak occurring between days 1,500 and 1,700, both models underpredict discharge values before and after the peak. However, the hybrid model still outperforms the purely data-driven model by better capturing the peak rather than failing to detect it. One notable feature of the SNGP model is that it has a considerably wider confidence interval compared to the other two UQ methods. Overall, the hybrid model using SNGP with a 95% confidence interval successfully covers most true values and delivers superior prediction performance compared to the purely data-driven model.

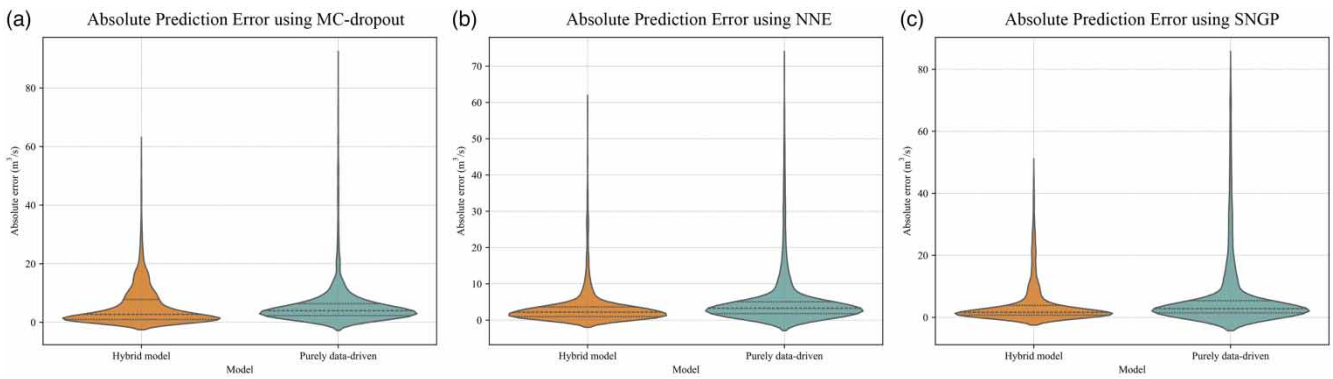

Figure 7 | Violin plots of absolute error between models and observation discharge using MC-dropout (a), NNE (b), and SNGP (c).

