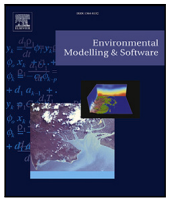
Environmental Modelling and Software 169 (2023) 105831

Contents lists available at ScienceDirect

Environmental Modelling and Software

journal homepage: www.elsevier.com/locate/envsoft



Position Paper

DeepGR4J: A deep learning hybridization approach for conceptual rainfall-runoff modelling

Arpit Kapoor [a,c,d,*], Sahani Pathiraja [a,c,d], Lucy Marshall [c,b], Rohitash Chandra [a,c,d]

[a] School Mathematics and Statistics, University of New South Wales, Sydney, 2052, NSW, Australia
[b] Faculty of Science and Engineering, Macquarie University, Sydney, 2109, NSW, Australia
[c] Data Analytics for Resources and Environments, Australian Research Council—Industrial Transformation Training Centre, Sydney, NSW, Australia
[d] UNSW Data Science Hub, University of New South Wales, Sydney, 2052, NSW, Australia

ARTICLE INFO

Dataset link: https://github.com/arpit-kapoor/hybrid-gr4j

Keywords:
DeepGR4J
GR4J
Rainfall-runoff modelling
Deep learning
Hybrid modelling
Convolutional neural networks
Long short-term memory

ABSTRACT

Despite the considerable success of deep learning methods in modelling physical processes, they suffer from a variety of issues such as overfitting and lack of interpretability. In hydrology, conceptual rainfall-runoff models are simple yet fast and effective tools to represent the underlying physical processes through lumped storage components. Although conceptual rainfall-runoff models play a vital role in supporting decision-making in water resources management and urban planning, they have limited flexibility to take data into account for the development of robust region-wide models. The combination of deep learning and conceptual models has the potential to address some of the aforementioned limitations. This paper presents a sub-model hybridization of the GR4J rainfall-runoff model with deep learning architectures such as convolutional neural networks (CNN) and long short-term memory (LSTM) networks. The results show that the hybrid models outperform both the base conceptual model as well as the canonical deep neural network architectures in terms of the Nash–Sutcliffe Efficiency (NSE) score across 223 catchments in Australia. We show that our hybrid model provides a significant improvement in predictive performance, particularly in arid catchments, and generalizing better across catchments.

1. Introduction

Rainfall-runoff modelling has a significant impact on governing water management practices. In past years, several approaches have been developed to improve the accuracy of modelling streamflow which represents core hydrological processes associated with extreme events such as flooding (Devia et al., 2015; Solomatine and Wagener, 2011). Traditionally, these approaches have relied on conceptual models that represent the underlying physical processes using simplified storage components as the main building blocks (Burnash, 1995; Burnash and Ferral, 1973; Perrin et al., 2007; Boughton, 2004; Bergström, 1976). The interconnected storage components represent non-spatially explicit physical elements in a catchment that are recharged by rainfall, infiltration and percolation while being discharged by evaporation, runoff, drainage, etc. Discretized semi-empirical equations that satisfy the physical laws such as the conservation of mass and energy are at the core foundation of rainfall-runoff models. Based on the complexity of the model, a varying number of storage parameters such as the maximum capacity of storage, initial storage value, and rate of moisture exchange are calibrated using a large volume of meteorological/hydrological data. The *Australian Water Balance Model* (AWBM) (Boughton, 2004), *Génie Rural à 4 paramètres Journalier* model (GR4J) (Perrin et al., 2003, 2007), and *Sacramento* (Burnash, 1995) are a few examples of such models that have shown promising results in streamflow modelling (Jehanzaib et al., 2022; Jaiswal et al., 2020). Rainfall-runoff models play a vital role in planning water resources for cities that are affected by the adverse effects of climate change (Hatmoko et al., 2020).

Alternatively, physically-based hydrologic models rely on a mathematically idealized representation of underlying physical processes (Abbott et al., 1986; Beven, 1989, 2002; Paniconi and Putti, 2015). Physically-based models refer to distributed or semi-distributed hydrologic models which differ from conceptual models in that they rely on physical equations that model the spatial characteristics of fluid flow within a catchment. The hydrological processes are represented by a system of partial or ordinary differential equations of measurable state variables as a function of both time and space. In these

* Corresponding author at: School Mathematics and Statistics, University of New South Wales, Sydney, 2052, NSW, Australia.
E-mail addresses: arpit.kapoor@unsw.edu.au (A. Kapoor), sahani.pathiraja@unsw.edu.au (S. Pathiraja), lucy.marshall@mq.edu.au (L. Marshall), rohitash.chandra@unsw.edu.au (R. Chandra).
URL: http://arpit-kapoor.com (A. Kapoor).

https://doi.org/10.1016/j.envsoft.2023.105831
Received 20 June 2023; Received in revised form 7 September 2023; Accepted 14 September 2023
Available online 18 September 2023

models, the parameters that describe the physical characteristics of the catchment are evaluated, as opposed to data-driven calibration in the case of conceptual models (Abbott et al., 1986). Due to their physical interpretation of the hydrologic processes, these models do not require large volumes of meteorological or climate data to calibrate. Thus, they overcome some limitations of the other modelling approaches. However, physically-based models suffer from scale-related problems, e.g. measurement scales typically differ from the simulation model scales, and from over-parameterization (Beven, 1989). Furthermore, given the distributed nature of these models, they are difficult to calibrate and are often outperformed by conceptual models (Jaiswal et al., 2020).

Alongside physics-based and conceptual hydrologic models, data-driven models that are based purely on empirical observations of streamflow have grown in popularity. The primary contenders in this space have been statistical time-series modelling techniques such as autoregressive moving average (ARIMA) models (Valipour, 2015; Mishra et al., 2018; Pektaş and Kerem Cigizoglu, 2013), multilinear regression models (Kisi et al., 2013; Adnan et al., 2021), and neural networks (Tokar and Johnson, 1999; Dawson and Wilby, 1998; Cigizoglu and Alp, 2004; Srinivasulu and Jain, 2006; Maniquiz et al., 2010; Kisi et al., 2013). Such mathematical models are typically physics-agnostic and only use concurrent input–output samples to supervise the model learning. Due to their flexible nature and the ability to learn latent patterns in large volumes of hydrological and meteorological data, machine learning models have been shown to outperform traditional modelling approaches in terms of accuracy even in ungauged basins where no measured runoff data is available (Adnan et al., 2021). These models are highly versatile (Reichstein et al., 2019); hence, the same models can be applied to a variety of modelling tasks using transfer learning for data-scarce regions (Ma et al., 2021), and flood forecasting (Feng et al., 2020; Nevo et al., 2022). Furthermore, machine learning models add the ability to model streamflow as a function of historical observations (Sezen and Partal, 2018) and capture data noise which physically-based models lack. Hence, machine learning models have the ability to leverage additional climate variables as inputs for improved predictive accuracy. Deep Learning models, especially recurrent neural networks (RNN) such as long short-term memory (LSTM) neural networks, have shown significant potential for rainfall-runoff modelling (Kratzert et al., 2018; Feng et al., 2020; Xiang et al., 2020; Lees et al., 2021).

Although deep learning models demonstrate high predictive accuracy in modelling physical processes when sufficient data is available, these models are physics-agnostic and lack interpretability (Samek et al., 2019). In complex systems, as in the case of hydrologic processes, the ability to understand and interpret the underlying interrelations of the processes is of importance to researchers and policymakers who rely on the predictions from these models. Hence, attempts have been made to improve the interpretability/explainability of machine learning models that have led to recent advancements in the field of explainable artificial intelligence (XAI), also known as interpretable machine learning (Montavon et al., 2018; Samek et al., 2019; Tjoa and Guan, 2021; Holzinger et al., 2022). In the case of deep learning models, XAI can highlight the correlations between the dynamics of the hidden states of the models that can provide connections to the underlying hydrologic process (Lees et al., 2022). However, interpreting machine learning models requires the use of these external frameworks that have their own caveats and assumptions and may not provide domain-specific interpretation capabilities. Conceptual models, on the other hand, provide more straightforward interpretability by incorporating the hydrologic processes as part of the model structure. The associated ease of interpretability of these models makes the incorporation of physics into machine learning desirable.

In recent years, environmental research communities have developed hybrid modelling frameworks where conceptual/physics-based models have been combined with data-driven (machine learning) models to improve the predictive capabilities while retaining interpretability (Bézenac et al., 2019; Reichstein et al., 2019; Razavi et al., 2022). Hybrid modelling approaches have shown a lot of promise in improving the predictive capability of conceptual/physics-based models without losing interpretability (Reichstein et al., 2019). Primarily, four hybridization approaches for conceptual/physically-based hydrologic models have been presented in the literature: (1.) parameterization using machine learning (Beck et al., 2016), (2.) machine learning for modelling prediction errors in physics-based/conceptual models (Vandal et al., 2018), (3.) machine learning for replacing sub-processes in a physically-based/conceptual model (Bézenac et al., 2019); and finally, (4.) machine learning model as a surrogate for the physically-based/conceptual model (Camps-Valls et al., 2018; Chevallier et al., 1998). The strength of hybrid modelling motivates the inclusion of novel conceptual and machine learning models into hybrid modelling frameworks. In the past, a variety of optimization approaches have been used to calibrate conceptual catchment models. These approaches have primarily relied on evolutionary optimization methods such as genetic algorithms for model calibration (Guo et al., 2013; Khu et al., 2008; Gill et al., 2006; Jiang et al., 2010). Other evolutionary optimization methods such as differential evolution and particle swarm optimization have shown promise (Liu, 2009; Napiorkowski et al., 2022; Wu et al., 2011; Franchini, 1996; Wang, 1991). Therefore, several hybridization approaches (eg. Okkan et al., 2021; Ersoy et al., 2022) used evolutionary optimization for the parameter estimation of both conceptual as well as machine learning models. Although evolutionary optimization algorithms are suitable for the calibration of hydrologic models with only a few to several parameters, gradient-based optimization via backpropagation has proven to be the most suitable approach for high-dimensional optimization problems such as in the case of deep learning models (Ruder, 2016). Additionally, gradient-based optimization has been successfully applied to conceptual hydrologic models (Krapu et al., 2019).

In this paper, we present a hybrid model framework where selected deep learning models are used to simulate sub-processes in a conceptual hydrologic model. We consider deep learning models such as CNNs and LSTM networks for modelling streamflow time series. We compare different hybrid models with conceptual models and data-driven models. We also present a hierarchical optimization framework that combines evolutionary optimization for the hydrologic model with gradient-based optimization for the deep learning models. Finally, we also investigate the impact of including streamflow observations from previous time steps in the model input. We release open-source code for further studies.

The rest of the paper is organized as follows. Section 2 presents a review of related work on hybrid rainfall-runoff modelling. Section 3 presents background on canonical hydrological and deep learning methods that compose the proposed methodology. Section 4 presents the proposed DeepGR4J hybrid model architecture with details of the proposed model training scheme. Section 5 presents the experiment design and results, Section 6 discusses the results and Section 7 concludes with future research directions.

2. Related work

Several hybridization approaches have also been extended to rainfall-runoff modelling. Tian et al. (2018) compared the performance of three RNNs for rainfall-runoff simulation tasks that include Elman RNNs, echo state networks, LSTM networks, and nonlinear autoregressive networks with exogenous inputs (NARX). The authors used the two best-performing models to model the prediction errors in the GR4J hydrologic model. The hybrid models performed the best with a significant reduction in prediction error of low flows. Similarly, Li et al. (2021b) successfully applied a similar approach to the distributed hydrologic model (MIKE-SHE), where probabilistic LSTM

Table 1
Parameter description of GR4J model.

