RGrN-MCLSTM flood flow prediction method with optimized input features

Han Liu[a,b], He-xuan Hu[a,b,*], Qiang Hu[a,b], Zhi-hao Chen[a,b], Qi Sun[a,b], Hao Lu[a,b], Ye Zhang[a,b]

[a] *Key Laboratory of Water Big Data Technology of Ministry of Water Resources, Hohai University, Nanjing* 211100, *China*
[b] *College of Computer and Information, Hohai University, Nanjing* 211100, *China*

Received 19 October, 2023; accepted 15 January, 2024

Abstract: Deep learning-based flood prediction methods rely on a large amount of observation data and face severe challenges in data-scarce areas, making it difficult to provide reliable flood guidance to local authorities. To address the problem of flood flow prediction in data-scarce areas, this paper proposes a RGrN-MCLSTM flood flow prediction method with optimized input features. First, the method uses RGrN to establish a relational graph network to obtain observation data from neighboring locations, which are supplemented in the model training to achieve data augmentation. Second, the method utilizes the quality accumulator of MCLSTM to add water balance constraints, guiding the model training and introducing prior knowledge. The combination of these two technologies effectively alleviates the problem of insufficient observation data in flood flow prediction in data-scarce areas. However, the input data of this method contains a lot of noise and redundant features, which can affect the prediction accuracy of the model in data-scarce areas. Therefore, the deep residual shrinkage network is improved from the aspects of soft threshold function and attention mechanism, and applied to the optimization of input features of the method to improve the prediction accuracy of the model. Experimental research shows that this method can improve the accuracy of flood flow prediction in data-scarce areas.

Keywords: Deep learning; Data scarcity; Flood flow prediction; RGrN-MCLSTM

1. Introduction

In recent years, neural networks have received widespread attention due to the rise of deep learning and have been playing an important role in engineering applications, especially in the field of water resources represented by hydrological modeling (Kratzert et al., 2018a). Like hydrological modeling, the success of deep learning methods largely relies on advancements in computer technology and the availability of massive datasets. However, the dependence of deep learning methods on a large amount of observational data poses significant challenges when it comes to modeling problems in data-scarce regions (Kratzer at al., 2019).

Floods are one of the most common and deadly natural disasters in the world, with an average of about 20 000 deaths and affecting approximately 75 million people globally each year (Sányal, 2023). Flood flow prediction is an effective method for disaster mitigation, which involves forecasting the runoff formed by hydrological processes such as rainfall and snowmelt in a watershed over a certain period of time through the establishment of mathematical models. Unfortunately, accurate predictions are difficult to obtain in most flood-affected areas due to the lack of complete and accurate hydrological observation data, leading to data scarcity (Momoi et al., 2023). Data-scarce regions are often remote areas with inadequate meteorological and hydrological monitoring equipment, resulting in insufficient observational data and rendering conventional flood flow prediction methods ineffective (Beh-Haim et al., 2019).

Data scarcity can be considered a low-resource problem in the realm of deep learning, and an effective solution is to augment the training data with data that is as similar as possible to the observational data used for training (Shorten and Khoshgoftaar, 2019). Sub-basins from upstream to downstream within the same watershed have interactions through lateral water flow, giving the observational data a certain degree of similarity (Ravindranath et al., 2016), Therefore, effectively utilizing the observational data from adjacent sub-basins in data-scarce regions as augmentation can facilitate deep learning methods in flood flow prediction modeling. Additionally, the field of hydrological modeling employs the concept of water balance, which refers to the balance between input and output water volumes within a basin, region, or water body over a given time period (e.g.,

This work was supported by the National Key Research and Development Program of China (Grant No. 2018YFC0407904)

* Corresponding author.
E-mail address: hexuan_hu@hhu.edu.cn.

hourly, daily, monthly, yearly, etc.). Incorporating prior knowledge into deep learning models by imposing water balance constraints during training can effectively learn the variations in hydrological data and reduce prediction errors caused by data scarcity, thereby improving the accuracy of deep learning methods in flood flow prediction in data-scarce regions (Beven, 2020).

Furthermore, it should be noted that the input of flood flow prediction models consists of multidimensional data, including flow, precipitation, maximum temperature, minimum temperature, evapotranspiration, shortwave radiation, etc. These data often contain noise and redundant features, which can affect the accuracy and efficiency of the models, especially in data-scarce regions (Li et al., 2023). Therefore, optimizing the multidimensional input data is necessary for deep learning methods in flood flow prediction modeling in data-scarce regions.

Currently, most flood flow prediction work is based on traditional process models and deep learning models. Traditional process models are generally based on hydraulic knowledge. Hu et al. (2022a) proposed a two-order attention mechanism (GAT-DALSTM) model using graph attention network and long short-term memory network (LSTM) to forecast runoff. The reliability and practicability of the model are verified by analyzing the hotspot map of attention coefficient from the perspective of hydrology and application. The proposed model can provide technical support for improving the prediction accuracy of watershed runoff volume and model transparency. Sellami et al. (2014) used the SWAT model in a Mediterranean basin and employed a parameter transfer approach for flow simulation, revealing that the higher the attribute similarity between basins, the more similar their hydrological characteristics. On the other hand, deep learning models mine the evolution trends of flow from historical data. Karpatne et al. (2017) and Wang et al. (2020) proposed theoretically guided neural network frameworks, formally conceptualizing the paradigm of theory-guided data science and translating scientific laws and engineering theories into loss functions to construct deep learning models for flow prediction. Kratzert et al. (2018b) applied LSTM to rainfall-runoff simulation and analyzed the generalization ability of LSTM for rainfall-runoff response by training a unified model for multiple basins. Hu et al. (2022b) proposed a spatial prediction technique for flood sensitivity based on multi-core learning (MKL), using MODIS remote sensing images to obtain historical flood inundation points in the study area. The results show that MODIS remote sensing images and flood sensitivity maps generated by MKL can provide effective help for researchers and decision makers in flood management.

Based on the analysis above, this paper addresses the low-resource problem faced by deep learning-based flood flow prediction in data-scarce areas by proposing a flood flow prediction method called RGrN-MCLSTM. The method adopts an encoder-decoder structure that combines the recurrent graph network (RGrN) and the mass-conserving LSTM (MCLSTM). The RGrN establishes a network of watershed relationships to augment the training data using neighboring observation data, while the MCLSTM incorporates prior knowledge of water balance through a quality accumulator module to constrain the model training. To address challenges such as high noise and redundant features in the input of the RGrN-MCLSTM flood flow prediction method, an improved deep residual shrinkage network (IDRS) is proposed to optimize the input features. The IDRS effectively suppresses noise and focuses on important features, enhancing the accuracy of the model's predictions.

2. IDRS-RGrN-MCLSTM

2.1. Flood flow prediction method based on RGrN-MCLSTM

The flood flow prediction method based on RGrN-MCLSTM is designed and constructed using a RGrN and a MCLSTM. It employs an encoder-decoder structure as illustrated in Fig. 1. The RGrN serves as the encoder, extracting features from hydrological observations of the watershed and its neighboring regions to generate intermediate states. The MCLSTM acts as the decoder, decoding the features of the intermediate states to produce the prediction results. By combining these two components, the model addresses the issue of data scarcity through data augmentation and the incorporation of prior knowledge. This enables the development of a flood flow prediction model.