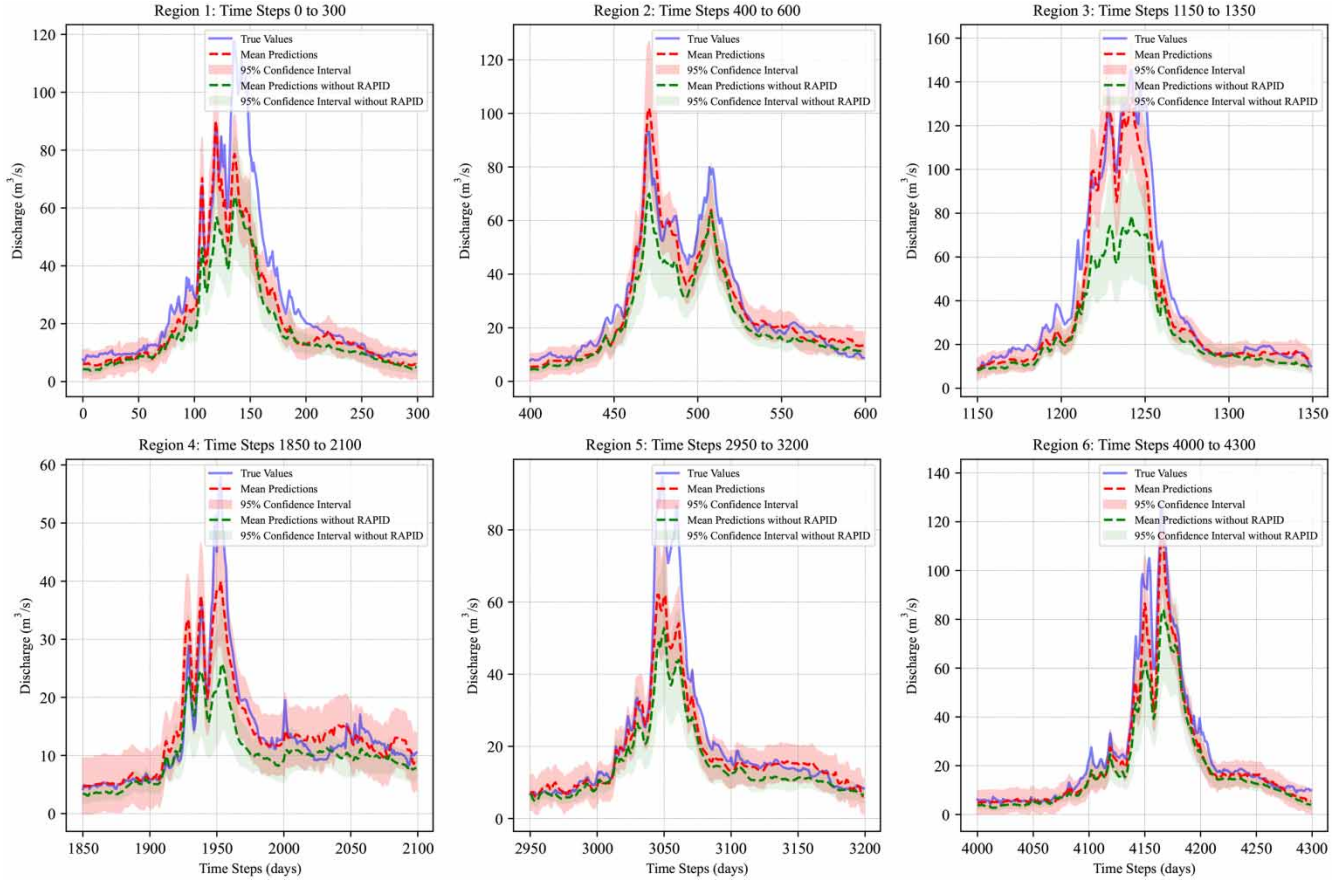

Figure 8 | Divided time-series prediction plot with 95% confidence interval and true values (NNE) between the purely data-driven and hybrid model.

Figure 7(c) presents the violin plot that illustrates the absolute errors between the models' predictions and observed discharge values using SNGP. Notably, the right plot reveals that the purely data-driven model has a wider error distribution, suggesting less reliability and higher average errors, including more high-value errors. Overall, these findings demonstrate that the hybrid model using SNGP outperforms the purely data-driven model, highlighting the benefits of a hybrid approach for prediction.

6.3. Comparison of uncertainty quantification quality using calibration curves

Figure 10(a) shows the calibration curves of MC-dropout for both the hybrid model and the purely data-driven model. The range of each calibration curve reflects the run-to-run variations of MC-dropout, achieved by repeating the algorithm multiple times. The results indicate that the model is conservative in its predictions, and both models exhibit underconfidence in predicting streamflow, meaning the certainty expressed by the models about their predictions is less than what is justified by their actual accuracy or performance. It is important to note that while this finding highlights that the uncertainty is higher for the purely data-driven model compared to the hybrid model, it does not directly represent the actual prediction accuracy. Figure 10(b) presents the calibration curves of NNE for both the hybrid model and the purely data-driven model. The results indicate that both models exhibit underconfidence in predicting streamflow, meaning the certainty expressed by the models about their predictions is less than what is justified by their actual accuracy or performance. Figure 10(c) depicts the calibration curves of SNGP for both the hybrid model and the purely data-driven model. Different from the previous two UQ methods, the calibration curves are closely aligned with the ideal line, indicating that both models generally make well-calibrated predictions. Specifically, the hybrid model