Parameter	Description	Unit
X_1	production store maximal capacity	mm
X_2	catchment water exchange coefficient	mm/day
X_3	one-day maximal capacity of the routing reservoir	mm
X_4	unit hydrograph time base	days

networks were used to model residuals (errors) of the hydrological predictions. They later extended this framework using variational inference for uncertainty quantification of model parameters (Li et al., 2021a). Furthermore, Okkan et al. (2021) proposes a nested hybrid rainfall-runoff model with simultaneous calibration of a conceptual model, namely the *dynamic water balance model* (dynwbm) and machine learning models. The study found that the nested framework outperformed the standalone models and coupled frameworks. Ersoy et al. (2022) embedded simple neural networks as a sub-model into the *dynwbm conceptual model* and reported that the hybrid model (dynNN) gave the best performance while disregarding the drawbacks of base models. Mohammadi et al. (2022) proposed a multi-conceptual and machine-learning model hybridization approach for modelling the rainfall-runoff process in a snow-covered basin in Switzerland. In this approach, three conceptual models, i.e., IHACRES (*Identification of unit Hydrographs And Component flows from Rainfall, Evaporation and Streamflow*), GR4J, and MISD (*Modello Idrologico Semi-Distribuito*) were coupled with machine learning algorithms such as support vector machines (SVM) and multilayer perception (MLP) that used predictions from conceptual models along with different meteorological features as input. Finally, data fusion and Whale Optimization Algorithm (WOA) have been used to improve predictions via the MLP-WOA model. Höge et al. (2022) proposed hydrologic neural ordinary differential equation (ODE) models for retaining the high accuracy of data-driven deep learning models while maintaining the interpretability of conceptual model (EXP-Hydro) in streamflow prediction tasks.

Our work extends the sub-model approach hybridization approach (Ersoy et al., 2022) to the GR4J rainfall-runoff model with LSTM and CNN neural network architecture. Although a similar approach has been explored in the literature (Kumanlioglu and Fistikoglu, 2019), these approaches only consider simple feedforward neural networks. Furthermore, these approaches rely on evolutionary/swarm optimization approaches more suited for simpler architectures, as they can be computationally expensive. We propose a hybrid training approach where we use an evolutionary algorithm for hydrologic model calibration whereas, backpropagation is used for the neural network model.

3. Background for methodology

3.1. GR4J hydrologic model

The GR4J (Perrin et al., 2003, 2007) is a conceptual catchment water balance model that relates catchment runoff to rainfall and evapotranspiration using daily data. The model consists of a production storage and a routing storage/reservoir. The storage accounts for the moisture stored in the catchment that is recharged by precipitation and percolation. Evapotranspiration and runoff are processes that discharge the storage in the catchment. The production storage is representative of the surface moisture while the routing storage accounts for the leakage and soil moisture exchange. GR4J is designed to model hydrological processes in the catchment while maintaining water balance in the storages, thus making it an interpretable model. Fig. 1 highlights the GR4J model architecture and shows the underlying symbolic processes connecting rainfall and runoff in a catchment.

Table 1 provides the details of GR4J parameters to be calibrated using observations. The conceptual model (GR4J) is composed of four subprocesses: the soil moisture subprocess, the effective precipitation

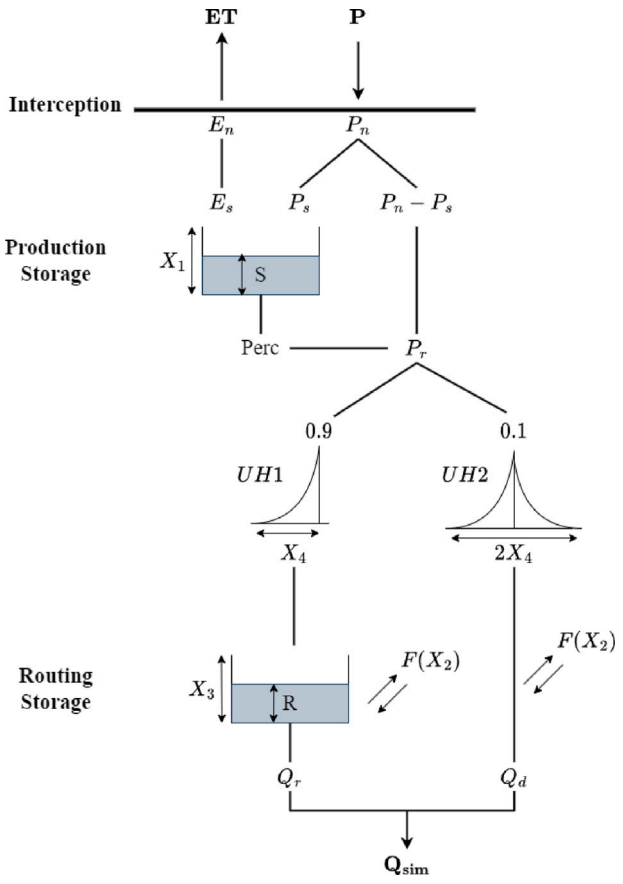


Fig. 1. GR4J hydrologic model architecture.

calculation, the slow flow subprocess, and the quick flow subprocess. The production storage that accounts for the soil moisture takes two inputs: precipitation (P) and potential evapotranspiration (E) at daily timestep t. First, the model computes net precipitation by subtracting the potential evapotranspiration from the precipitation. We follow a similar process for computing the net evapotranspiration:

$$P_n^{(t)} = \max(P^{(t)} - E^{(t)}, 0) \tag{1}$$

$$E_n^{(t)} = \max(E^{(t)} - P^{(t)}, 0) \tag{2}$$

A part of the net precipitation (denoted by P_s) enters the production storage, while the remaining part ($P_n - P_s$) along with the percolation ($Perc$) from the production storage moves to the flow-routing reservoir. The moisture in the production storage at time t is given as

$$S^{(t)} = S^{(t-1)} + P_s^{(t)} - E_s^{(t)} \tag{3}$$

where, $P_s^{(t)}$ and $E_s^{(t)}$ are given as

$$P_s^{(t)} = \frac{X_1\left[1 - \left\{\frac{S^{(t-1)}}{X_1}\right\}^2\right]\tanh\left(\frac{P_n^{(t)}}{X_1}\right)}{1 + \frac{S^{(t-1)}}{X_1}\tanh\left(\frac{P_n^{(t)}}{X_1}\right)} \tag{4}$$

$$E_s^{(t)} = \frac{S^{(t-1)}\left[2 - \frac{S^{(t-1)}}{X_1}\right]\tanh\left(\frac{E_n^{(t)}}{X_1}\right)}{1 + \left[1 - \frac{S^{(t-1)}}{X_1}\right]\tanh\left(\frac{E_n^{(t)}}{X_1}\right)} \tag{5}$$

The flow entering the routing storage consists of two components: leakage percolation from the production storage and remaining net precipitation after P_s is absorbed by the production storage. The leakage percolation ($Perc$) is computed as the power function of the moisture in the storage (S). The percolation and the net precipitation going into

the routing storage (P_r) are then computed as

$$Perc^{(t)} = S^{(t)}\left[1-\left\{1+\left(\frac{4}{9}\frac{S^{(t)}}{X_1}\right)^4\right\}^{-1/4}\right] \quad (6)$$

$$P_r^{(t)} = Perc + P_n^{(t)} - P_s^{(t)} \quad (7)$$

The combined inflow is split into 90% runoff routed by a unit hydrograph ($HU1$) to the nonlinear routing reservoir and 10% runoff routed by a single unit hydrograph ($HU2$) to direct runoff, both of which depend on the X_4 parameter. The two parts of the inflow Q_1 and Q_9 are then computed by

$$Q_9^{(t)} = 0.9 \times \sum_{k=1}^{l} HU1(k) \times P_r^{(t-k+1)} \quad (8)$$

$$Q_1^{(t)} = 0.1 \times \sum_{k=1}^{m} HU2(k) \times P_r^{(t-k+1)} \quad (9)$$

where, the limits l and m are equal to $\lfloor X_4 \rfloor$ and $\lfloor 2X_4 \rfloor$, respectively. The routing storage (represented by R) is then updated after accounting for the groundwater exchange (represented by F) and the runoff from the routing storage (Q_r) is then given by

$$F^{(t)} = X_2\left(\frac{R^{(t-1)}}{X_3}\right)^{7/2} \quad (10)$$

$$R^{(t)} = \max\left(R^{(t-1)} + Q_9^{(t)} + F^{(t)}; 0\right) \quad (11)$$

$$Q_r^{(t)} = R^{(t)}\left[1-\left\{1+\left(\frac{R^{(t)}}{X_3}\right)^4\right\}^{-1/4}\right] \quad (12)$$

Finally, the direct runoff (Q_d) is computed and the total runoff is obtained by adding these two runoffs together as given by

$$Q_d^{(t)} = \max\left(Q_1^{(t)} + F^{(t)}; 0\right) \quad (13)$$

$$Q^{(t)} = Q_r^{(t)} + Q_d^{(t)} \quad (14)$$

where $Q^{(t)}$ is the simulated total streamflow (mm) at the time-step t. Other implementations of the GR4J model have been explored with an additional number of parameters such as GR5J (Le Moine, 2008) and GR6J (Pushpalatha et al., 2011) and so on with 5 and 6 trainable parameters, respectively. We selected GR4J for this study due to its simplicity with a two-storage setup and proven effective performance with just 4 tunable parameters.

3.2. Deep learning models

Deep learning models such as LSTM networks and CNNs have been useful for modelling spatiotemporal data (Wang et al., 2020). They have also shown excellent performance at multi-step ahead univariate time series forecasting problems (Chandra et al., 2021) and hence, are suitable for our study.

3.2.1. Long short-term memory (LSTM)

RNNs model temporal dependencies in discrete time data via recurrent (state) neurons that provide prediction as a function of previous states and current inputs (Elman, 1990). As a result of this property, RNNs have been quite successful in time-series prediction problems in environmental data modelling and hydrology (Kratzert et al., 2018; Feng et al., 2020; Xiang et al., 2020; Lees et al., 2021). However, a major drawback of traditional RNN models has been the problem of *vanishing gradients*, wherein during training, the gradient used for back-propagation decreases given the increasing depth of the network based on recurrent neurons defined by long sequences of data (Hochreiter, 1998). Therefore, vanilla RNN architectures (Elman, 1990) are typically not suitable for modelling long-term dependencies in sequences. LSTM networks (Hochreiter and Schmidhuber, 1997) addressed this problem by using sophisticated memory cells that include gating units. LSTM

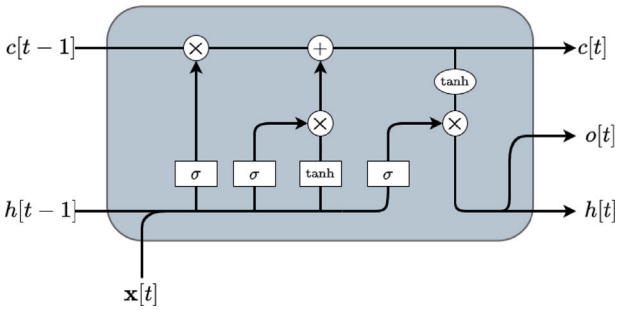


Fig. 2. Architecture of an LSTM cell.

memory cells conventionally consist of an input gate that regulates the flow of input, a cell state that stores the information from previous states in latent space, a forget gate that regulates the dissipation of memory and an output gate that predicts the output states as shown in Fig. 2. The input to the LSTM at time-step t is processed through these gates by:

$$i[t] = \sigma\left(W_{ii}\mathbf{x}[t] + b_{ii} + W_{ih}h[t-1] + b_{ih}\right) \quad (15)$$

$$f[t] = \sigma\left(W_{fi}\mathbf{x}[t] + b_{fi} + W_{fh}h[t-1] + b_{fh}\right) \quad (16)$$

$$o[t] = \sigma\left(W_{oi}\mathbf{x}[t] + b_{oi} + W_{oh}h[t-1] + b_{oh}\right) \quad (17)$$

$$\hat{c}[t] = \tanh\left(W_{oi}\mathbf{x}[t] + b_{oi} + W_{oh}h[t-1] + b_{oh}\right) \quad (18)$$

where $x[t]$ is the current input state and $h[t]$ is the current hidden state which also acts as the output for the LSTM state. Forget, input and output gates are represented by $f[t]$, $i[t]$ and $o[t]$, respectively. The output from these gates is used to update the hidden and the memory states at time-step t as:

$$c[t] = f[t] \odot c[t-1] + i[t] \odot \hat{c}[t] \quad (19)$$

$$h[t] = o[t] \odot \tanh(c[t]) \quad (20)$$

where $\hat{c}[t]$ is referred to as the candidate or memory state and $c[t]$ refers to the cell state (See Fig. 2). The gated recurrent unit (GRU) (Chung et al., 2014) network provides a simpler implementation of the LSTM network that has been reported to save computational time.