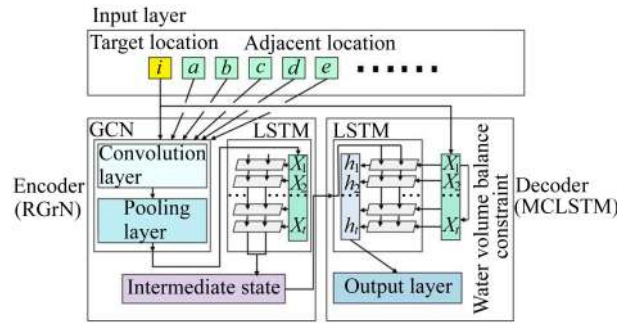


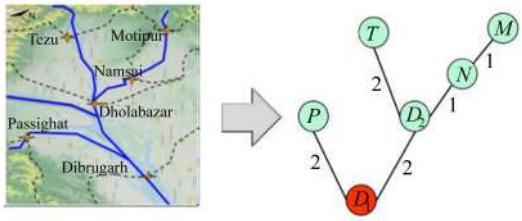
Fig. 1. Coding and decoding framework of RGrN-MCLSTM flood flow prediction model

2.1.1. Encoder (RGrN)

RGrN is a global deep learning model architecture that combines the principles of the recurrent neural network (RNN) and graph convolutional neural network (GCN). The objective of RGrN is to capture the dynamic changes in flow within a connected set of sub-watershed locations. The connections between these loca-

tions can be represented using a graph structure $G=\{V, E, \boldsymbol{A}\}$, where V represents the set of locations, E represents the set of connections between locations, and $\boldsymbol{A}$ represents the adjacency matrix that describing the adjacency level between each pair of locations.

By establishing a relational graph using a GCN, the relevant locations within the same watershed are connected, transforming the concrete physical watershed into an abstract undirected graph with connectivity. This connectivity is stored using an adjacency matrix, as shown in Fig. 2. The spatial graph convolution operation is then used to extract features based on the adjacency matrix generated from the relational graph.



	D_1	D_2	N	M	T	P
D_1	0	2	0	0	0	2
D_2	2	0	1	0	2	0
N	0	1	0	1	0	0
M	0	0	1	0	0	0
T	0	2	0	0	0	0
P	2	0	0	0	0	0

Fig. 2. GCN construction diagram and its adjacency matrix example

When predicting the flow at the target location, it is necessary to merge hydrological observations from previous time steps and neighboring locations (primarily upstream positions) as inputs, including flow data and meteorological data, as shown in Fig. 3.

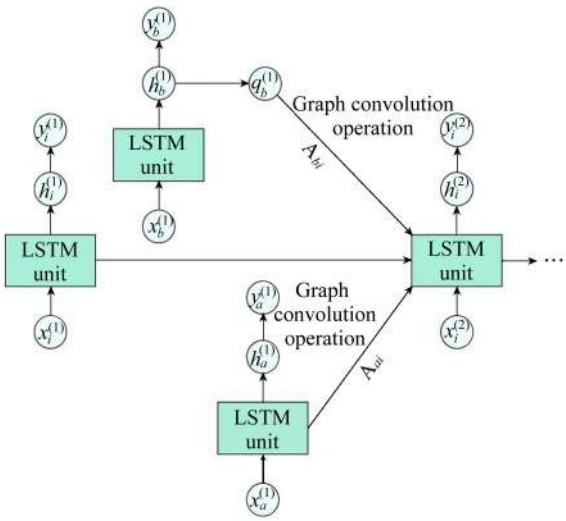


Fig. 3. RGrN structure

For each position i at time step $t-1$, the model extracts latent variables containing hydrological observations from neighboring positions and propagates them downstream. For example, the flow at this position directly affects the flow variations in its downstream segment. The transition variable $q_i^{(t-1)}$ is generated from the latent variable $h_i^{(t-1)}$ as follows:

$$q_i^{(t-1)} = \tanh\left(W_q h_i^{(t-1)} + b_q\right) \tag{1}$$

where W_q and b_q are model parameters used to convert hidden representations into transfer variables, and the superscript $(t-1)$ denotes the physical quantity at time step $t-1$.

After collecting the transition variables for all positions, a new recurrent unit structure is established for the target position i, which integrates the transition variables from upstream positions into the computation unit state $c_i^{(t)}$. This can be represented as follows:

$$c_i^{(t)} = f_i^{(t)} \otimes \left(c_i^{(t-1)} + \sum_{(j,i)\in\varepsilon} A_{ji} q_j^{(t-1)}\right) + g_i^{(t)} \otimes \bar{c}_i^{(t)} \tag{2}$$

where $f_i^{(t)}$ and $g_i^{(t)}$ are the forget gate and input gate for position i at time step t, respectively; $\bar{c}_i^{(t)}$ is a new candidate value at the target position i created by a tanh layer at time t, and A_{ji} is the adjacency level between positions j and i. $q_j^{(t-1)}$ from the previous time step is used because of the time delay when transferring from upstream to downstream positions.

After obtaining the cell state, the hidden representation $h_i^{(t)}$ can be computed. Finally, the flow prediction $\hat{y}_i^{(t)}$ for the target position can be generated from the hidden representation, which corresponds to the intermediate state in the encoder.

$$\hat{y}_i^{(t)} = W_y h_i^{(t)} + b_y \tag{3}$$

where W_y and b_y are model parameters.

After applying this recurrent process to all time steps, a loss L_{RGrN} can be defined using the real observations $\boldsymbol{Y}=\{y_i^{(t)}\}$ available at certain time steps and segments, as shown below:

$$L_{\mathrm{RGrN}} = \frac{1}{|\boldsymbol{Y}|} \sum_{|(i,t)\ |\ y_i^{(t)}\in\boldsymbol{Y}\}} \left(y_i^{(t)} - \hat{y}_i^{(t)}\right)^2 \tag{4}$$

2.1.2. Decoder (MCLSTM)

The original LSTM introduced a memory cell into the RNN. In this model, the value of the memory cell at time t is represented as $\boldsymbol{c}^{(t)}$, and its recurrence can be formulated as

$$\boldsymbol{c}^{(t)} = \boldsymbol{c}^{(t-1)} + f(\boldsymbol{x}^{(t)}, \boldsymbol{h}^{(t-1)}) \tag{5}$$

where $\boldsymbol{x}^{(t)}$ and $\boldsymbol{h}^{(t)}$ represent the forward input and recurrent input, respectively; and f is a function that calculates the increment of the memory cell.

MCLSTM (Fig. 4) modifies the recursion to ensure balanced inputs of conservation quantities, with the key idea being to use the memory cell from LSTM as a mass accumulator.