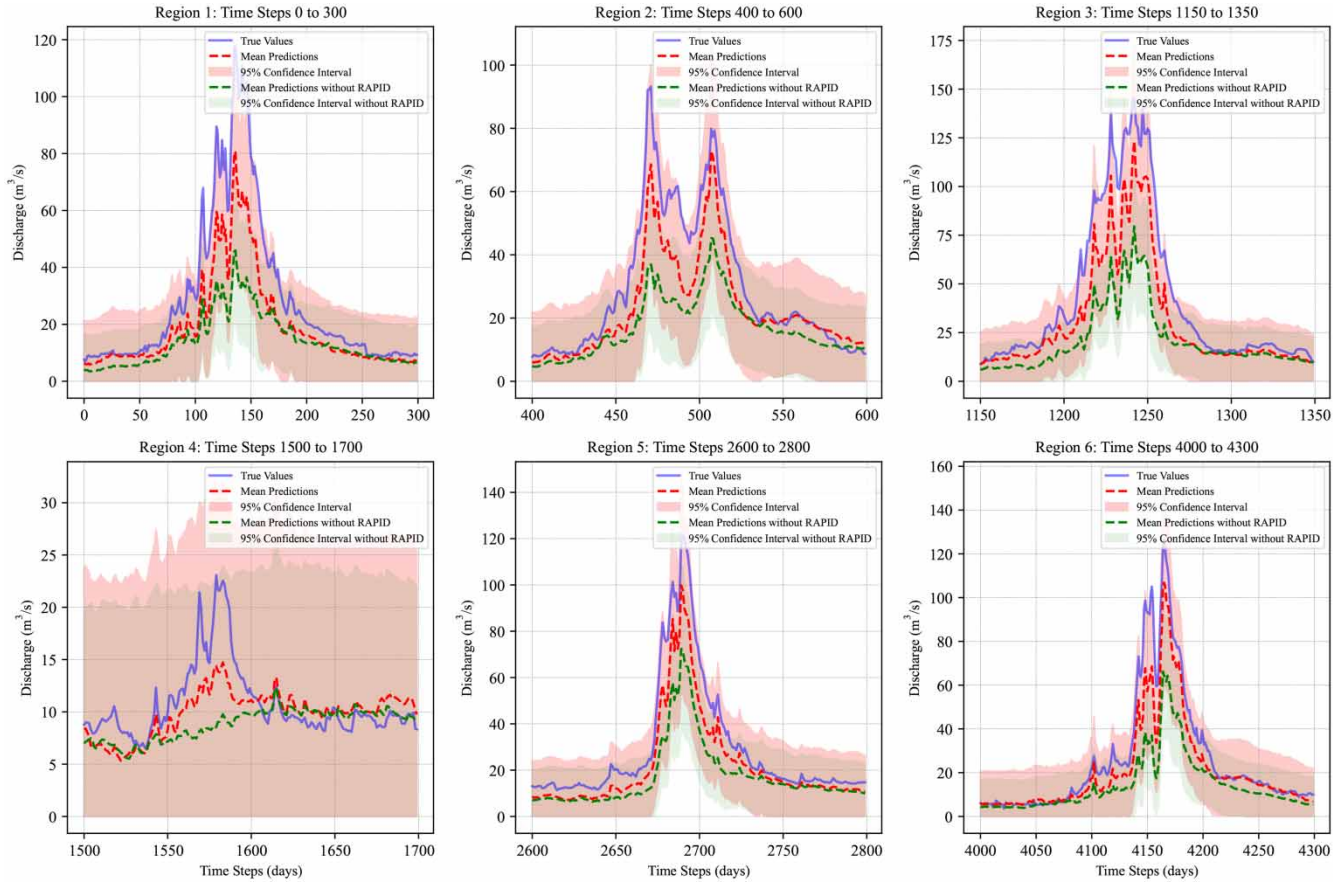

Figure 9 | Divided time-series prediction plot with 95% confidence interval and true values (SNGP) between purely data-driven and hybrid model.