3.2.2. Convolutional neural networks

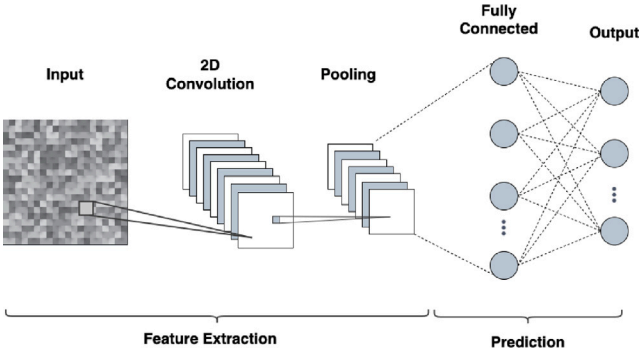


Fig. 3. A generic CNN architecture.

CNNs have been inspired by natural visual systems that were developed by LeCun et al. (1989) for the recognition of handwritten digits. The convolution operation was designed to mimic filter operations common in the traditional image processing literature. CNNs have been immensely successful in several computer vision tasks such as image recognition (Alzubaidi et al., 2021; Pouyanfar et al., 2018). LeCun et al. (1995) later highlighted that CNNs can also be used for other problems such as speech recognition and time series prediction. Chandra et al.

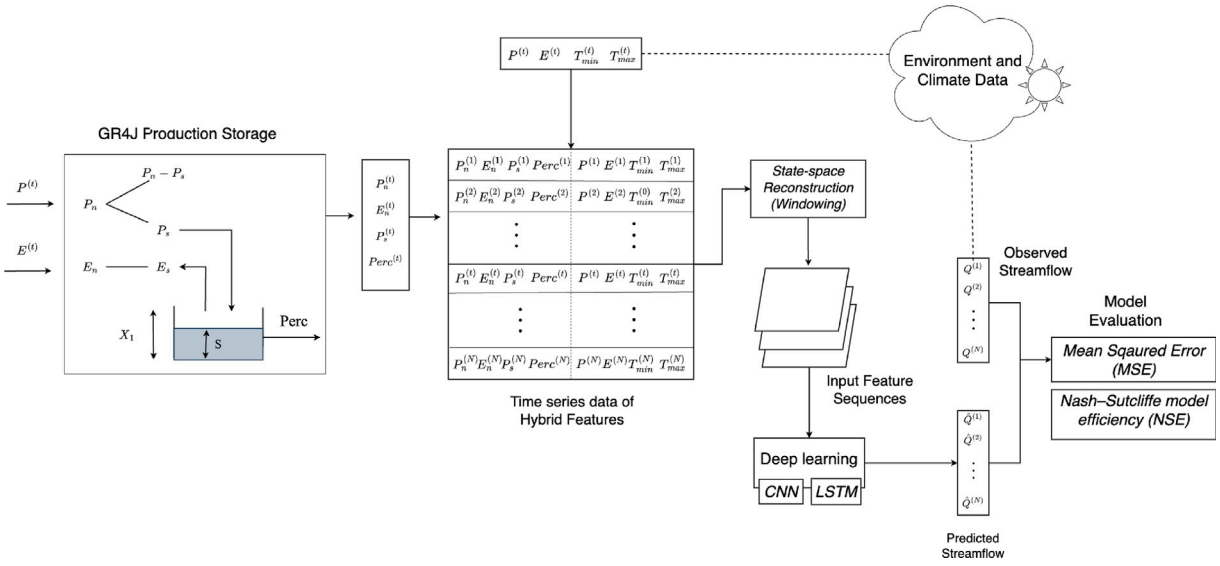


Fig. 4. DeepGR4J hybrid rainfall-runoff model architecture.

(2021) have shown that CNNs provide better accuracy than LSTM models for multi-step time-series forecasting problems.

A conventional CNN model learns the spatial hierarchies of multi-dimensional features in data via a series of convolution and pooling operations that are followed by fully connected layer(s). Fig. 3 shows how a two-dimensional image tensor is processed through different layers in a CNN. Intuitively, a sequence of features (a multivariate time series) can be represented as a two-dimensional tensor (a matrix) of shape $(n \times d)$; where n is the number of input time steps and d is the number of features recorded at each time-step. In a single step-ahead forecasting setting, the sequences are reconstructed (windowed) taking into account Taken's theorem (Takens, 2006) with a window size of α. We represent each training example at time-step t using an input matrix $\mathbf{X_t}$ of shape $(\alpha \times d)$, and a target value as $y_{t+\alpha+1}$. The input matrix at time step t is then written as:

$$\mathbf{X_t} = \begin{bmatrix} x_1^{(t+1)} & x_2^{(t+1)} & \dots & x_d^{(t+1)} \\ x_1^{(t+2)} & x_2^{(t+2)} & \dots & x_d^{(t+2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(t+\alpha)} & x_2^{(t+\alpha)} & \dots & x_d^{(t+\alpha)} \end{bmatrix} \tag{21}$$

A CNN can be used for spatiotemporal modelling using convolution and pooling operations followed by a series of fully connected layers to generate the output $\hat{y}_t$.

4. DeepGR4J: hybrid rainfall-runoff model

We develop the proposed hybrid rainfall-runoff model (DeepGR4J) by integrating the production storage from the GR4J model (highlighted in Section 3.1) with the deep learning models (CNN and LSTM) in Section 3.2. Fig. 4 shows the architecture of the DeepGR4J rainfall-runoff model. In this setting, we only retain one parameter from the canonical GR4J model (X_1) and discard the remaining components. As depicted in Fig. 4, we pass the minimum and maximum average daily surface temperatures (denoted by T_{min} and T_{max}) and vapour pressure (VP) as inputs to the model in addition to the precipitation (P) and potential evapotranspiration (E).

At each time step t, we simulate the net precipitation (P_n), net evapotranspiration (E_n) at interception, net precipitation entering the production storage (P_s) and percolation ($Perc$) using the trained X_1 parameter as given in Eqs. (2)–(6). The simulated features at each time step t are then concatenated with the model inputs into a single feature

Data: Hydrological data for a gauged catchment
Result: Optimum values for X_1 parameter and neural network parameters θ
Stage 1: Initialization
i. Define GR4J model $Q = gr4j(P, E; \delta)$
ii. Initialize GR4J parameters $\delta = \{X_1, X_2, X_3, X_4\}$, window size α and Q_{in} flag
iii. Reconstruct data into input vectors: P and E and target vector Q
iv. Define neural network as $x = g(\tilde{\mathbf{X}}; \theta)$
v. Initialize the neural network parameters, θ
Stage 2: GR4J Calibration
vi. Obtain optimal values of GR4J parameters as, $\hat{\delta}$ using differential evolution
vii. Define production storage with optimized X_1 as $x = prod(P, E; \hat{X}_1)$
Stage 3: Hybrid feature generation
viii. Simulate features from the production storage:
for $t = 1, \dots, T$ **do**
  1. Compute the feature from production storage:
    $\mathbf{x_{prod}^{(t)}} := prod(P^{(t)}, E^{(t)}; \hat{X}_1)$
  2. Concatenate production $\mathbf{x_{prod}^{(t)}}$ with input features to get the input feature vector for the neural network:
  if Q_{in} *is True* **then**
    Compute $\tilde{\mathbf{x}}^{(t)}$ using Equation (24)
  else
    Compute $\tilde{\mathbf{x}}^{(t)}$ using Equation (21)
  end
end
Stage 4: Neural network training
ix. Set neural network hyper-parameters: i.e. training epochs N_{epoch}, learning rate η
x. Train the neural network model: **for** $n = 1, \dots, N_{epoch}$ **do**
  for $t = 1, \dots, T$ **do**
    1. Obtain input and output pair using Equation (21): $(\tilde{\mathbf{X}}^{(t)}, Q^{(t+\alpha+1)})$
    2. Predict streamflow using the neural network,
      $\hat{Q}^{(t+\alpha+1)} := g(\tilde{\mathbf{X}}^{(t)}; \theta)$
    3. Compute loss $\mathcal{L}$
    5. Compute gradients: $\Delta\theta := \frac{\partial \mathcal{L}}{\partial \theta}$
    6. Update parameters:
      $\theta := \theta - \eta\Delta\theta$
  end
end

Algorithm 1: Hierarchical training of DeepGR4J model

vector $\tilde{\mathbf{x}}$. We then reconstruct the data using Taken's theorem with window size α to create sequences for this multivariate time series. We represent the input sequence of the feature vector at time step t, given by a matrix $\tilde{\mathbf{X}}_t$ defined as

$$\tilde{\mathbf{X}}^{(t)} = \begin{bmatrix} \tilde{\mathbf{x}}_{t+1}^T \\ \tilde{\mathbf{x}}_{t+2}^T \\ \vdots \\ \tilde{\mathbf{x}}_{t+\alpha}^T \end{bmatrix} \quad where, \quad \tilde{\mathbf{x}}_t = \begin{bmatrix} P^{(t)} \\ E^{(t)} \\ T_{min}^{(t)} \\ T_{max}^{(t)} \\ P_n^{(t)} \\ E_n^{(t)} \\ P_s^{(t)} \\ Perc^{(t)}. \end{bmatrix} \tag{22}$$

We then pass these sequences as input to the neural network model and the simulated runoff at time-step $(t+\alpha+1)$ denoted by $\hat{Q}^{(t+\alpha+1)}$ and given by

$$\hat{Q}^{(t+\alpha+1)} = g(\tilde{\mathbf{X}}^{(t)}, \boldsymbol{\theta}) \tag{23}$$

where $g(.)$ can be either an LSTM network or a CNN parameterized by $\boldsymbol{\theta}$. We call the LSTM variation of the hybrid GR4J model DeepGR4J-LSTM and CNN variation DeepGR4J-CNN. We also construct variations of the DeepGR4J model where the observed antecedent streamflow ($Q^{(t-1)}$) is passed as an input to the neural network model. These variations of the hybrid LSTM and CNN models are called DeepGR4J-LSTM-Q_{in} and DeepGR4J-CNN-Q_{in}, respectively; where Q_{in} represents the input antecedent streamflow. Input features to the model ($\tilde{\mathbf{x}}$) can then be updated as

$$\tilde{\mathbf{x}}_t = \begin{bmatrix} P^{(t)} \\ E^{(t)} \\ T_{min}^{(t)} \\ T_{max}^{(t)} \\ Q^{(t-1)} \\ P_n^{(t)} \\ E_n^{(t)} \\ P_s^{(t)} \\ Perc^{(t)}. \end{bmatrix} \tag{24}$$

The trainable parameters for different variations of the DeepGR4J hybrid rainfall-runoff model consist of X_1 from the production storage of the conceptual GR4J model and parameters $\boldsymbol{\theta}$ of the respective deep learning models (CNN and LSTM). We note that evolutionary algorithms are best suited for optimizing (calibrating) GR4J model parameters (Napiorkowski et al., 2023), while gradient-based optimization (backpropagation) is primarily used for optimizing (training) deep learning models. Therefore, we present a hierarchical optimization approach as shown in Algorithm 1. The model calibration (training/optimization) consists of four stages. During Stage 1, we initialize the GR4J and neural network model parameters, along with other parameters such as the time-series state-space reconstruction (window size *alpha*). In Stage 2, we optimize the base GR4J model parameters (X_1, X_2, X_3, and X_4) using *differential evolution* (DE) (Storn and Price, 1997) optimization algorithm. Afterwards, we use the optimized values of the GR4J parameters and generate latent features P_n, E_n, P_s, P_{perc} (Stage 3). In Stage 3, we also gather the meteorological features from the dataset (in real-time implementation, these can be obtained from the meteorological station). In Stage 4, we use the latent and meteorological features from Stage 3 to construct hybrid input feature sequences and train the respective deep learning models (LSTM or CNN) using the *adaptive movement estimation* (Adam) optimizer (Kingma and Ba, 2014).