The MCLSTM distinguishes between conservation

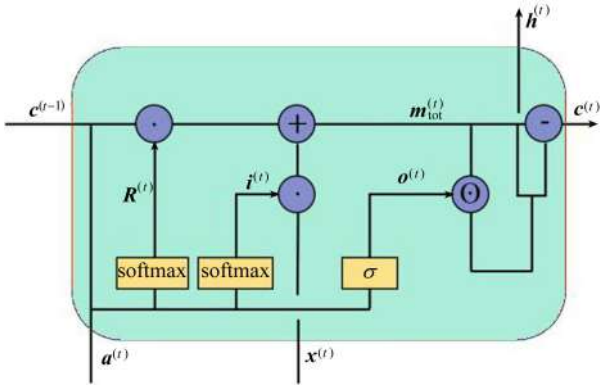


Fig. 4. Detailed representation of major operations in MCLSTM structure

quantity inputs (such as flow and rainfall) $\boldsymbol{x}$ and auxiliary inputs (such as minimum and maximum temperature, shortwave radiation, and evapotranspiration) $\boldsymbol{a}$ in the encoder state obtained from the previous section.

The forward propagation of MCLSTM at time step t can be specified as follows:

$$\boldsymbol{m}_{\text{tot}}^{(t)} = \boldsymbol{R}^{(t)} \cdot \boldsymbol{c}^{t-1} + \boldsymbol{i}^{(t)} \cdot \boldsymbol{x}^{(t)} \tag{6}$$

$$\boldsymbol{c}^{(t)} = (\boldsymbol{I} - \boldsymbol{o}^{(t)}) \odot \boldsymbol{m}_{\text{tot}}^{(t)} \tag{7}$$

$$\boldsymbol{h}^{(t)} = \boldsymbol{o}^{(t)} \odot \boldsymbol{m}_{\text{tot}}^{(t)} \tag{8}$$

where $\boldsymbol{i}^{(t)}$ and $\boldsymbol{o}^{(t)}$ are the input and output gates, respectively; $\boldsymbol{R}^{(t)}$ is a positive left stochastic matrix ($\boldsymbol{I}^{\mathrm{T}} \cdot \boldsymbol{R}^{(t)} = \boldsymbol{I}^{\mathrm{T}}$), which is used for redistributing the conservation quantity in the mass accumulator, $\boldsymbol{I}$ is the unit matrix; and $\boldsymbol{m}_{\text{tot}}^{(t)}$ is the total conservation quantity. The current conservation quantity in the system is stored in $\boldsymbol{c}^{(t)}$. Finally, $\boldsymbol{h}^{(t)}$ is the conservation quantity leaving the system.

The gate computations for MCLSTM at time step t are as follows:

$$\boldsymbol{i}^{(t)} = \text{softmax}\left(\boldsymbol{W}_i \cdot \boldsymbol{a}^{(t)} + \boldsymbol{U}_i \cdot \frac{\boldsymbol{c}^{(t-1)}}{\| \boldsymbol{c}^{(t-1)} \|_1} + \boldsymbol{b}_i\right) \tag{9}$$

$$\boldsymbol{o}^{(t)} = \sigma\left(\boldsymbol{W}_o \cdot \boldsymbol{a}^{(t)} + \boldsymbol{U}_o \cdot \frac{\boldsymbol{c}^{(t-1)}}{\| \boldsymbol{c}^{(t-1)} \|_1} + \boldsymbol{b}_o\right) \tag{10}$$

$$\boldsymbol{R}^{(t)} = \text{softmax}\left(\boldsymbol{B}_r\right) \tag{11}$$

where softmax $(\cdot)$ is a column-wise softmax operator; σ represents the logistic sigmoid function; $\boldsymbol{U}_i$ and $\boldsymbol{U}_o$ are the weight matrix of input gate and output gate, respectively; and $\boldsymbol{W}_i$, $\boldsymbol{b}_i$, $\boldsymbol{W}_o$, $\boldsymbol{b}_o$, and $\boldsymbol{B}_r$ are learnable model parameters.

The prediction of the redistribution matrix for each sample and time step follows a similar procedure as the gate computations:

$$\boldsymbol{R}^{(t)} = \text{softmax}\left(\boldsymbol{W}_r \cdot \boldsymbol{a}^{(t)} + \boldsymbol{U}_r \cdot \frac{\boldsymbol{c}^{(t-1)}}{\| \boldsymbol{c}^{(t-1)} \|_1} + \boldsymbol{B}_r\right) \tag{12}$$

where $\boldsymbol{W}_r$ and $\boldsymbol{U}_r$ are weight tensors, and their product produces a $K \times K$ matrix.

The original LSTM model has an explicit input-state-output structure that is recurrent in time, but it does not follow the principle of mass balance and lacks a unit to store conservation quantities in its internal state. The previous approach addressed mass conservation by replacing the storage unit with a mass accumulator, leading to overfitting in practical processes.

To address the aforementioned issues, this study enhances conservation in MCLSTM through two operations. First, special activation functions are used in certain gates to ensure mass conservation between the inputs and the previous cell state. Second, mass outflow is subtracted from the cell state. An important property of the special activation functions is that the sum of all elements is equal to 1. In practice, other standard activation functions (e. g., sigmoid, ReLU, as shown in Fig. 5) can also be used as long as the activations are normalized.

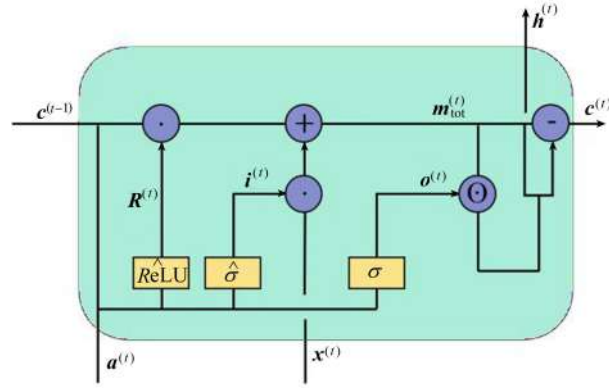


Fig. 5. MCLSTM structure with improved activation functions

2.2. *RGrN-MCLSTM flood flow prediction method with optimized input feature*

The input features of the proposed RGrN-MCLSTM flood flow prediction model face challenges of multidimensionality and noise, which can mislead the model training process and affect the accuracy of flow prediction. To address these challenges, this study proposes an IDRS network to optimize the input features of the RGrN-MCLSTM flood flow prediction method. The IDRS network is enhanced from two aspects: the soft-threshold function and the attention mechanism. This aims to enhance the network's ability to suppress noise data and be sensitive to important features, thereby obtaining high-quality inputs and improving the prediction accuracy of the model.

The overall structure of the deep residual shrinkage network is illustrated in Fig. 6, where K is the number of the convolution kernel of the convlution layer, C is the number of channels in the feature graph, W is the width of the feature graph, M is the fully connected layer neurons, X is the input feature, z is the output fea-

ture, and a is a cofficiont between 0 and 1 under the action of the sigmoid function.