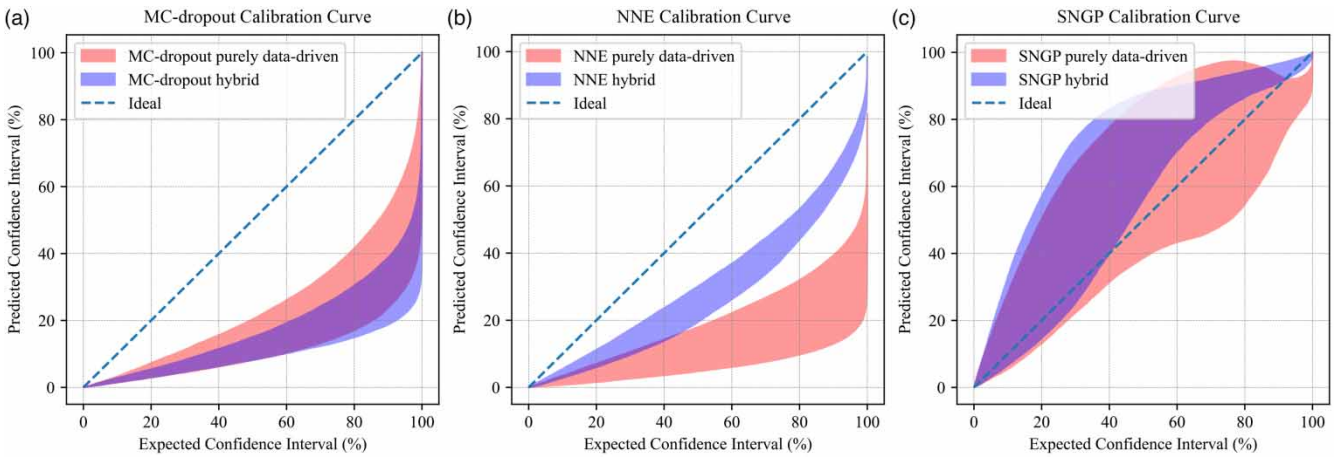

Figure 10 | Calibration curve comparisons between hybrid model and purely data-driven model using (a) MC-dropout, (b) NNE, and (c) SNGP.

exhibits slight overconfidence, although it remains near the ideal line. In contrast, the purely data-driven model closely follows the ideal line throughout, suggesting accurate calibration between predicted probabilities and observed outcomes.

6.4. Comparison of computational time

In this subsection, we compare the computational time required by different methods. All the computational costs are based on an Apple M3 Pro GPU, leveraging PyTorch's Metal Performance Shaders (MPS) backend for optimized computation on macOS. The computational costs required by different methods are summarized as follows.

- MC Dropout: Since the model is trained repeatedly for several times in MC dropout, the training process is notably time-intensive. During the training, the model operates at an average speed of 95.20 iterations per second. With 6,597 batches per epoch, each epoch is completed in approximately 69.30 seconds. Each MC-dropout model takes an average of 23.10 hours to train. For prediction, each model requires an average of 27.5 minutes to complete.
- Neural Network Ensemble: Considering all parameters, the model achieves an average training speed of 104.10 iterations per second. With 6,597 batches per epoch, each epoch is completed in approximately 63.37 seconds. Each NNE model requires approximately 14.08 hours to train. During the prediction phase, each model takes an average of 33 minutes to predict the streamflow for a single river location.
- SNGP: The SNGP training process is significantly faster compared to the LSTM-based models (i.e., MC Dropout and NNE). Each epoch requires only 6.70 seconds on average, and training a single model takes approximately 1.86 hours. The SNGP model is also notably faster during the prediction phase compared to LSTM-based models. On average, it takes just 33 seconds for each SNGP model to make predictions for a single river location.