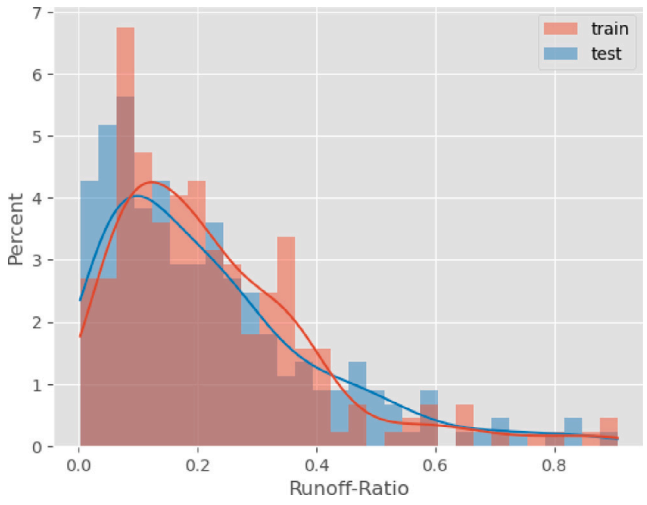


Fig. 5. Histogram of runoff-ratios for train and test data of 222 catchments in CAMELS-AUS data.

We use the trained models to generate streamflow predictions, given by $\hat{Q}$. We use *mean squared error* (MSE) as the training objective function (also called the loss function) for the evolutionary optimization of the GR4J model as well as the gradient-based optimization of the deep learning models. We compute the MSE for streamflow prediction using

$$\mathcal{L} = MSE = \frac{\sum_{t=1}^{T}(\hat{Q}^{(t)} - Q^{(t)})^2}{T} \tag{25}$$

where T is the number of daily timesteps in data, $\hat{Q}^{(t)}$ is the predicted streamflow at time step t and $Q^{(t)}$ is the observed streamflow at time step t. Finally, we evaluate the streamflow prediction performance of our models using the *Nash–Sutcliffe model efficiency coefficient* (NSE) (Nash and Sutcliffe, 1970) given by

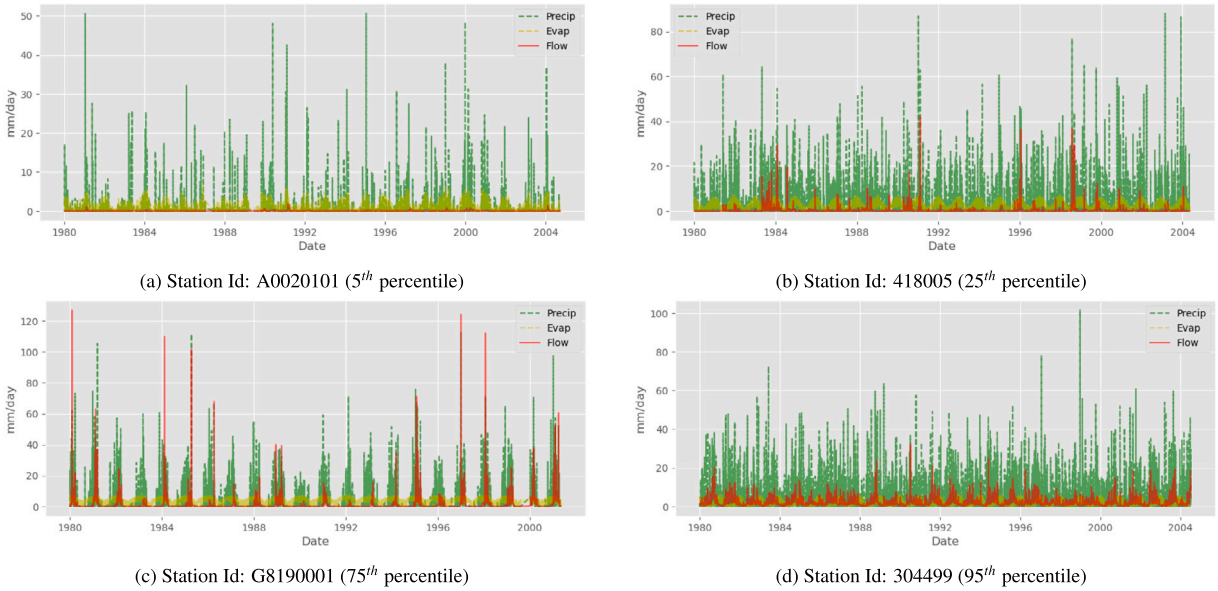
$$NSE = \frac{\sum_{t=1}^{T}(Q^{(t)} - \hat{Q}^{(t)})^2}{\sum_{t=1}^{T}(Q^{(t)} - \overline{Q})^2} \tag{26}$$

where T is the number of daily timesteps in data, $Q^{(t)}$ and $\hat{Q}^{(t)}$ are the observed and predicted streamflow at time step t, respectively. Moreover, $\overline{Q}$ is the mean of observed streamflow across T time steps. The computed value of NSE lies in the range $(-\infty, 1]$. NSE typically compares the model performance to a mean estimator where negative values denote that the model performance is worse than a mean estimator and positive values denote that the model performance is better than the mean estimator. NSE values close to 1 are desired. Although there are other metrics used to evaluate hydrologic model predictions, NSE is most commonly used for streamflow prediction problems. Hence, we use NSE to keep our evaluation metric consistent with the literature and in order to use different metrics to train and evaluate the models. As mentioned, NSE can take values from $-\infty$ to 1 and to standardize these values in a fixed positive range of $[0, 1]$, we normalize the NSE to compute normalized NSE (NNSE) given by

$$NNSE = \frac{1}{2 - NSE}. \tag{27}$$

We provide the open-source software for the DeepGR4J model implemented in Python using the *PyTorch* deep learning library[1] as part of the GitHub repository[2].

[1] pytorch website: https://pytorch.org/
[2] https://github.com/arpit-kapoor/hybrid-gr4j



(a) Station Id: A0020101 (5^{th} percentile)

(b) Station Id: 418005 (25^{th} percentile)

(c) Station Id: G8190001 (75^{th} percentile)

(d) Station Id: 304499 (95^{th} percentile)

Fig. 6. Hydrographs for catchments having Runoff-ratios closest to 5th, 25th, 75th and 95th percentile.

5. Experiments and results

5.1. Data processing

In this study, we use *CAMELS-AUS* hydrometeorological time-series data for 222 unregulated catchments across Australia (Fowler et al., 2021). The CAMELS-AUS data consists of hydrological features (streamflow) and 12 climate variables along with 134 attributes related to geology, soil, topography, land cover, anthropogenic influence, and hydro climatology recorded daily for more than a decade. In about 85% of catchments, the data spans more than 40 years and constitutes catchments with high interannual variability and hence suitable for the study of arid-zone hydrology. This dataset is freely available as part of PANGAEA's Earth and environmental data repository[3]. We used the *camels-aus-py* python package[4] to download and preprocess the data for the experimentation. The runoff-ratio computed as $RR = \frac{\sum_{t=1}^{T} Q^{(t)}}{[\sum_{t=1}]^{T} P^{(t)}}$ is the proportion of precipitation discharged as runoff. This proportion of precipitation is not infiltrated or evapotranspirated. Fig. 5 shows a histogram of runoff ratios of all the catchments in the dataset. Fig. 6 shows the time-series plot of streamflow along with the precipitation and evapotranspiration for 4 catchments consisting of two dry and wet catchments each based on their runoff ratio in the dataset. For all catchments, we selected the data from 1980 to 2014 for the evaluation of different models. We then train each model on 70% of the selected data and use the remaining 30% for testing. In all the variations of the DeepGR4J model, we also normalize the input features and target streamflow values (computed in Eqs. (21) and (24)) by subtracting the mean and dividing by the standard deviation of the respective values.

5.2. Experiment design

We compare the performance of the four variations of our DeepGR4J model (DeepGR4J-CNN, DeepGR4J-CNN, DeepGR4J-LSTM-Q_{in} and DeepGR4J-CNN-Q_{in}) with canonical methods, i.e., GR4J (calibrated using DE) and deep learning models (LSTM and CNN). Hence, we train and test 7 models on all 222 catchments in the CAMELS-AUS dataset. We set the time series reconstruction window size for hybrid models to $\alpha = 7$ denoting 7 days of memory in streamflow prediction. Table 2 shows the model architecture used in the DeepGR4J-LSTM-Q_{in} hybrid model with the number of trainable parameters in each layer. Similarly, Table 3 shows the architecture and number of parameters used in the DeepGR4J-CNN-Q_{in} hybrid model. The first value of the output shape denotes the batch size used for training and testing the deep learning and hybrid models. We empirically determined the number of parameters in each layer in trial experiments and used the best configuration as shown in Table 3. Note that the model variations without the input streamflow (ie. DeepGR4J-LSTM and DeepGR4J-CNN) take one less input, i.e. the number of input features for these models is $d-1$ if d is the number of inputs for the corresponding model with input streamflow.

We used the University of New South Wales (UNSW) high-performance computing (HPC) environment - *Katana* for training

Table 2
LSTM network architecture and parameter configuration for DeepGR4J-LSTM-Q_{in}.