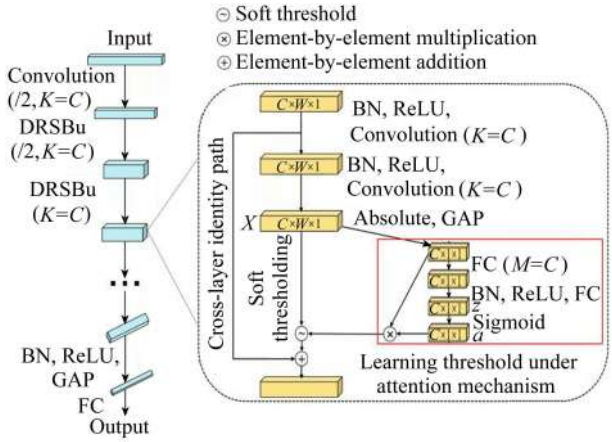


Fig. 6. Building units and overall structure of deep residual shrinkage network

2.2.1. Deep residual network

The original deep residual network is a type of deep convolutional network. Increasing the width and depth of the network can improve its performance. The structural components in the deep residual network are residual units, which are designed based on the concepts of shortcut connections and identity mappings. Fig. 7 shows the structure of the deep residual network.

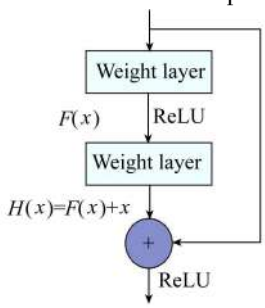


Fig. 7. Structure of deep residual network

In Fig. 7, H denotes the shortcut connection, x represents the input to the residual structure, F is the residual function, and ReLU represents the activation function. If the input to the residual network structure is x_{i+1}, then the output of the entire residual structure is given by Eq. (13):

$$x_{l+1} = f(H(x_l) + F(x_l, w_l)) \tag{13}$$

where w_l is the width of the lth layer.

In this case, the forward propagation of the deep residual network is as follows:

$$x_{l+1} = x_l + F(x_l, w_l) \tag{14}$$

The result of the forward propagation of the deep residual network to the $(l+2)$th output is

$$x_{l+2} = x_l + F(x_{l+1}, w_{l+1}) = x_l + F(x_l, w_l) + F(x_{l+1}, w_{l+1}) \tag{15}$$

The forward propagation results from layer l to layer L are as follows:

$$x_L = x_l + \sum_{i=1}^{L-l} F(x_i, w_i) \tag{16}$$

Derivation of Eq. (16) gives the backpropagation of the residual network as follows:

$$\frac{\partial E}{\partial x_l} = \frac{\partial E}{\partial x_L}\left(1 + \frac{\partial}{\partial x_l}\sum_{i=L}^{L-l} F(x_i, w_i)\right) \tag{17}$$

We can see from Eq. (17) that during backpropagation in the residual network, only the portion before the chain rule is computed. This means that the gradients from the Lth layer can be stably propagated to the lth layer in the process of backpropagation in the residual network.

2.2.2. Variable soft threshold function

The variable soft threshold function (VST) structure is

$$Y = \begin{cases} (1-\beta)x + \beta \cdot \operatorname{sgn}\left(\dfrac{x\, e^{|x|-\lambda} - \lambda}{e^{|x|-\lambda} x \geqslant \lambda}\right) & x \geqslant \lambda \\ 0 & x < \lambda \end{cases} \tag{18}$$

where λ is a threshold that is not less than zero, and $\beta = e^{-\left(\frac{|x|-\lambda}{\lambda^n}\right)}$.

2.2.3. Distance-aware local attention

For multiple adjacent regions, the distance between them is calculated. The distance (D_{ij}) from x_i's adjacent label to x_j is as follows:

$$D_{ij} = \min d_s(v_k, v_j) \quad k \in [i-1, i+1] \tag{19}$$

where v_k and v_j are the kth and jth vertices, respectindy; and $d_s(v_k, v_j)$ is the distance between v_k and v_j.

As shown in Fig. 8, this study explores the use of distance-aware local attention (DLA) mechanism as an alternative to the original global attention mechanism to achieve better performance.

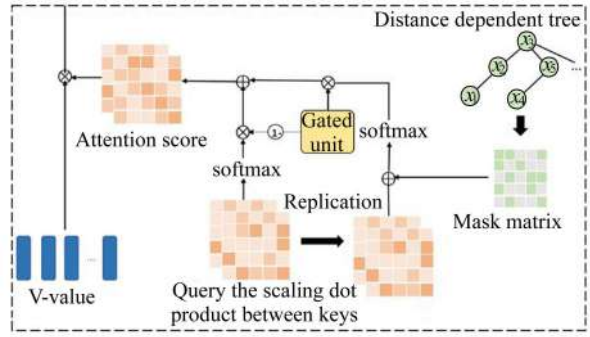


Fig. 8. Schematic diagram of local attention mechanism of distance perception

To determine whether x_j can attend to x_i, a threshold m is used to limit the distance D_{ij}. For simplicity, the element of the masking matrix $\boldsymbol{M}_{\text{loc}}$ can be expressed as

$$M_{\text{loc}ij} = \begin{cases} 0 & D_{ij} \leqslant m \\ -\infty & D_{ij} > m \end{cases} \tag{20}$$

Taking into account the query $\boldsymbol{Q}$ and the keys $\boldsymbol{K}$ projected from the hidden vector $\boldsymbol{H}$, the distance-aware local attention score $\boldsymbol{S}_{\text{loc}}$ is formally defined as

$$\boldsymbol{S}_{\text{loc}} = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\text{T}}}{\sqrt{d}} + \boldsymbol{M}_{\text{loc}}\right) \tag{21}$$

where d represents the dimensionality of the keys.

The attention mechanism in the IDRS network is an aggregation of distance-aware local attention and global attention. A gating unit is used to combine the scores from both global and local attention. The computation method for the gating value g_i of each token x_i is as follows:

$$g_i = \sigma(\boldsymbol{W}_g \boldsymbol{h}_i + b_g) \tag{22}$$

where $\boldsymbol{h}_i$ represents the hidden vector from the upstream region x_i, $\boldsymbol{W}_g$ is a learnable linear transformation, and b_g is the bias term. The attention output $\hat{A}_i$ is computed as the weighted average of the context vector $\boldsymbol{V}$, where the weights are determined by the scores of global and local attention.

$$\hat{A}_i = [g_i \boldsymbol{S}_{\text{loc}i} + (1 - g_i)\boldsymbol{S}_{\text{loc}i}]\boldsymbol{V} \tag{23}$$

Compared to the original structure, the improved self-attention layer in the model has an additional input ($\boldsymbol{M}_{\text{loc}}$) and two trainable parameters ($\boldsymbol{W}_g$ and b_g). This modification enables the attention mechanism to pay more attention to neighboring regions, effectively capturing the spatial dependencies and ensuring better performance.

3. Experiments and Results

3.1. Experimental data and evaluation metrics

In this study, we used the catchment attributes and meteorology for large-sample studies (CAMELS) (Addor et al., 2017) dataset as the basis for research. The CAMELS dataset is a publicly available dataset that contains comprehensive hydrological data for 671 watershed observation points in the United States, recorded by the United States Geological Survey (USGS). The dataset includes daily observed streamflow data and meteorological data starting from 1980. This study mainly used precipitation, maximum temperature, minimum temperature, evapotranspiration, and shortwave radiation data from the meteorological records.