7. CONCLUSIONS

This paper addresses the critical challenge of predicting river discharge in ungauged locations, with the goal of enhancing the accuracy and reliability of streamflow predictions. The absence of gauged data for these rivers presents a significant obstacle. To address the issue of high dimensionality, this study employs SVD to reduce the number of dimensions. However, using a purely SVD-based approach for all variables carries the risk of excluding important hydrological variables from the model. To mitigate this risk, we adopt a two-stage approach that integrates SVD with a feature selection method. In the first stage, SVD is applied exclusively to the runoff data to reduce its dimensionality. Subsequently, a feature selection method is used to identify the most important features for constructing a machine learning model for streamflow prediction. By leveraging physics-based models like RAPID and incorporating carefully selected features along with discharge data from neighboring gauged rivers, this study demonstrates that a hybrid model, built on ML algorithms, can provide accurate predictions with uncertainty bounds in such scenarios.

The primary objectives of this paper are twofold: (1) to develop a hybrid model trained on selected features, runoff data, and physics-based model outputs as inputs, with discharge data from neighboring gauged rivers as outputs. This approach enables the model to capture both the river dynamics described by physics-based models and the nonlinear relationships between river dynamics and environmental factors. Additionally, training on data from multiple gauged rivers improves the model's ability to generalize across diverse conditions; and (2) to integrate multiple UQ methods that offer probabilistic predictions of discharge in ungauged rivers. Given the complexity of river dynamics and the noise-ridden nature of hydrological data, quantifying uncertainty is crucial for enabling risk-informed decision-making. Our findings highlight the potential of hybrid modeling approaches and UQ methods in tackling the challenges of hydrological predictions, particularly for ungauged rivers. While effective in the presented case studies, the approach depends on historical data and pre-trained models like RAPID, which assume homogeneity and transferability of hydrological behaviors – assumptions that may not hold universally across all regions and climates. Additionally, hybrid modeling and UQ methods can be computationally demanding, with their performance reliant on the quality and availability of hydrological data, which may be sparse or unreliable in certain areas.

Future research directions could focus on expanding the geographic scope of these methods, incorporating additional hydrological and meteorological variables, improving computational efficiency, exploring real-time prediction capabilities, and enhancing data quality through initiatives such as remote sensing techniques. In addition, the proposed method is demonstrated using a catchment in Colorado. Testing the generalization of the proposed method using catchments beyond this study area is worth studying in the future. In conclusion, this study represents a significant advancement in predicting river discharge in ungauged locations. However, addressing current limitations and continuing to refine these advanced modeling techniques will be essential for unlocking their full potential.

ACKNOWLEDGEMENTS

This work was supported by the United States Army Corps of Engineers through the U.S. Army Engineer Research and Development Center Research Cooperative Agreement W912HZ24C0044. The support is gratefully acknowledged.

DATA AVAILABILITY STATEMENT

All relevant data are included in the paper or its Supplementary Information.

CONFLICT OF INTEREST

The authors declare there is no conflict.

REFERENCES

Ajami, N. K., Duan, Q. & Sorooshian, S. (2007) An integrated hydrologic Bayesian multimodel combination framework: Confronting input, parameter, and model structural uncertainty in hydrologic prediction, *Water Resources Research*, **43** (1), W01403.

Bartlett, P. L., Evans, S. N. & Long, P. M (2018) Representing smooth functions as compositions of near-identity functions with implications for deep network optimization, arXiv preprint arXiv:1804.05012.

Chiew, F., Zhou, S. & McMahon, T. (2003) Use of seasonal streamflow forecasts in water resources management, *Journal of Hydrology*, **270** (1-2), 135–144.

Clark, M. P., Wilby, R. L., Gutmann, E. D., Vano, J. A., Gangopadhyay, S., Wood, A. W., Fowler, H. J., Prudhomme, C., Arnold, J. R. & Brekke, L. D. (2016) Characterizing uncertainty of the hydrologic impacts of climate change, *Current Climate Change Reports*, **2** (2), 55–64.