Layer (type)	Output shape	No. of params
LSTM-1	[256, 7, 64]	26,368
LSTM-2	[256, 7, 64]	26,368
Dropout-1	[256, 128]	0
Linear-1	[256, 32]	4,128
Linear-2	[256, 1]	33
Total params: 56,897		
Trainable params: 56,897		
Non-trainable params: 0		

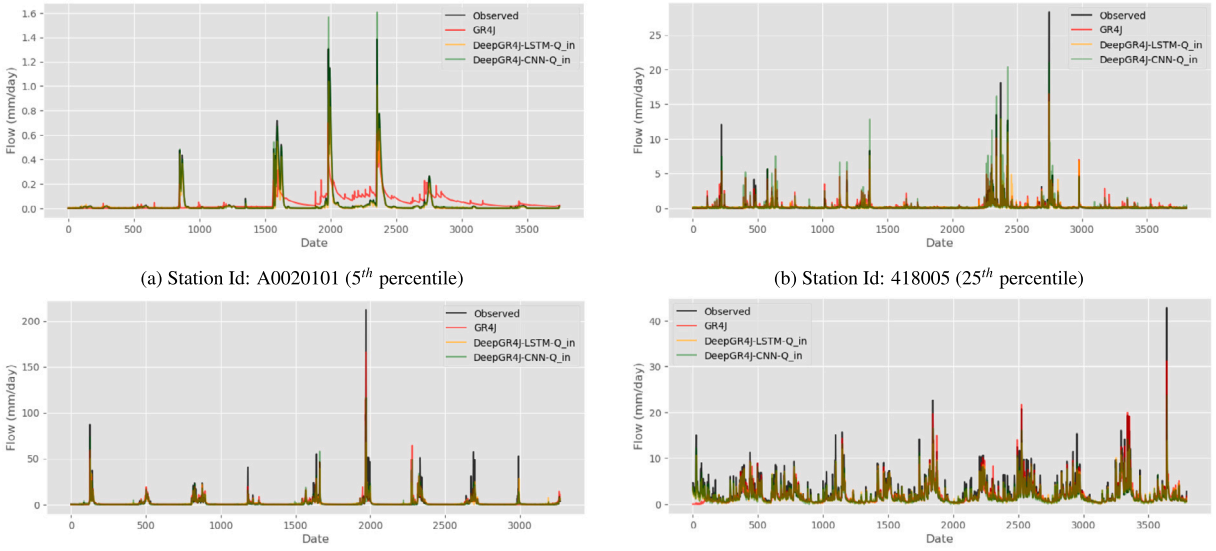
Table 3
CNN architecture and parameter configuration for DeepGR4J-CNN-Q_{in}.

Layer (type)	Output shape	No. of params
Conv2d-1	[256, 8, 6, 10]	24
Conv2d-2	[256, 8, 4, 8]	584
MaxPool2d-3	[256, 8, 2, 8]	0
Conv2d-4	[256, 6, 1, 7]	198
Flatten-5	[256, 42]	0
Dropout-6	[256, 42]	0
Linear-7	[256, 1]	43
Total params: 849		
Trainable params: 849		
Non-trainable params: 0		

[3] https://doi.pangaea.de/10.1594/PANGAEA.921850
[4] https://readthedocs.org/projects/camels-aus-py/

Table 4
Streamflow prediction performance across all catchments in CAMELS Australia dataset. The first value in each column is the mean across all the catchments followed by the 5th and 95th percentile.

Model	NSE		Normalized-NSE	
	Train	Test	Train	Test
GR4J	[0.5840, 0.1685, 0.8737]	[0.2297, −0.507, 0.8634]	[0.7643, 0.5460, 0.8879]	[0.7071, 0.3990, 0.8799]
LSTM	[0.5908, 0.2296, 0.8032]	[0.3753, −0.039, 0.7064]	[0.7201, 0.5649, 0.8356]	[0.6298, 0.4905, 0.7730]
CNN	[0.4375, −0.001, 0.7971]	[0.3520, −0.065, 0.7325]	[0.6586, 0.4997, 0.8313]	[0.6257, 0.4843, 0.7889]
DeepGR4J-LSTM	[0.7081, 0.5657, 0.8483]	[0.5459, 0.2144, 0.7775]	[0.7777, 0.6972, 0.8683]	[0.6986, 0.5601, 0.8180]
DeepGR4J-CNN	[0.7546, 0.5485, 0.8892]	[0.6237, 0.3107, 0.8387]	[0.8089, 0.6889, 0.9003]	[0.7378, 0.5920, 0.8611]
DeepGR4J-LSTM-Q_{in}	[0.7653, 0.6106, 0.9174]	[0.6574, 0.3169, 0.8749]	[0.8148, 0.7198, 0.9237]	[0.7561, 0.5942, 0.8888]
DeepGR4J-CNN-Q_{in}	[0.8256, 0.6478, 0.9556]	[0.7272, 0.4374, 0.9262]	[0.8579, 0.7395, 0.9574]	[0.7981, 0.6400, 0.9313]



(a) Station Id: A0020101 (5^{th} percentile)

(b) Station Id: 418005 (25^{th} percentile)

(c) Station Id: G8190001 (75^{th} percentile)

(d) Station Id: 304499 (95^{th} percentile)

Fig. 7. Test data prediction performance in catchments having Runoff-ratios closest to 5th, 25th, 75th and 95th percentile.

the respective models. Each model run consists of training and evaluation, a single node with 2 high-performance cores of *Intel Xeon Gold 6248 CPU @ 2.50GHz* and 8 GB of memory. We allocated each node to a separate catchment and trained in isolation from the other nodes.

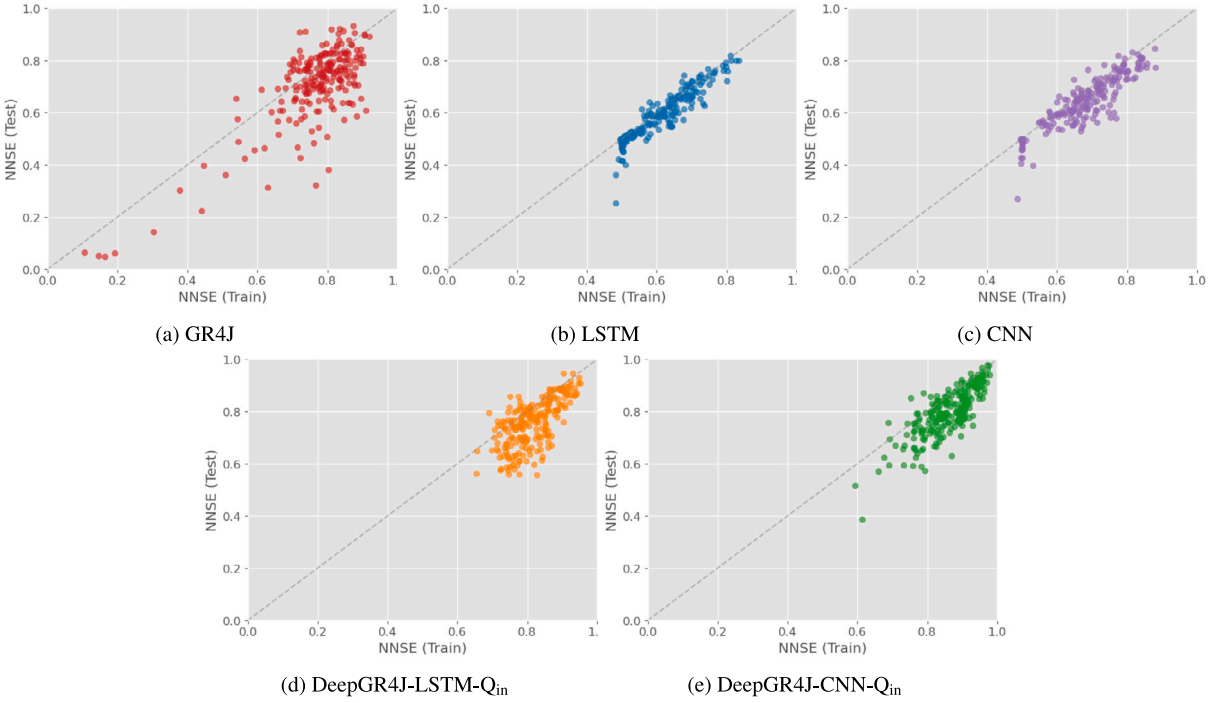
5.3. Results

Table 4 presents NSE performance results for training and test sets of streamflow prediction tasks for 222 catchments in the CAMELS-AUS dataset. The table features the *mean*, 5th percentile and the 95th percentile of two evaluation metrics (NSE and NNSE) across the 222 catchments. We compare four variations of the hybrid rainfall-runoff model (DeepGR4J-CNN, DeepGR4J-CNN, DeepGR4J-LSTM-Q_{in} and DeepGR4J-CNN-Q_{in}) with the base conceptual model (GR4J) and base deep learning models (LSTM and CNN). We observe that out of the 7 models, the CNN-based hybrid GR4J model with input streamflow (DeepGR4J-CNN-Q_{in}) provides the best overall performance both in terms of NSE and the NNSE metrics followed by the LSTM counterpart (DeepGR4J-LSTM-Q_{in}). Although the hybrid models without input streamflow outperform the base models i.e. the GR4J hydrologic model and deep learning models (LSTM and CNN), the addition of historical streamflow (7 days) improves the overall performance. We note that while GR4J performs better than the neural network models on test NNSE, this is not the case when we look at the NSE performance of these models. We infer that this behaviour is due to the range of the two metrics; while the NSE value typically goes from $-\infty$ to 1, the NNSE is normalized between the range 0 and 1. In some outlier catchments with significantly high infiltration (low runoff ratio), the GR4J model shows a high magnitude negative value of NSE. Thus, lowering the overall performance across catchments.

Fig. 7 presents hydrographs with the predicted streamflows from the GR4J model and the two best performing hybrid models (DeepGR4J-LSTM-Q_{in} and DeepGR4J-CNN-Q_{in}). We compare the streamflow predictions with the observed streamflow for the test period for 4 catchments in the data set. We chose the 4 catchments based on the runoff ratios with catchments having runoff ratios closest to the 5th, 25th, 75th, and 95th percentile. We observe that the GR4J predictions for catchments with a high runoff ratio are close to observed and DeepGR4J models. GR4J tends to overestimate the streamflow for catchments with a low runoff ratio (A0020101 and 418005). This problem is mitigated by incorporating data-driven learning in the DeepGR4J model as shown in Fig. 7.

Fig. 8 shows a scatter plot of train-vs-test NNSE performance of various models for all catchments in the CAMELS-AUS dataset. The dotted line denotes a linear function with unit slope ($y = x$). We observe that for most catchments, DeepGR4J models show high train and test NNSE performance with few outliers overfitting (train NNSE > test NNSE). On the other hand, the GR4J model has a high number of catchments scattered below the reference line ($y = x$), meaning GR4J shows poor test predictive performance in a high number of catchments. Both deep learning models (LSTM and CNN) show a tight linear relationship between train and test performance, meaning the models are not overfitting. However, they show a lower NNSE score on both train and test sets for most catchments.

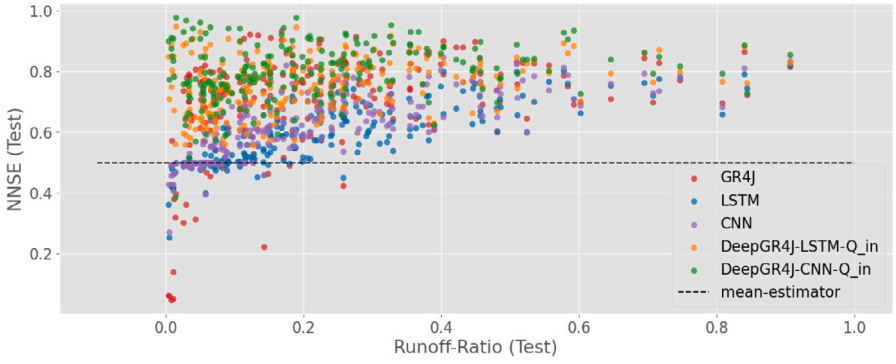
Fig. 9(a) presents a scatter plot of test data NNSE performance against the runoff ratio of all catchments in the CAMELS-AUS dataset. We fit a regression curve of degree 8 for each model (Fig. 9(b)) to show the approximate trend for each model. We compare the performances of the base models (GR4J, LSTM and CNN) with best performing hybrid models. Note that a mean estimator would result in an NNSE value of



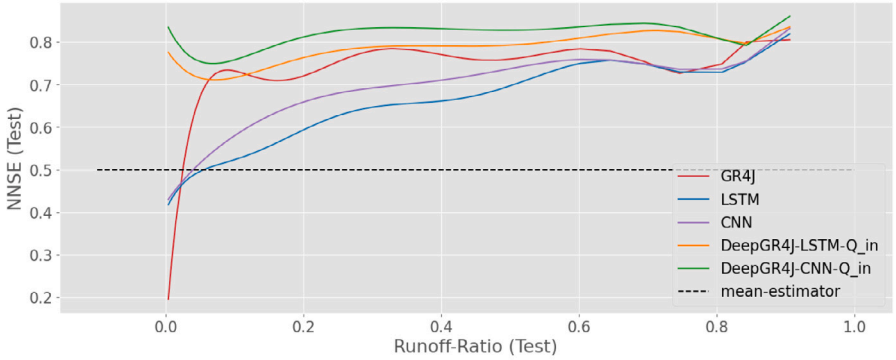
(a) GR4J (b) LSTM (c) CNN

(d) DeepGR4J-LSTM-Q_{in} (e) DeepGR4J-CNN-Q_{in}

Fig. 8. Train vs Test data NNSE performance for selected models.