This study employed the commonly used three evaluation metrics for streamflow prediction: mean absolute error (MAE), root mean square error (RMSE), and Nash-Sutciliffe coefficent (NSE), which can be expressed as

$$I_{\text{MAE}} = \frac{\sum_{i=1}^{n} | y_i - q_i |}{n} \tag{24}$$

$$I_{\text{RMSE}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (y_i - q_i)^2} \tag{25}$$

$$I_{\text{NSE}} = 1 - \frac{\sum_{i=1}^{n} (y_i - q_i)^2}{\sum_{i=1}^{n} (q_i - \bar{q})^2} \tag{26}$$

where I_{MAE}, I_{RMSE} and I_{NSE} are the values of MAE, RMSE and NSE, respectively; y_i represents the predicted value of the ith index of the prediction sequence; q_i represents the observed or true value of the ith index of the observed sequence; $\bar{q}$ represents the average value of all the values in the observed sequence; and n is the number of samples.

3.2. Evaluation of flow prediction accuracy for RGrN-MCLSTM model

In order to assess the accuracy of the proposed flow prediction method based on the RGrN-MCLSTM model in data-scarce regions. we selected the observation point with the identifier 02082950 (Neuse River At Kinston) that allows access to information from neighboring catchments. The dataset used includes daily average streamflow data and daily meteorological data from both the upstream and target areas, obtained from January 1, 1981, to December 31, 2010 (a significant amount of missing streamflow data after the year 2000). The experimental setup considered the data in the period from 1981 to 1990 as the training data, 1991 to 2000 as validation data, and 2001 to 2010 as the testing data to validate the effectiveness of the proposed method.

In this study, we investigate the performance of different deep learning algorithms for flow prediction in data-scarce regions from different perspectives. The benchmark algorithms used for comparison are as follows:

(1) CNN: A deep convolutional neural network specifically designed to process data with grid-like structures.

(2) GRU: A variant of LSTM that includes a reset gate and an update gate.

(3) TCN: A temporal convolutional neural network that exploits the parallelism of convolutional architectures without complex gating mechanisms.

(4) stGCN: A network that combines GCN with fully connected networks to extract features from neighboring locations and capture temporal relationships, aiming to explore the role of spatiotemporal feature extraction.

In this experiment, each benchmark algorithm has a different input data format, as shown in Table 1.

The parameter settings for the RGrN-MCLSTM modeling method are as follows: the number of hidden units is 256, the dropout rate is 0.2, the learning rate is 0.001, the batch size is 256, the sequence length is 270, the optimizer is Adam, the loss function is the mean squared error (MSE), and the number of epochs

is 50.

Table 1

Input data features for different algorithms

Model	Upstream streamflow and meteorological data	Streamflow data for target area	Meteorological data for target area
CNN		√	
GRU		√	√
TCN		√	√
stGCN	√	√	√
RGrN-MCLSTM	√	√	√

3.2.1. Comparison of prediction accuracy of different models

To visually illustrate the performance differences of these flow prediction models in data-scarce regions, a line graph of daily streamflow observations from January 1, 2001, to December 31, 2010, is plotted in Fig. 9.

Fig. 9 presents the comparison of predicted results of the RGrN-MCLSTM modeling method and other benchmark algorithms in data-scarce regions. It can be seen from Fig. 9 that CNN, GRU, and TCN show poorer fitting performance, with larger prediction fluctuations during periods of steady streamflow. Additionally, due to the lack of a relational graph module, these algorithms are unable to utilize data from neighboring areas, resulting in lower overall prediction accuracy and highly inaccurate peak predictions. On the other hand, stGCN, RGrN, and RGrN-MCLSTM achieve higher levels of prediction accuracy overall. They exhibit better alignment during periods of steady streamflow and improve peak predictions.

Table 2 shows the performance comparison of different models in predicting streamflow of the observation points with the identifier 02089500 (Ruse River at Kinston). In terms of various evaluation metrics, all algorithms that are capable of extracting temporal features outperform the simple CNN algorithm. Compared to CNN, GRU, and TCN, RGrN-MCLSTM improves the NSE metric by 34.1%, 19.1%, and 15.3%, respectively, reduces the RMSE metric by 22.1%, 18.7%, and 14.3%, respectively, and decreases the MAE metric by 28.1%, 19.6%, and 17.6%, respectively.

Table 2

Comparison and accuracy analysis of different models

Model	RMSE	MAE	NSE
CNN	102.65	33.48	0.642
GRU	99.83	31.25	0.723
TCN	96.17	30.74	0.747
stGCN	88.54	28.28	0.815
RGrN	86.79	27.96	0.834
RGrN-MCLSTM	84.07	26.13	0.861

Based on the aforementioned experiments, it can be concluded that the proposed RGrN-MCLSTM effectively addresses the low-resource issue in data-scarce regions by utilizing data from neighboring locations and incorporating water balance constraints. Its neural network structure is more suitable for flow prediction in data-scarce regions and demonstrates excellent performance in terms of multiple evaluation metrics in the target area, with the highest prediction accuracy.