Cunge, J. A. (1969) On the subject of a flood propagation computation method (Muskingum method), *Journal of Hydraulic Research*, **7**, 205–230.

David, C. H., Habets, F., Maidment, D. R. & Yang, Z. -L. (2011) RAPID applied to the SIM-France model, *Hydrological Processes*, **25** (22), 3412–3425.

David, C. H., Maidment, D. R., Niu, G. -Y., Yang, Z. -L., Habets, F. & Eijkhout, V. (2011) River network routing on the NHDPlus dataset, *Journal of Hydrometeorology*, **12** (5), 913–934.

Difi, S., Heddam, S., Zerouali, B., Kim, S., Elmeddahi, Y., Bailek, N., Augusto Guimarães Santos, C. & Abida, H. (2024) Improved daily streamflow forecasting for semi-arid environments using hybrid machine learning and multi-scale analysis techniques, *Journal of Hydroinformatics*, **26** (12), 3266–3286.

Feng, D., Fang, K. & Shen, C. (2020) Enhancing streamflow forecast and extracting insights using long-short term memory networks with data integration at continental scales, *Water Resources Research*, **56** (9), e2019WR026793.

Gal, Y. & Ghahramani, Z (2016) Dropout as a Bayesian approximation: representing model uncertainty in deep learning. In: Balcan, M. F. & Weinberger, K. Q. (eds.) *Proceedings of the 33rd International Conference on Machine Learning, Proceedings of Machine Learning Research*, Vol. 48. New York, New York, USA: PMLR, pp. 1050–1059.

Gupta, A. & Govindaraju, R. S. (2023) Uncertainty quantification in watershed hydrology: which method to use? *Journal of Hydrology*, **616**, 128749.

He, K., Zhang, X., Ren, S. & Sun, J (2016) Deep residual learning for image recognition, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Hochreiter, S. & Schmidhuber, J. (1997) Long short-term memory, *Neural Computation*, **9** (8), 1735–1780.

Hu, Z., Ao, D. & Mahadevan, S. (2017) Calibration experimental design considering field response and model uncertainty, *Computer Methods in Applied Mechanics and Engineering*, **318**, 92–119.

Kasiviswanathan, K. S. & Sudheer, K. P. (2013) Quantification of the predictive uncertainty of artificial neural network based river flow forecast models, *Stochastic Environmental Research and Risk Assessment*, **27** (1), 137–146.

Khosravi, A., Nahavandi, S., Creighton, D. & Atiya, A. F. (2011) Comprehensive review of neural Network-based prediction intervals and new advances, *IEEE Transactions on Neural Networks*, **22** (9), 1341–1356.

Kilinc, H. C. & Haznedar, B. (2022) A hybrid model for streamflow forecasting in the basin of euphrates, *Water*, **14** (1), 80.

Klotz, D., Kratzert, F., Gauch, M., Keefe Sampson, A., Brandstetter, J., Klambauer, G., Hochreiter, S. & Nearing, G. (2022) Uncertainty estimation with deep learning for rainfall–runoff modeling, *Hydrology and Earth System Sciences*, **26** (6), 1673–1693.

Kratzert, F., Klotz, D., Brenner, C., Schulz, K. & Herrnegger, M. (2018) Rainfall–runoff modelling using long short-term memory (LSTM) networks, *Hydrology and Earth System Sciences*, **22** (11), 6005–6022.

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

Li, C. & Mahadevan, S. (2016) An efficient modularized sample-based method to estimate the first-order Sobol' index, *Reliability Engineering & System Safety*, **153**, 110–121.

Li, D., Marshall, L., Liang, Z., Sharma, A. & Zhou, Y. (2021) Bayesian LSTM with stochastic variational inference for estimating model uncertainty in process-based hydrological models, *Water Resources Research*, **57** (9), e2021WR029772.