(a) Scatter plot.



(b) Fitted regression curve.

Fig. 9. NNSE vs Runoff Ratio performance for test data.

0.5. We use the mean estimator as the reference for the performance comparison. We observe that for catchments with high infiltration, the base models (GR4J, LSTM and CNN) perform worse than the mean estimator, while the hybrid models perform consistently well across catchments with varying runoff ratios.

6. Discussion

Our results show that deep learning-based models have the potential for modelling streamflow as an alternative to conceptual models and physically-based hydrologic models. By leveraging large volumes of

observed data, deep learning-based models are capable of modelling streamflow in catchments with varying characteristics. We refer to the GR4J conceptual hydrologic model and the deep learning models as the base models. We have demonstrated that by hybridizing conceptual models and deep learning models, we can overcome the shortcomings of conceptual models (low predictive performance) and deep learning models (overfitting and lack of interpretability). We developed a subprocess hybridization of the GR4J conceptual rainfall-runoff model with deep learning resulting in the DeepGR4J model. We observed that the DeepGR4J hybrid models provided better predictive performance when compared to the base models in catchments with low runoff ratios as well as high runoff ratios. Moreover, by preserving the subprocesses from the production storage and replacing only the subprocesses in the routing storage with a deep learning model, we preserve the interpretability of the conceptual model. The deep learning models the routed runoff as a function of net rainfall and runoff from the production storage.

Furthermore, by hybridizing GR4J with a deep learning model, we were able to incorporate other useful features that influence the total runoff such as minimum and maximum daily surface temperatures into the model. The DeepGR4J model is capable of capturing autocorrelation in the streamflow values. This is presented with further experiments by incorporating antecedent streamflow for the last 7 days into the input features. The results show that the addition of antecedent streamflow on top of the minimum and maximum surface temperatures over a short window (7 days) improves the overall predictive performance of the hybrid model in terms of NSE score.

The subprocesses in the production storage, also known as the surface storage component, compute the net precipitation entering the storage, net evapotranspiration and percolation leaving the storage while maintaining the *water balance* and storage capacity. The water balance condition for the deep learning model input is already satisfied by retaining the production storage in the DeepGR4J hybrid model. Although this restricts the overfitting of the deep learning model in high-flow scenarios, the hybrid models tend to overfit especially in low-flow scenarios. In order to overcome these caveats, the hyperparameters of the model (such as window size, learning rate and weight decay) must be carefully selected. We found that the addition of dropout ($p = 0.2$) in the models and weight decay parameter in the Adam optimizer for both CNN ($\lambda = 0.02$) and LSTM ($\lambda = 0.04$) showed the best results.

In the literature, evolutionary algorithms such as particle swarm optimization (PSO) and differential evolution (DE) have been shown to be well suited for the optimization of GR4J parameters (Napiorkowski et al., 2023). We note that hybridization approaches for streamflow modelling have primarily used genetic algorithms for parameter optimization (Okkan et al., 2021; Ersoy et al., 2022). It is well known that gradient-based optimization (backpropagation) is most suitable for training deep neural networks with high parameter spaces. Therefore, we presented a hierarchical training scheme for the DeepGR4J models, where we first train the GR4J model and retain the calibrated production storage, before training the deep learning model using backpropagation. In GR4J, most of the literature uses evolutionary optimization since gradient-based approaches require gradients which is difficult to obtain.

We analyse the effect of runoff ratio on the predictive performance of conceptual, data-driven and hybrid models. A low runoff ratio corresponds to high infiltration in the catchment whereas a high runoff ratio means low catchment infiltration. Our analysis shows that DeepGR4J-CNN-Q_{in} performs the best across all values of runoff ratio, followed by DeepGR4J-LSTM-Q_{in}. GR4J conceptual model and neural networks typically perform worse in catchments with high infiltration. Hybridization improves the generalization of the model across low to high-infiltration catchments.

While the proposed DeepGR4J sub-process hybridization method has shown promising results, it is currently set up to produce point estimates with no uncertainty quantification. Further work is needed to incorporate Bayesian methods such as Markov Chain Monte Carlo (MCMC) sampling and variational inference into the training scheme for quantifying prediction uncertainty. MCMC methods have been effective for geoscientific models such as landscape evolution (Chandra et al., 2019) and reef evolution models (Pall et al., 2020), and have also been applied to hydrologic models (Smith and Marshall, 2008). Hence, MCMC can be used for GR4J and deep learning based on variational inference (Kapoor et al., 2023) can be used for the deep learning models. Furthermore, physics-informed neural networks (PINNs) (Raissi et al., 2019) can be utilized for more robust modelling of the catchment's dynamics. Lastly, mass conserving LSTM (Hoedt et al., 2021) networks can be incorporated for implementing constraints for conserving total moisture in the catchment.

7. Conclusion

We presented a deep learning-based sub-process hybridization scheme for conceptual hydrological rainfall-runoff models. In the proposed model, we replaced the processes in the routing storage of the GR4J rainfall-runoff model with selected deep learning models (LSTM and CNN). Additionally, we incorporated antecedent streamflow from the last 7 days to improve the overall prediction performance across 222 catchments in the CAMELS-AUS data set. We also developed a hierarchical training scheme for the proposed DeepGR4J model. DeepGR4J used a combination of evolutionary optimization and gradient-descent (backpropagation) for deterministic optimization. The results show that the DeepGR4J hybrid model outperforms the base conceptual model as well as base deep learning models. In particular, the DeepGR4J model shows better predictive performance in arid catchments and provides better generalization where GR4J and conventional deep learning models struggle. Our results motivate future work that can extend the DeepGR4J model by incorporating uncertainty quantification via Bayesian inference, and the use of physics laws into the model training via soft constraints offered by PINNs.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Software and data availability

The data and open-source code can be accessed at the associated GitHub (https://github.com/arpit-kapoor/hybrid-gr4j) repository with the meta-information available in Table 5.

Acknowledgments

The authors would like to thank the Australian Government for supporting this research through the Australian Research Council's Industrial Transformation Training Centre in Data Analytics for Resources and Environments (DARE) (project IC190100031). We thank the University of New South Wales for providing high-performance computing (HPC) resources for training and evaluating the models.

Table 5
Repository information.

Name:	DeepGR4J
Developers:	Arpit Kapoor
Contact email:	arpit.kapoor@unsw.edu.au
Programming language:	Python 3.9+
Compatible Operating System:	MacOS/Linux/Windows
Developed and tested:	Debian GNU/Linux 11
Year Published:	2023
Source availability:	GitHub

References

Abbott, M.B., Bathurst, J.C., Cunge, J.A., O'Connell, P.E., Rasmussen, J., 1986. An introduction to the European hydrological system — Systeme hydrologique Europeen, "SHE", 2: Structure of a physically-based, distributed modelling system. J. Hydrol. 87 (1), 61–77. http://dx.doi.org/10.1016/0022-1694(86)90115-0, URL: https://www.sciencedirect.com/science/article/pii/0022169486901150.

Adnan, R.M., Petroselli, A., Heddam, S., Santos, C.A.G., Kisi, O., 2021. Comparison of different methodologies for rainfall–runoff modeling: machine learning vs conceptual approach. Nat. Hazards 105 (3), 2987–3011. http://dx.doi.org/10.1007/s11069-020-04438-2.

Alzubaidi, L., Zhang, J., Humaidi, A.J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M.A., Al-Amidie, M., Farhan, L., 2021. Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. J. Big Data 8, 1–74.

Beck, H.E., van Dijk, A.I.J.M., de Roo, A., Miralles, D.G., McVicar, T.R., Schellekens, J., Bruijnzeel, L.A., 2016. Global-scale regionalization of hydrologic model parameters. Water Resour. Res. 52 (5), 3599–3622. http://dx.doi.org/10.1002/2015WR018247, URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/2015WR018247. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/2015WR018247.

Bergström, S., 1976. Development and Application of a Conceptual Runoff Model for Scandinavian Catchments. URL: http://urn.kb.se/resolve?urn=urn:nbn:se:smhi:diva-5738.

Beven, K., 1989. Changing ideas in hydrology — The case of physically-based models. J. Hydrol. 105 (1), 157–172. http://dx.doi.org/10.1016/0022-1694(89)90101-7, URL: https://www.sciencedirect.com/science/article/pii/0022169489901017.

Beven, K., 2002. Towards an alternative blueprint for a physically based digitally simulated hydrologic response modelling system. Hydrol. Process. 16 (2), 189–206. http://dx.doi.org/10.1002/hyp.343, URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/hyp.343. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/hyp.343.

Bézenac, E.d., Pajot, A., Gallinari, P., 2019. Deep learning for physical processes: Incorporating prior scientific knowledge*. J. Stat. Mech. Theory Exp. 2019 (12), 124009. http://dx.doi.org/10.1088/1742-5468/ab3195.

Boughton, W., 2004. The Australian water balance model. Environ. Model. Softw. 19 (10), 943–956. http://dx.doi.org/10.1016/j.envsoft.2003.10.007, URL: https://www.sciencedirect.com/science/article/pii/S1364815203002196.

Burnash, R.J.C., 1995. The NWS river forecast system - Catchment modeling. Comput. Models Watershed Hydrol. 311–366, URL: https://www.cabdirect.org/cabdirect/abstract/19961904770. Publisher: Water Resources Publications.

Burnash, R.J., Ferral, R.L., 1973. A Generalized Streamflow Simulation System: Conceptual Modeling for Digital Computers. US Department of Commerce, National Weather Service, and State of California.

Camps-Valls, G., Martino, L., Svendsen, D.H., Campos-Taberner, M., Muñoz-Marí, J., Laparra, V., Luengo, D., García-Haro, F.J., 2018. Physics-aware Gaussian processes in remote sensing. Appl. Soft Comput. 68, 69–82. http://dx.doi.org/10.1016/j.asoc.2018.03.021, URL: https://www.sciencedirect.com/science/article/pii/S1568946618301431.

Chandra, R., Azam, D., Müller, R.D., Salles, T., Cripps, S., 2019. BayesLands: A Bayesian inference approach for parameter uncertainty quantification in Badlands. Comput. Geosci. 131, 89–101.

Chandra, R., Goyal, S., Gupta, R., 2021. Evaluation of deep learning models for multi-step ahead time series prediction. IEEE Access 9, 83105–83123. http://dx.doi.org/10.1109/ACCESS.2021.3085085, Conference Name: IEEE Access.