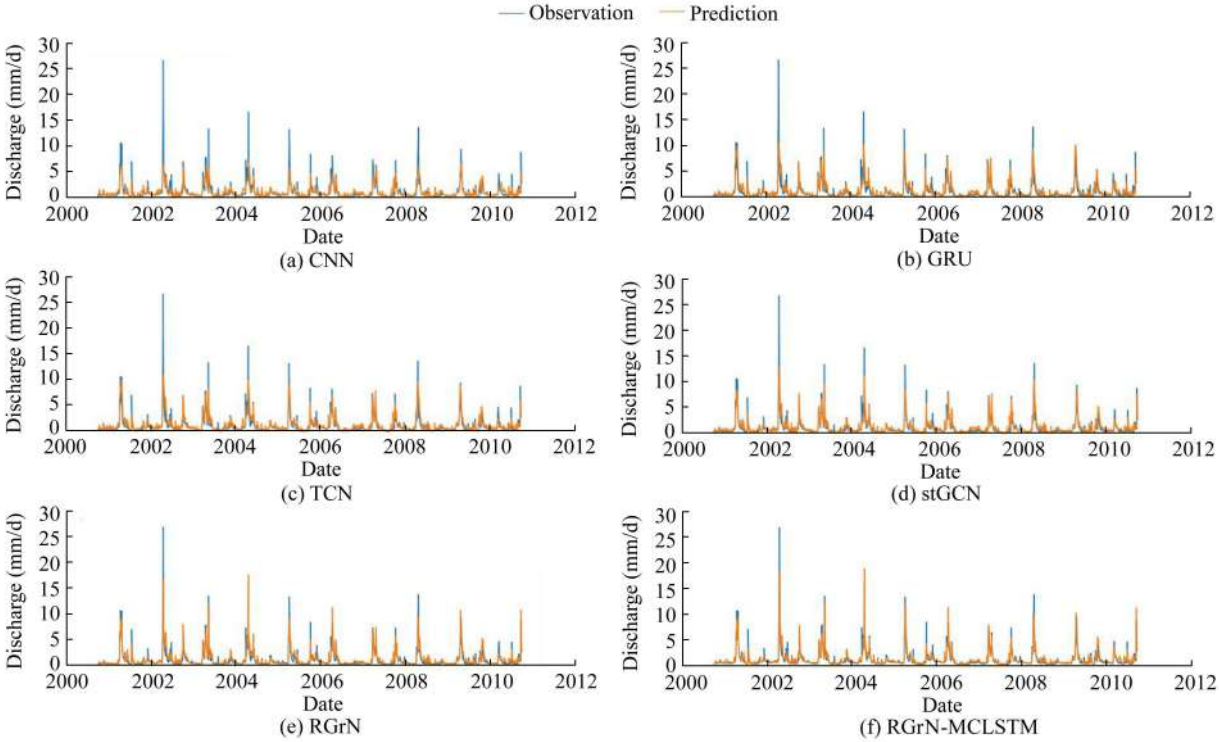


Fig. 9. Comparison of prediction results obtained from different models

3.2.2. Comparison of prediction accuracy at multiple time scales

Fig. 10 presents the prediction results of the RGrN-MCLSTM model at three different time scales: yearly daily average, yearly hourly average, and monthly hourly average. It can be observed that in the data-scarce regions, the prediction accuracy of the model gradually decreases as the time scale becomes finer. This is expected, and even at the finest grain of monthly hourly average, the prediction accuracy of the proposed RGrN-MCLSTM is still acceptable.

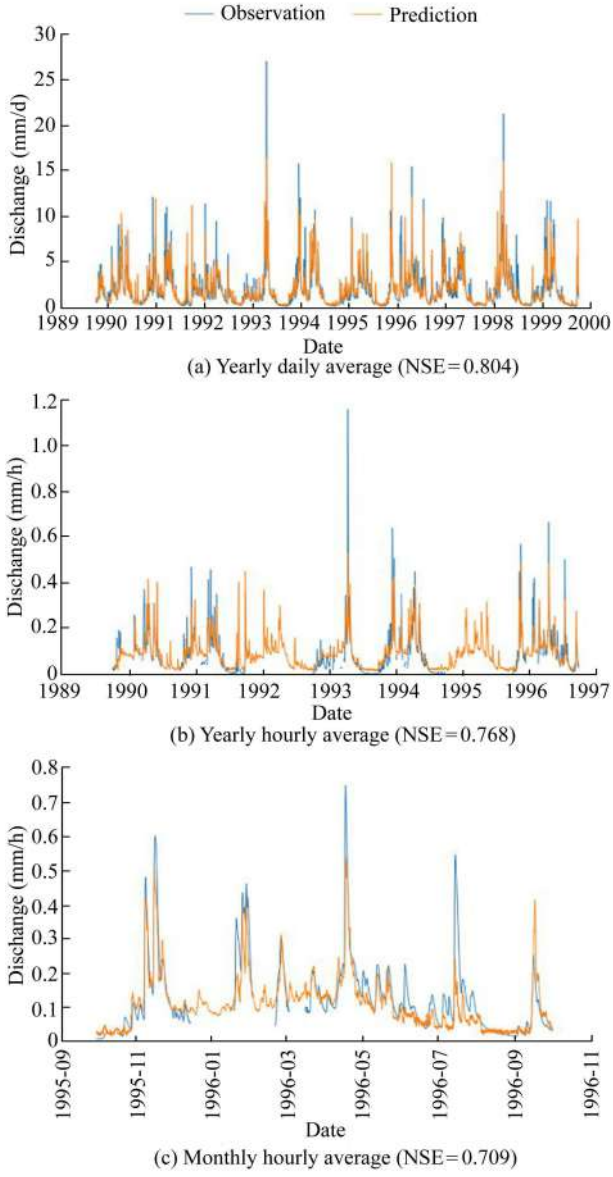


Fig. 10. Comparison of prediction performance of RGrN-MCLSTM model at multiple time scales

3.2.3. Sensitivity analysis

A series of sensitivity experiments are conducted to further validate the effectiveness of different components of the RGrN-MCLSTM flow prediction method. Table 3 shows the results of sensitivity experiments, where the RGrN-MCLSTM model is the complete model proposed in this paper, the GCN-MCLSTM model combines the GCN and MCLSTM, the RNN-MCLSTM model combines the RNN and MCLSTM, the RGrN-LSTM model combines the RGrN and original LSTM without the mass conservation accumulator, and the RGrN-MCGRU model combines the RGrN and gated recurrent unit with the mass conservation accumulator.

Table 3
Results of sensitivity experiments conducted in target basin

Model	RMSE	MAE	NSE
GCN-MCLSTM	88.30	30.88	0.791
RNN-MCLSTM	89.26	33.52	0.768
RGrN-LSTM	86.33	29.17	0.824
RGrN-MCGRU	84.83	27.28	0.846
RGrN-MCLSTM	84.07	26.13	0.861

It can be observed from Table 3 that:

(1) When adjacent location data augmentation is applied, the RGrN-MCLSTM model, which includes the extraction of temporal features, outperforms the GCN-MCLSTM model that lacks the ability to extract temporal features. Effective extraction of temporal features helps the model learn runoff characteristics better and make more accurate predictions.

(2) When both temporal features and adjacent location data augmentation are utilized, the RGrN-MCLSTM model performs better than the RNN-MCLSTM model that only extracts temporal features. Additionally, the performance of RNN-MCLSTM model is lower than that of GCN-MCLSTM model, indicating that data augmentation is more crucial than temporal feature extraction in data-scarce regions.

(3) The performance gap between the RGrN-LSTM model based on the original LSTM and the proposed RGrN-MCLSTM model is considerably reduced in the experiments. This may be due to the lengthy input time series in this study, where the mass conservation accumulator module in the RGrN-MCLSTM model processes all hydrological data, which is not the case for the original LSTM model. As a result, the advantage of the RGrN-MCLSTM model in long sequence prediction is less evident, and it may exhibit better performance in short-term or extreme conditions.

(4) Calculating with LSTM units instead of using a single gating unit maximizes the effectiveness of the mass conservation accumulator and further improves prediction accuracy.

3.3. Validation of model accuracy for flow prediction optimized by IDRS

The denoising effect and feature attention capabilities of the IDRS algorithm are evaluated. The experimental cluster consists of 41 observation points in the i-

dentifier 030202 basin (Neuse River Basin), with each observation point treated as a data-scarce area. The model input consists of daily average flow data and daily meteorological data from the upstream and target areas observed between January 1, 1981, and December 31, 2010 (excluding daily average flow data from the target area after 2000). The experimental setup for training, validation, and testing remains the same.

In order to validate the performance of the IDRS algorithm, five comparative algorithms are used as follows:

(1) ResNet: Residual blocks within the residual network with skip connections.

(2) SENet: Enhancing useful features and attenuating irrelevant features through feature recalibration using global information.

(3) DRS: A deep residual shrinking network composed of ResNet and SENet.

(4) IDRS (no VST): Using the original soft thresholding function module.

(5) IDRS (no DLA): Using the original global attention module.

The input data for the five comparative algorithms in this experiment are the same.

3.3.1. Performance analysis of IDRS algorithm

The performance of the IDRS algorithm is analyzed from two aspects: denoising effect and feature attention.

The predictive performance of the model optimized by IDRS under increasing noise accompanying the input data is shown in Fig. 11.