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J. & Liu, H. (2017) Feature selection: a data perspective, *ACM Computing Surveys*, **50** (6), 1–45.

Lin, Y., Wang, D., Wang, G., Qiu, J., Long, K., Du, Y., Xie, H., Wei, Z., Shangguan, W. & Dai, Y. (2021) A hybrid deep learning algorithm and its application to streamflow prediction, *Journal of Hydrology*, **601**, 126636.

Liu, J., Lin, Z., Padhy, S., Tran, D., Bedrax Weiss, T. & Lakshminarayanan, B. (2020) Simple and principled uncertainty estimation with deterministic deep learning via distance awareness, *Advances in Neural Information Processing Systems*, **33**, 7498–7512.

Liu, S., Lu, D., Painter, S. L., Griffiths, N. A. & Pierce, E. M. (2023) Uncertainty quantification of machine learning models to improve streamflow prediction under changing climate and environmental conditions, *Frontiers in Water*, **5**, 1150126.

Lu, D., Konapala, G., Painter, S. L., Kao, S.-C. & Gangrade, S. (2021) Streamflow simulation in data-scarce basins using Bayesian and physics-informed machine learning models, *Journal of Hydrometeorology*, **22** (6), 1421–1438.

Magni, M., Sutanudjaja, E. H., Shen, Y. & Karssenberg, D. (2023) Global streamflow modelling using process-informed machine learning, *Journal of Hydroinformatics*, **25** (5), 1648–1666.

McCarthy, G. T. (1939) *The Unit Hydrograph and Flood Routing*. Providence: Army Engineer District.

Nearing, G., Cohen, D., Dube, V., Gauch, M., Gilon, O., Harrigan, S., Hassidim, A., Klotz, D., Kratzert, F., Metzger, A., Nevo, S., Pappenberger, F., Prudhomme, C., Shalev, G., Shenzis, S., Tekalign, T. Y., Weitzner, D. & Matias, Y. (2024) Global prediction of extreme floods in ungauged watersheds, *Nature*, **627** (8004), 559–563.

Nearing, G. S., Kratzert, F., Sampson, A. K., Pelissier, C. S., Klotz, D., Frame, J. M., Prieto, C. & Gupta, H. V. (2021) What role does hydrological science play in the age of machine learning? *Water Resources Research*, **57** (3), e2020WR028091.

Nemani, V., Biggio, L., Huan, X., Hu, Z., Fink, O., Tran, A., Wang, Y., Zhang, X. & Hu, C. (2023) Uncertainty quantification in machine learning for engineering design and health prognostics: a tutorial, *Mechanical Systems and Signal Processing*, **205**, 110796.

Ni, L., Wang, D., Singh, V. P., Wu, J., Wang, Y., Tao, Y. & Zhang, J. (2020) Streamflow and rainfall forecasting by two long short-term memory-based models, *Journal of Hydrology*, **583**, 124296.

Niu, G.-Y., Yang, Z.-L., Mitchell, K. E., Chen, F., Ek, M. B., Barlage, M., Kumar, A., Manning, K., Niyogi, D., Rosero, E., Tewari, M. & Xia, Y. (2011) The community Noah land surface model with multiparameterization options (Noah-MP): 1. Model description and evaluation with local-scale measurements, *Journal of Geophysical Research*, **116** (D12), D12109.

Pokharel, S. & Roy, T. (2024) A parsimonious setup for streamflow forecasting using CNN-LSTM, *Journal of Hydroinformatics*, **26** (11), 2751–2761.

Qin, C., Zeng, Y., Zhao, Y., Gugaratshan, G. & Hu, Z (2024a) Recalibration of neural networks using transfer learning for streamflow forecasting, *Volume 3A: 50th design automation conference (DAC)*, American Society of Mechanical Engineers, Paper No. V03AT03A038.

Qin, C., Zeng, Y., Zhao, Y., Gugaratshan, G. & Hu, Z (2024b) Recalibration of neural networks using transfer learning for streamflow forecasting, *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Vol. 88360. American Society of Mechanical Engineers, p. V03AT03A038.