Chevallier, F., Chéruy, F., Scott, N.A., Chédin, A., 1998. A neural network approach for a fast and accurate computation of a longwave radiative budget. J. Appl. Meteorol. Climatol. 37 (11), 1385–1397. http://dx.doi.org/10.1175/1520-0450(1998)037<1385:ANNAFA>2.0.CO;2, URL: https://journals.ametsoc.org/view/journals/apme/37/11/1520-0450_1998_037_1385_annafa_2.0.co_2.xml. Publisher: American Meteorological Society Section: Journal of Applied Meteorology and Climatology.

Chung, J., Gulcehre, C., Cho, K., Bengio, Y., 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv.org. URL: https://arxiv.org/abs/1412.3555v1.

Cigizoglu, H.K., Alp, M., 2004. Rainfall-runoff modelling using three neural network methods. In: ICAISC. Springer, pp. 166–171.

Dawson, C.W., Wilby, R., 1998. An artificial neural network approach to rainfall-runoff modelling. Hydrol. Sci. J. 43 (1), 47–66. http://dx.doi.org/10.1080/02626669809492102, Publisher: Taylor & Francis _eprint.

Devia, G.K., Ganasri, B.P., Dwarakish, G.S., 2015. A review on hydrological models. INTERNATIONAL CONFERENCE ON WATER RESOURCES, COASTAL AND OCEAN ENGINEERING (ICWRCOE'15), Aquatic Procedia INTERNATIONAL CONFERENCE ON WATER RESOURCES, COASTAL AND OCEAN ENGINEERING (ICWRCOE'15), 4, 1001–1007. http://dx.doi.org/10.1016/j.aqpro.2015.02.126.URL: https://www.sciencedirect.com/science/article/pii/S2214241X15001273,

Elman, J.L., 1990. Finding structure in time. Cogn. Sci. 14 (2), 179–211. http://dx.doi.org/10.1016/0364-0213(90)90002-E, URL: https://www.sciencedirect.com/science/article/pii/036402139090002E.

Ersoy, Z.B., Okkan, U., Fıstıkoğlu, O., 2022. Hybridizing a conceptual hydrological model with neural networks to enhance runoff prediction. Manchester J. Artif. Intell. Appl. Sci. 3 (1), 6, URL: https://www.mjaias.co.uk/mj-en/article/view/36. Number: 1.

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 Resour. Res. 56 (9), http://dx.doi.org/10.1029/2019WR026793, e2019WR026793. URL: https://onlinelibrary.wiley.com/doi/abs/10.1029/2019WR026793. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/2019WR026793.

Fowler, K.J.A., Acharya, S.C., Addor, N., Chou, C., Peel, M.C., 2021. CAMELS-AUS: hydrometeorological time series and landscape attributes for 222 catchments in Australia. Earth Syst. Sci. Data 13 (8), 3847–3867. http://dx.doi.org/10.5194/essd-13-3847-2021, URL: https://essd.copernicus.org/articles/13/3847/2021/. Publisher: Copernicus GmbH.

Franchini, M., 1996. Use of a genetic algorithm combined with a local search method for the automatic calibration of conceptual rainfall-runoff models. Hydrol. Sci. J. 41 (1), 21–39. http://dx.doi.org/10.1080/02626669609491476, Publisher: Taylor & Francis _eprint.

Gill, M.K., Kaheil, Y.H., Khalil, A., McKee, M., Bastidas, L., 2006. Multiobjective particle swarm optimization for parameter estimation in hydrology. Water Resour. Res. 42 (7).

Guo, J., Zhou, J., Zou, Q., Liu, Y., Song, L., 2013. A novel multi-objective shuffled complex differential evolution algorithm with application to hydrological model parameter optimization. Water Resour. Manag. 27, 2923–2946.

Hatmoko, W., Diaz, B., et al., 2020. Comparison of rainfall-runoff models for climate change projection–case study of Citarum River Basin, Indonesia. In: IOP Conf. Ser. Earth Environ. Sci.. 423, (1), IOP Publishing, 012045.

Hochreiter, S., 1998. The vanishing gradient problem during learning recurrent neural nets and problem solutions. Int. J. Uncertain. Fuzziness Knowl.-Based Syst. 06 (02), 107–116. http://dx.doi.org/10.1142/S0218488598000094, URL: https://www.worldscientific.com/doi/abs/10.1142/s0218488598000094. Publisher: World Scientific Publishing Co.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780. http://dx.doi.org/10.1162/neco.1997.9.8.1735, Conference Name: Neural Computation.

Hoedt, P.-J., Kratzert, F., Klotz, D., Halmich, C., Holzleitner, M., Nearing, G.S., Hochreiter, S., Klambauer, G., 2021. MC-LSTM: Mass-conserving LSTM. In: Proceedings of the 38th International Conference on Machine Learning. PMLR, pp. 4275–4286, URL: https://proceedings.mlr.press/v139/hoedt21a.html. ISSN: 2640-3498.

Höge, M., Scheidegger, A., Baity-Jesi, M., Albert, C., Fenicia, F., 2022. Improving hydrologic models for predictions and process understanding using neural ODEs. Hydrol. Earth Syst. Sci. 26 (19), 5085–5102. http://dx.doi.org/10.5194/hess-26-5085-2022, URL: https://hess.copernicus.org/articles/26/5085/2022/. Publisher: Copernicus GmbH.

Holzinger, A., Saranti, A., Molnar, C., Biecek, P., Samek, W., 2022. Explainable AI methods - a brief overview. In: Holzinger, A., Goebel, R., Fong, R., Moon, T., Müller, K.-R., Samek, W. (Eds.), XxAI - beyond Explainable AI: International Workshop, Held in Conjunction with ICML 2020, July 18, 2020, Vienna, Austria, Revised and Extended Papers. In: Lecture Notes in Computer Science, Springer International Publishing, Cham, pp. 13–38. http://dx.doi.org/10.1007/978-3-031-04083-2_2.

Jaiswal, R.K., Ali, S., Bharti, B., 2020. Comparative evaluation of conceptual and physical rainfall–runoff models. Appl. Water Sci. 10 (1), 48. http://dx.doi.org/10.1007/s13201-019-1122-6.

Jehanzaib, M., Ajmal, M., Achite, M., Kim, T.-W., 2022. Comprehensive review: Advancements in rainfall-runoff modelling for flood mitigation. Climate 10 (10), 147. http://dx.doi.org/10.3390/cli10100147, URL: https://www.mdpi.com/2225-1154/10/10/147. Number: 10 Publisher: Multidisciplinary Digital Publishing Institute.

Jiang, Y., Liu, C., Huang, C., Wu, X., 2010. Improved particle swarm algorithm for hydrological parameter optimization. Appl. Math. Comput. 217 (7), 3207–3215.

Kapoor, A., Negi, A., Marshall, L., Chandra, R., 2023. Cyclone trajectory and intensity prediction with uncertainty quantification using variational recurrent neural networks. Environ. Model. Softw. 162, 105654, Publisher: Elsevier.

Khu, S.-T., Madsen, H., Di Pierro, F., 2008. Incorporating multiple observations for distributed hydrologic model calibration: An approach using a multi-objective evolutionary algorithm and clustering. Adv. Water Resour. 31 (10), 1387–1398.

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kisi, O., Shiri, J., Tombul, M., 2013. Modeling rainfall-runoff process using soft computing techniques. Comput. Geosci. 51, 108–117. http://dx.doi.org/10.1016/j.cageo.2012.07.001, URL: https://www.sciencedirect.com/science/article/pii/S0098300412002257.

Krapu, C., Borsuk, M., Kumar, M., 2019. Gradient-based inverse estimation for a rainfall-runoff model. Water Resour. Res. 55 (8), 6625–6639. http://dx.doi.org/10.1029/2018WR024461, URL: https://onlinelibrary.wiley.com/doi/abs/10.1029/2018WR024461. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/2018WR024461.

Kratzert, F., Klotz, D., Brenner, C., Schulz, K., Herrnegger, M., 2018. Rainfall–runoff modelling using Long Short-Term Memory (LSTM) networks. Hydrol. Earth Syst. Sci. 22 (11), 6005–6022. http://dx.doi.org/10.5194/hess-22-6005-2018, URL: https://hess.copernicus.org/articles/22/6005/2018/. Publisher: Copernicus GmbH.

Kumanlioglu, A.A., Fistikoglu, O., 2019. Performance enhancement of a conceptual hydrological model by integrating artificial intelligence. J. Hydrol. Eng. 24 (11), 04019047. http://dx.doi.org/10.1061/(ASCE)HE.1943-5584.0001850, URL: https://ascelibrary.org/doi/10.1061/%28ASCE%29HE.1943-5584.0001850. Publisher: American Society of Civil Engineers.

Le Moine, N., 2008. Le Bassin Versant De Surface Vu Par Le Souterrain: Une Voie D'amélioration Des Performances Et Du Réalisme Des Modèles Pluie-Débit? (Ph.D. thesis). Doctorat Géosciences et Ressources Naturelles, Université Pierre et Marie.

LeCun, Y., Bengio, Y., et al., 1995. Convolutional networks for images, speech, and time series. In: The Handbook of Brain Theory and Neural Networks, Vol. 3361, No. 10. Citeseer, p. 1995.

LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., Jackel, L., 1989. Handwritten digit recognition with a back-propagation network. In: Advances in Neural Information Processing Systems, Vol. 2. Morgan-Kaufmann, URL: https://proceedings.neurips.cc/paper/1989/hash/53c3bce66e43be4f209556518c2fcb54-Abstract.html.

Lees, T., Buechel, M., Anderson, B., Slater, L., Reece, S., Coxon, G., Dadson, S.J., 2021. Benchmarking data-driven rainfall–runoff models in Great Britain: a comparison of long short-term memory (LSTM)-based models with four lumped conceptual models. Hydrol. Earth Syst. Sci. 25 (10), 5517–5534. http://dx.doi.org/10.5194/hess-25-5517-2021, URL: https://hess.copernicus.org/articles/25/5517/2021/. Publisher: Copernicus GmbH.

Lees, T., Reece, S., Kratzert, F., Klotz, D., Gauch, M., De Bruijn, J., Kumar Sahu, R., Greve, P., Slater, L., Dadson, S.J., 2022. Hydrological concept formation inside long short-term memory (LSTM) networks. Hydrol. Earth Syst. Sci. 26 (12), 3079–3101. http://dx.doi.org/10.5194/hess-26-3079-2022, URL: https://hess.copernicus.org/articles/26/3079/2022/. Publisher: Copernicus GmbH.

Li, D., Marshall, L., Liang, Z., Sharma, A., Zhou, Y., 2021a. Bayesian LSTM with stochastic variational inference for estimating model uncertainty in process-based hydrological models. Water Resour. Res. 57 (9), http://dx.doi.org/10.1029/2021WR029772, e2021WR029772. URL: https://onlinelibrary.wiley.com/doi/abs/10.1029/2021WR029772. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/2021WR029772.

Li, D., Marshall, L., Liang, Z., Sharma, A., Zhou, Y., 2021b. Characterizing distributed hydrological model residual errors using a probabilistic long short-term memory network. J. Hydrol. 603, 126888, Publisher: Elsevier.

Liu, Y., 2009. Automatic calibration of a rainfall–runoff model using a fast and elitist multi-objective particle swarm algorithm. Expert Syst. Appl. 36 (5), 9533–9538. http://dx.doi.org/10.1016/j.eswa.2008.10.086, URL: https://www.sciencedirect.com/science/article/pii/S0957417408007823.

Ma, K., Feng, D., Lawson, K., Tsai, W.-P., Liang, C., Huang, X., Sharma, A., Shen, C., 2021. Transferring hydrologic data across continents – leveraging data-rich regions to improve hydrologic prediction in data-sparse regions. Water Resour. Res. 57 (5), http://dx.doi.org/10.1029/2020WR028600, e2020WR028600. URL: https://onlinelibrary.wiley.com/doi/abs/10.1029/2020WR028600. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/2020WR028600.