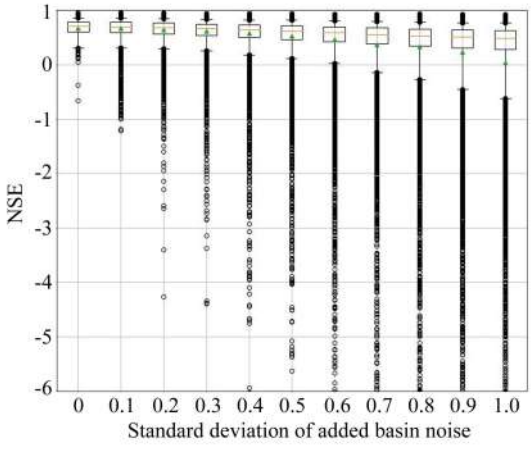


Fig. 11. Impact of noisy input data on model performance

The boxplot in Fig. 11 demonstrates the effect of increasing noise levels in the input data on the model's performance. As expected, the model's performance decreases as the training dataset size increases. When the input data has a small standard deviation of noise (e.g., $\sigma_1 = 0.1$ and $\sigma_1 = 0.2$), the average NSE (green marks) and the median NSE (orange lines) remain relatively stable. The overall median NSE decreases from 0.74 with lower noise levels to 0.56 with the added noise equaling the total variance of the input features ($\sigma_1 = 1$). This confirms that the IDRS algorithm's ability to suppress noise is helpful in improving model performance.

Fig. 12 shows the impact of different types of input in the input layer of the model optimized by IDRS on the predictive performance (deeper blue indicating a higher influence and deeper yellow indicating a lower influence), where each point roughly corresponds to an observation point.

3.3.2. Comparative experiments

To visualize the performance differences of these optimization algorithms in improving rainfall-runoff prediction in data-scarce areas, the cumulative density functions of NSE values for all comparative algorithms at 41 observation points are plotted in Fig. 13.

It can be seen from Fig. 13 that the model achieved the best performance after being optimized by IDRS, with an average NSE value reaching 0.67 in the regions. The performance of IDRS with only the improved variable soft threshold function is similar to the best performance, with an average NSE value of 0.65 in the regions. However, when using the general soft threshold function, the average NSE values are significantly low at 0.62 with the deep residual shrinking network and 0.60 with the original to optimize the model input, respectively. This once again confirms that the variable soft threshold function optimizes the input data by denoising it, thereby greatly improving the performance of the model's predictions.

The average prediction results of the observation cluster with the identifier 030202 (Neuse River Basin, consisting of 41 observation points) obtained from different models are evaluated and the results are shown in Table 4.

Table 4
Comparison and accuracy analysis of different models

Model	RMSE	MAE	NSE
ResNet	82.69	27.89	0.859
SENet	81.95	26.50	0.861
DRS	80.06	25.13	0.865
IDRS(no VST)	80.94	25.86	0.868
IDRS(no DLA)	80.33	24.07	0.870
IDSR	79.16	24.53	0.872

It can be seen from Table 4 that:

(1) The SENet based on attention mechanism is significantly better than the ResNet based on residual networks. From a theoretical perspective, enhancing the model's sensitivity to data features can effectively optimize the input features.

(2) Combining attention mechanism and soft threshold function improves the performance of the re-

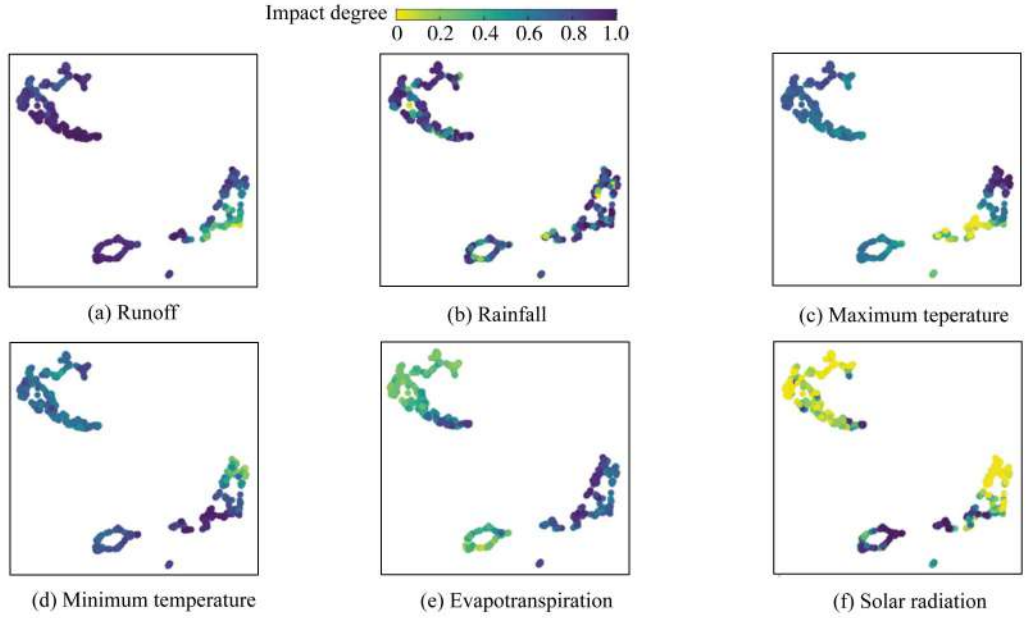


Fig. 12. Impact of different input features in different regions on predictive performance

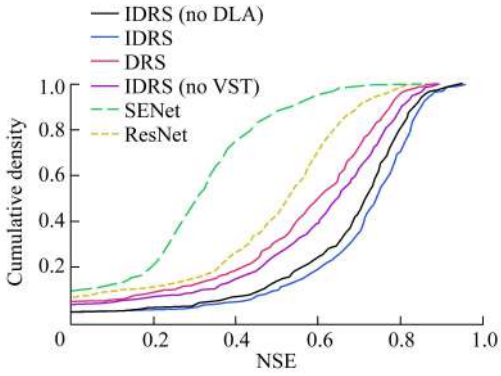


Fig. 13. Comparison of prediction results of optimized models of each algorithm

sidual network, allowing better extraction of model input features. The combination of these two modules enhances the model's ability to extract important features while eliminating the negative impact of redundant information in feature learning.

(3) The optimization effect of the improved deep residual shrinking network based on the VST is more significant than the optimization effect based on the DLA. This may indicate that excellent noise reduction in the data has greater advantages in feature learning compared to accurately assigning attention weights. However, the difference between the two approaches is relatively small.

(4) Combining VST and DLA to improve the deep residual shrinking network can better optimize the model's input, thereby improving the accuracy of the model's predictions. This demonstrates the effectiveness of combining these two techniques to enhance the model's performance.

Overall, these observations highlight the importance of feature optimization, noise reduction, and attention mechanism in improving the accuracy of model predictions.

3.3.3. Performance analysis of model with optimized input features by IDRS

The performance of IDRS is evaluated from two aspects: flow hydrograph and evaluation metrics. The relevant results are shown in Fig. 14 and Table 5.