Rahimzad, M., Moghaddam Nia, A., Zolfonoon, H., Soltani, J., Danandeh Mehr, A. & Kwon, H.-H. (2021) Performance comparison of an LSTM-based deep learning model versus conventional machine learning algorithms for streamflow forecasting, *Water Resources Management*, **35** (12), 4167–4187.

Renard, B., Kavetski, D., Kuczera, G., Thyer, M. & Franks, S. W. (2010) Understanding predictive uncertainty in hydrologic modeling: the challenge of identifying input and structural errors, *Water Resources Research*, **46** (5), W05521.

Sattari, A., Foroumandi, E., Gavahi, K. & Moradkhani, H. (2025) A probabilistic machine learning framework for daily extreme events forecasting, *Expert Systems with Applications*, **265**, 126004.

Schutte, C., van der Laan, M. & van der Merwe, B. (2024) Leveraging historic streamflow and weather data with deep learning for enhanced streamflow predictions, *Journal of Hydroinformatics*, **26** (4), 835–852.

Shamshirband, S., Hashemi, S., Salimi, H., Samadianfard, S., Asadi, E., Shadkani, S., Kargar, K., Mosavi, A., Nabipour, N. & Chau, K. W. (2020) Predicting standardized streamflow index for hydrological drought using machine learning models, *Engineering Applications of Computational Fluid Mechanics*, **14** (1), 339–350.

Syed, Z., Mahmood, P., Haider, S., Ahmad, S., Jadoon, K. Z., Farooq, R., Syed, S. & Ahmad, K. (2023) Short–long-term streamflow forecasting using a coupled wavelet transform–artificial neural network (WT–ANN) model at the Gilgit River Basin, Pakistan, *Journal of Hydroinformatics*, **25** (3), 881–894.

Tan, W. Y., Lai, S. H., Pavitra, K., Teo, F. Y. & El-Shafie, A. (2023) Deep learning model on rates of change for multi-step ahead streamflow forecasting, *Journal of Hydroinformatics*, **25** (5), 1667–1689.

Wang, Y., Zhang, T., Guo, X. & Shen, Z (2024) Gradient based feature attribution in explainable AI: a technical review, arXiv preprint arXiv:2403.10415.

Xu, T. & Liang, F. (2021) Machine learning for hydrologic sciences: an introductory overview, *WIREs Water*, **8** (5), e1533.

Zadrozny, B. & Elkan, C. (2002) Transforming classifier scores into accurate multiclass probability estimates, *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, pp. 694–699.

Zhao, T., Wang, Q., Bennett, J. C., Robertson, D. E., Shao, Q. & Zhao, J. (2015) Quantifying predictive uncertainty of streamflow forecasts based on a Bayesian joint probability model, *Journal of Hydrology*, **528**, 329–340.

Zhao, Y., Chadha, M., Barthlow, D., Yeates, E., Mcknight, C. J., Memarsadeghi, N. P., Gugaratshan, G., Todd, M. D. & Hu, Z. (2024) Physics-enhanced machine learning models for streamflow discharge forecasting, *Journal of Hydroinformatics*, **26** (10), 2506–2537.

Zhao, Y., Chadha, M., Olsen, N., Yeates, E., Turner, J., Gugaratshan, G., Qian, G., Todd, M. D. & Hu, Z. (2023) Machine learning-enabled calibration of river routing model parameters, *Journal of Hydroinformatics*, **25** (5), 1799–1821.

Zhong, M., Zhang, H., Jiang, T., Guo, J., Zhu, J., Wang, D. & Chen, X. (2023) A hybrid model combining the cama-flood model and deep learning methods for streamflow prediction, *Water Resources Management*, **37** (12), 4841–4859.

First received 8 February 2025; accepted in revised form 8 August 2025. Available online 26 August 2025