Maniquiz, M.C., Lee, S., Kim, L.-H., 2010. Multiple linear regression models of urban runoff pollutant load and event mean concentration considering rainfall variables. J. Environ. Sci. 22 (6), 946–952. http://dx.doi.org/10.1016/S1001-0742(09)60203-5, URL: https://www.sciencedirect.com/science/article/pii/S1001074209602035.

Mishra, S., Saravanan, C., Dwivedi, V.K., Shukla, J.P., 2018. Rainfall-runoff modeling using clustering and regression analysis for the river Brahmaputra basin. J. Geol. Soc. India 92 (3), 305–312. http://dx.doi.org/10.1007/s12594-018-1012-9.

Mohammadi, B., Safari, M.J.S., Vazifehkhah, S., 2022. IHACRES, GR4J and MISD-based multi conceptual-machine learning approach for rainfall-runoff modeling. Sci. Rep. 12 (1), 12096. http://dx.doi.org/10.1038/s41598-022-16215-1, URL: https://www.nature.com/articles/s41598-022-16215-1. Number: 1 Publisher: Nature Publishing Group.

Montavon, G., Samek, W., Müller, K.-R., 2018. Methods for interpreting and understanding deep neural networks. Digit. Signal Process. 73, 1–15. http://dx.doi.org/10.1016/j.dsp.2017.10.011, URL: http://arxiv.org/abs/1706.07979. arXiv:1706.07979 [cs, stat].

Napiorkowski, J.J., Piotrowski, A.P., Karamuz, E., Senbeta, T.B., 2022. Calibration of conceptual rainfall-runoff models by selected differential evolution and particle swarm optimization variants. Acta Geophys. http://dx.doi.org/10.1007/s11600-022-00988-0.

Napiorkowski, J.J., Piotrowski, A.P., Karamuz, E., Senbeta, T.B., 2023. Calibration of conceptual rainfall-runoff models by selected differential evolution and particle swarm optimization variants. Acta Geophys. 71 (5), 2325–2338.

Nash, J.E., Sutcliffe, J.V., 1970. River flow forecasting through conceptual models part I — A discussion of principles. J. Hydrol. 10 (3), 282–290. http://dx.doi.org/10.1016/0022-1694(70)90255-6, URL: https://www.sciencedirect.com/science/article/pii/0022169470902556.

Nevo, S., Morin, E., Gerzi Rosenthal, A., Metzger, A., Barshai, C., Weitzner, D., Voloshin, D., Kratzert, F., Elidan, G., Dror, G., Begelman, G., Nearing, G., Shalev, G., Noga, H., Shavitt, I., Yuklea, L., Royz, M., Giladi, N., Peled Levi, N., Reich, O., Gilon, O., Maor, R., Timnat, S., Shechter, T., Anisimov, V., Gigi, Y., Levin, Y., Moshe, Z., Ben-Haim, Z., Hassidim, A., Matias, Y., 2022. Flood forecasting with machine learning models in an operational framework. Hydrol. Earth Syst. Sci. 26 (15), 4013–4032. http://dx.doi.org/10.5194/hess-26-4013-2022, URL: https://hess.copernicus.org/articles/26/4013/2022/. Publisher: Copernicus GmbH.

Okkan, U., Ersoy, Z.B., Ali Kumanlioglu, A., Fistikoglu, O., 2021. Embedding machine learning techniques into a conceptual model to improve monthly runoff simulation: A nested hybrid rainfall-runoff modeling. J. Hydrol. 598, 126433. http://dx.doi.org/10.1016/j.jhydrol.2021.126433, URL: https://www.sciencedirect.com/science/article/pii/S0022169421004807.

Pall, J., Chandra, R., Azam, D., Salles, T., Webster, J.M., Scalzo, R., Cripps, S., 2020. Bayesreef: a Bayesian inference framework for modelling reef growth in response to environmental change and biological dynamics. Environ. Model. Softw. 125, 104610.

Paniconi, C., Putti, M., 2015. Physically based modeling in catchment hydrology at 50: Survey and outlook. Water Resour. Res. 51 (9), 7090–7129. http://dx.doi.org/10.1002/2015WR017780, URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/2015WR017780. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/2015WR017780.

Pektaş, A.O., Kerem Cigizoglu, H., 2013. ANN hybrid model versus ARIMA and ARIMAX models of runoff coefficient. J. Hydrol. 500, 21–36. http://dx.doi.org/10.1016/j.jhydrol.2013.07.020, URL: https://www.sciencedirect.com/science/article/pii/S0022169413005374.

Perrin, C., Michel, C., Andréassian, V., 2003. Improvement of a parsimonious model for streamflow simulation. J. Hydrol. 279 (1), 275–289. http://dx.doi.org/10.1016/S0022-1694(03)00225-7, URL: https://www.sciencedirect.com/science/article/pii/S0022169403002257.

Perrin, C., Michel, C., Andréassian, V., 2007. Modèles hydrologiques du génie rural (GR). In: Cemagref, UR Hydrosystèmes et Bioprocédés. Vol. 16.

Pouyanfar, S., Sadiq, S., Yan, Y., Tian, H., Tao, Y., Reyes, M.P., Shyu, M.-L., Chen, S.-C., Iyengar, S.S., 2018. A survey on deep learning: Algorithms, techniques, and applications. ACM Comput. Surv. 51 (5), 1–36.

Pushpalatha, R., Perrin, C., Le Moine, N., Mathevet, T., Andréassian, V., 2011. A downward structural sensitivity analysis of hydrological models to improve low-flow simulation. J. Hydrol. 411 (1), 66–76. http://dx.doi.org/10.1016/j.jhydrol.2011.09.034, URL: https://www.sciencedirect.com/science/article/pii/S0022169411006846.

Raissi, M., Perdikaris, P., Karniadakis, G., 2019. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J. Comput. Phys. 378, 686–707. http://dx.doi.org/10.1016/j.jcp.2018.10.045, URL: https://linkinghub.elsevier.com/retrieve/pii/S0021999118307125.

Razavi, S., Hannah, D.M., Elshorbagy, A., Kumar, S., Marshall, L., Solomatine, D.P., Dezfuli, A., Sadegh, M., Famiglietti, J., 2022. Coevolution of machine learning and process-based modelling to revolutionize Earth and environmental sciences: A perspective. Hydrol. Process. 36 (6), e14596. http://dx.doi.org/10.1002/hyp.14596, URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/hyp.14596. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/hyp.14596.

Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N., Prabhat, 2019. Deep learning and process understanding for data-driven Earth system science. Nature 566 (7743), 195–204. http://dx.doi.org/10.1038/s41586-019-0912-1, URL: https://www.nature.com/articles/s41586-019-0912-1. Number: 7743 Publisher: Nature Publishing Group.

Ruder, S., 2016. An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747.

Samek, W., Montavon, G., Vedaldi, A., Hansen, L.K., Müller, K.-R. (Eds.), 2019. Explainable AI: Interpreting, Explaining and Visualizing Deep Learning. In: Lecture Notes in Computer Science, vol. 11700, Springer International Publishing, Cham, http://dx.doi.org/10.1007/978-3-030-28954-6, URL: http://link.springer.com/10.1007/978-3-030-28954-6.

Sezen, C., Partal, T., 2018. The utilization of a GR4J model and wavelet-based artificial neural network for rainfall–runoff modelling. Water Supply 19 (5), 1295–1304. http://dx.doi.org/10.2166/ws.2018.189.

Smith, T.J., Marshall, L.A., 2008. Bayesian methods in hydrologic modeling: A study of recent advancements in Markov chain Monte Carlo techniques. Water Resour. Res. 44 (12).

Solomatine, D.P., Wagener, T., 2011. 2.16 - Hydrological modeling. In: Wilderer, P. (Ed.), Treatise on Water Science. Elsevier, Oxford, pp. 435–457. http://dx.doi.org/10.1016/B978-0-444-53199-5.00044-0, URL: https://www.sciencedirect.com/science/article/pii/B9780444531995000440.

Srinivasulu, S., Jain, A., 2006. A comparative analysis of training methods for artificial neural network rainfall–runoff models. Appl. Soft Comput. 6 (3), 295–306. http://dx.doi.org/10.1016/j.asoc.2005.02.002, URL: https://www.sciencedirect.com/science/article/pii/S1568494605000190.

Storn, R., Price, K., 1997. Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J. Glob. Optim. 11 (4), 341.

Takens, F., 2006. Detecting strange attractors in turbulence. In: Dynamical Systems and Turbulence, Warwick 1980: Proceedings of a Symposium Held At the University of Warwick 1979/80. Springer, pp. 366–381.

Tian, Y., Xu, Y.-P., Yang, Z., Wang, G., Zhu, Q., 2018. Integration of a parsimonious hydrological model with recurrent neural networks for improved streamflow forecasting. Water 10 (11), 1655. http://dx.doi.org/10.3390/w10111655, URL: https://www.mdpi.com/2073-4441/10/11/1655. Number: 11 Publisher: Multidisciplinary Digital Publishing Institute.

Tjoa, E., Guan, C., 2021. A survey on explainable artificial intelligence (XAI): Toward medical XAI. IEEE Trans. Neural Netw. Learn. Syst. 32 (11), 4793–4813. http://dx.doi.org/10.1109/TNNLS.2020.3027314, Conference Name: IEEE Transactions on Neural Networks and Learning Systems.

Tokar, A.S., Johnson, P.A., 1999. Rainfall-runoff modeling using artificial neural networks. J. Hydrol. Eng. 4 (3), 232–239. http://dx.doi.org/10.1061/(ASCE)1084-0699(1999)4:3(232), URL: https://ascelibrary.org/doi/10.1061/%28ASCE%291084-0699%281999%294%3A3%28232%29. Publisher: American Society of Civil Engineers.

Valipour, M., 2015. Long-term runoff study using SARIMA and ARIMA models in the United States. Meteorol. Appl. 22 (3), 592–598. http://dx.doi.org/10.1002/met.1491, URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/met.1491. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/met.1491.

Vandal, T., Kodra, E., Ganguly, S., Michaelis, A., Nemani, R., Ganguly, A.R., 2018. Generating high resolution climate change projections through single image super-resolution: An abridged version. pp. 5389–5393, URL: https://www.ijcai.org/proceedings/2018/759.

Wang, Q.J., 1991. The genetic algorithm and its application to calibrating conceptual rainfall-runoff models. Water Resour. Res. 27 (9), 2467–2471. http://dx.doi.org/10.1029/91WR01305, URL: https://onlinelibrary.wiley.com/doi/abs/10.1029/91WR01305. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/91WR01305.

Wang, S., Cao, J., Philip, S.Y., 2020. Deep learning for spatio-temporal data mining: A survey. IEEE Trans. Knowl. Data Eng. 34 (8), 3681–3700.

Wu, S.-J., Lien, H.-C., Chang, C.-H., 2011. Calibration of a conceptual rainfall–runoff model using a genetic algorithm integrated with runoff estimation sensitivity to parameters. J. Hydroinform. 14 (2), 497–511. http://dx.doi.org/10.2166/hydro.2011.010.

Xiang, Z., Yan, J., Demir, I., 2020. A rainfall-runoff model with LSTM-Based sequence-to-sequence learning. Water Resour. Res. 56 (1), http://dx.doi.org/10.1029/2019WR025326, e2019WR025326. URL: https://onlinelibrary.wiley.com/doi/abs/10.1029/2019WR025326. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/2019WR025326.