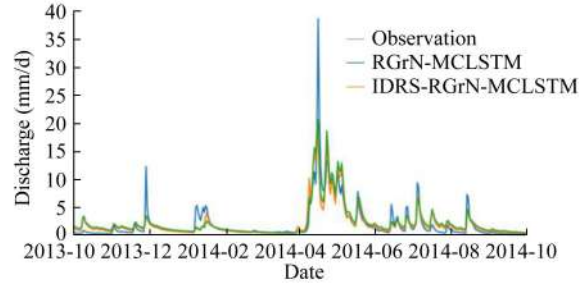


Fig. 14. Prediction effect of model with IDRS feature optimization module

Table 5
Comparison and accuracy analysis of different models

Model	RMSE	MAE	NSE
RGrN-MCLSTM	83.42	27.19	0.854
IDRS-RGrN-MCLSTM	79.16	24.53	0.872

It can be observed from Fig. 14 that both RGrN-MCLSTM and IDRS-optimized models achieve reasonably accurate forecasting results. In comparison to the unoptimized model, the flow hydrograph of the IDRS-optimized model aligns more closely with the actual observed values. However, there is still room for improvement in peak flow capture ability.

It can be observed from Table 5 that the IDRS-RGrN-MCLSTM model, using IDRS-optimized input pa-

rameters proposed in this study, exhibits lower prediction errors. Specifically, the RMSE and MAE values are reduced by 5.4% and 10.8%, respectively, compared to the RGrN-MCLSTM model. The NSE value of the IDRS-RGrN-MCLSTM model is higher, indicating that the noise reduction in the input data and the increased sensitivity to important data inputs improve the predictive performance of the model. These experiments verify the effectiveness of IDRS in optimizing model input from multiple perspectives.

4. Conclusions

This paper proposes a flow prediction method, IDRS-RGrN-MCLSTM, which utilizes IDRS to optimize the input features. It addresses the issue of insufficient observed data in areas with scarce data for flood flow prediction models, as well as the presence of noise and redundant features in the multidimensional input data of the prediction models. The proposed method achieves higher accuracy compared to other deep learning-based flood flow prediction methods. Furthermore, the data augmentation, incorporation of prior knowledge, and feature optimization methods employed in this model are general and can be applicable to other fields, rendering them insightful. Subsequently, it is important to focus on developing pre-trained models that can be fine-tuned for various river basins globally, making it easier to transfer this method to data-scarce areas. Additionally, applying this method to different datasets can verify its universality across different data dimensions.

References

Addor, N., Newman, A. J., Mizukami, N., Clark, M. P., 2017. The CAMELS data set: Catchment attributes and meteorology for large-sample studies. Hydrol. Earth Syst. Sci. 21 (10): 5293-5313. https://doi.org/10.5194/hess-21-5293-2017.

Ben-Haim, Z., Anisimov, V., Yonas, A., 2019. Inundation modeling in data scarce regions. In: the 33rd Conference and Workshop on Neural Information Processing Systems (NeurIPS). NeurIPS, Vancouver, pp. 1910-1916.

Beven, K., 2020. Deep learning, hydrological processes and the uniqueness of place. Hydrol. Process. 34(16), 3608-3613. https://doi.org/10.1002/hyp.13805.

Hu, H., Sui, H., Hu, Q., Zhang, Y., Hu, Z., Ma, N., 2022a. Runoff forecast model based on graph attention network and dual-stage attention mechanism. J. Comput. Appl. 42(5), 1607-1615. https://doi.org/10.11772/j.issn.1001-9081.2021050829 (in Chinese).

Hu, Q., Zhu, Y., Hu, H., Guan, Z., Qian, Z., Yang, A., 2022b. Multiple kernel learning with maximum inundation extent from MODIS imagery for spatial prediction of flood susceptibility. Water Resour. Manag. 36, 55-73. https://doi.org/10.1007/s11269-021-03010-2.

Li, H., Wang, P., Hu, H., Su, Z., Li, L., Yue, Z., 2023. Data-driven reliability assessment with scarce samples considering multidimensional dependence. Probab. Eng. Mech. 72, 103440. https://doi.org/10.1016/j.probengmech.2023.103440.

Karpatne, A., Atluri, G., Faghmous, J. H., Steinbach, M., Banerjee, A., Ganguly, A., Shekhar, S., Samatova, N., Kumar, V., 2017. Theory-guided data science: A new paradigm for scientific discovery from data. IEEE Trans. Knowl. Data Eng. 2017, 29(10), 2318-2331. https://doi.org/10.1109/TKDE.2017.2720168.

Kratzert, F., Klotz, D., Brenner, C., Schulz, K., Herrnegger, M., 2018a. Rainfall-runoff modelling using long short-term memory (LSTM) networks. Hydrol. Earth Syst. Sci. 22 (11), 6005-6022. https://doi.org/10.5194/hess-22-6005-2018.

Kratzert, F., Klotz, D., Herrnegger, M., Hochreiter, S., 2018b. A glimpse into the unobserved: Runoff simulation for ungauged catchments with LSTMs. In: the 32rd Conference and Workshop on Neural Information Processing Systems (NeurIPS). NeurIPS, Montreal, pp. 1468-1473.

Kratzert, F., Klotz, D., Herrnegger, M., Sampson, A. K., Hochreiter, S., Nearing, G. S., 2019. Toward improved predictions in ungauged basins: Exploiting the power of machine learning. Water Resour. Res. 55(12), 11344-11354. https://doi.org/10.1029/2019WR026065.

Momoi, M., Kotsuki, S., Kikuchi, R., Watanabe, S., Yamada, M., Abe, S., 2023. Emulating rainfall-runoff-inundation model using deep neural network with dimensionality reduction. Artific. Intellig. Earth Syst. 2(1), 1-25. https://doi.org/10.1175/AIES-D-22-0036.1.

Ravindranath, A., Devineni, N., Kolesar, P., 2016. An environmental perspective on the water management policies of the Upper Delaware River Basin. Water Policy 18(6), 1399-1419. https://doi.org/10.2166/wp.2016.166.

Sanyal, J., 2023. Flood inundation modelling in data-sparse flatlands: Challenges and prospects. In: Islam, A., Shit, P. K., Datta, D. K., Islam, M. S., Roy, S., Ghosh, S., Das, B. C., eds., Floods in the Ganga-Brahmaputra-Meghna Delta. Springer, Gewerbestrasse, pp. 19-35.

Sellami, H., La Jeunesse, I., Benabdallah, S., Baghdadi, N., Vanclooster, M., 2014. Uncertainty analysis in model parameters regionalization: A case study involving the SWAT model in Mediterranean catchments (Southern France). Hydrol. Earth Syst. Sci. 18(6), 2393-2413. https://doi.org/10.5194/hess-18-2393-2014.

Shorten, C., Khoshgoftaar. T. M., 2019. A survey on image data augmentation for deep learning. J. Big Data 6, 60. https://doi.org/10.1186/s40537-019-0197-0.

Wang, N., Zhang, D., Chang, H., Li, H., 2020. Deep learning of subsurface flow via theory-guided neural network. J. Hydrol. 584, 124700. https://doi.org/10.1016/j.jhydrol.2020.124700.