IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 5, 2022, accepted May 21, 2022, date of publication May 27, 2022, date of current version June 6, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3178521

Improved Transformer Model for Enhanced Monthly Streamflow Predictions of the Yangtze River

CHUANFENG LIU[1], DARONG LIU[1,3], AND LIN MU[2,3]
[1]College of Marine Science and Technology, China University of Geosciences, Wuhan 430074, China
[2]College of Life Science and Oceanography, Shenzhen University, Shenzhen 518061, China
[3]Southern Marine Science and Engineering Guangdong Laboratory (Guangzhou), Guangzhou 511458, China

Corresponding author: Darong Liu (lidr1169@cug.edu.cn)

This work was supported in part by the Key Special Project for Introduced Talents Team of Southern Marine Science and Engineering Guangdong Laboratory (Guangzhou) under Grant GML2019ZD0604, the National Natural Science Foundation of China under Grant U2006210, and in part by the Shenzhen Fundamental Research Program under Grant JCYJ20200109110220482.

ABSTRACT Over the past few decades, floods have severely damaged production and daily life, causing enormous economic losses. Streamflow forecasts prepare us to fight floods ahead of time and mitigate the disasters arising from them. Streamflow forecasting demands a high-capacity model that can make precise long-term predictions. Traditional physics-based hydrological models can only make short-term predictions for streamflow, while current machine learning methods can only obtain acceptable results in normal years without floods. Previous studies have demonstrated a close relation between El Niño-Southern Oscillation (ENSO) and the streamflow of the Yangtze River. However, traditional models, holding the encoder–decoder architecture, only have one encoder block that can not support bivariate time series forecasting. In this study, a transformer-based double-encoder-enabled model was proposed, called the double-encoder Transformer, with a distinctive characteristic: ''cross-attention'' mechanism that can capture the relation between two time series sequences. Using river flow observation collected by the Yangtze River Water Resources Commission and El Niño-Southern Oscillation (ENSO) observation collected by the National Oceanic and Atmospheric Administration, the model can achieve better performance. By using variational mode decomposition (VMD) technique for preprocessing, the model can make precise long-term predictions for the river flow of the Yangtze River. A monthly prediction of 21 years (from January 1998 to December 2018) was made, and the results indicate that the double-encoder Transformer outperforms mainstream time series models.

INDEX TERMS Streamflow prediction, Yangtze River, deep learning, transformer, variational modal decomposition, flood forecasts.

I. INTRODUCTION

The Yangtze River has seen numerous floods in its history. Affected by various factors, including the El Nino weather pattern [1]–[3], the river flow time series are complex and nonlinear [4]. Flow prediction is a practical problem and has been drawing an increasing amount of attention. Many studies have been done to predict the river flow for months [5], [6] or even years in advance [7], [8]. Streamflow predictions help in the ability to fight floods in advance and help local administrators make better decisions and mitigate disasters [9].

The associate editor coordinating the review of this manuscript and approving it for publication was Ehab Elsayed Elattar.

Over the past few decades, many numerical and machine learning methods have been developed to predict streamflow.

Numerical prediction models [10]–[12] can simulate the interactions of various physical processes, such as atmospheric circulation and the evolution of long-term weather in the physical world [13]. Numerical models can be analogous to conducting a particular physics experiment as a way to achieve satisfactory results in short-term forecasting [14]. The soil and water assessment tool (SWAT) [10] was proposed as a way to predict the effects of land management on water, sediment, and chemicals in a large watershed with complex and varied soil types, land-use patterns, and management practices. SWAT is a distributed watershed

hydrologic model based on the geographic information system (GIS). SWAT primarily uses the space-based information from remote sensing and geographical information systems to simulate various hydrological physical and chemical processes [10]. However, hydrologic numerical models have some severe drawbacks: 1. Large amounts of data with high accuracy are required; 2. long-term forecasting is less accurate; and 3. numerical methods are computationally intensive, requiring vast computing resources. Statistical prediction methods generally belong to traditional machine learning. Many statistical models have been employed in predicting streamflow [15]–[17]. Support vector machines (SVMs) [18] was designed to be a strict theoretical and mathematical basis classifier. In 2016, Shuang Zhu *et al*. used SVMs to predict the upper reaches of the Yangtze River; here, the R^2 could reach 0.87 in a single-year monthly forecast [19]. Statistical hydrological models can perform well in normal years but perform poorly in flood years because the organizational structure of its parameters limits the complexity of the model, hence not being able to predict the anomalies precisely.

Traditional deep learning models for extracting features that can be used as prediction models. Artificial neural network (ANN), convolutional neural network (CNN) [20], and recurrent neural network(RNN) [21] are promising methods for predicting river flow. An ANN can approximate the unknown and nonlinear functions with arbitrary precision, namely universal function approximators [22]. ANN models have been used to predict streamflow [23]–[25]. However, the ANN has two drawbacks: 1. With the increase of sequence length, the amount of trainable parameters increases sharply. 2. An ANN cannot capture sequence information (e.g., position and order). It is generally accepted that an ANN is not suitable for processing time series. CNNs and RNNs were designed to overcome the limitations of ANNs. Previous studies have found CNNs and RNNs to be more accurate than other models for dealing with time series. Shun-Yao Shih *et al*. used a CNN for multivariate time series forecasting [26]. Shaojie Bai *et al*. proposed a new CNN architecture named a temporal convolutional network (TCN) [27] and obtained good results on various time series datasets. An RNN was specifically designed to deal with time series [21]. However, the vanilla RNN architecture is full of problems that cannot be used directly. Based on RNN architecture, long short-term memory (LSTM) [28] has been proposed, as a way to alleviate problems within an RNN. Currently, LSTM is widely used in hydrologic prediction tasks [13], [29], [30]. In 2020, Liu *et al*. used LSTM to predict the middle reaches of the Yangtze River; here, in flood years, the R^2 monthly prediction could reach 0.89 [31]. However, there are also some shortcomings in CNNs and RNNs: 1. RNNs are not computationally parallel, making them slow and time-consuming. 2. CNNs will lose critical sequence information while processing with time series, decreasing their accuracy. 3. CNNs and RNNs cannot capture the long-term dependence effectively.

The attention mechanism was first proposed by Bengio *et al*. in 2014 [32], and since then, it has been widely applied in various fields of deep learning. With the attention mechanism, models can ignore low-value information and focus on high-value information. The attention layer can capture the long-term dependency by computing the ''attention'' between all pairs of points. The process does not need to be sequential, so it is computationally parallel. The attention mechanism can also be used as a feature extractor, outperforming CNN and RNN architectures. Based on an attention mechanism, Google proposed a new model called Transformer [33]. Transformer abandoned the traditional CNNs and RNNs, and the entire network structure is completely composed of the attention mechanism. Transformer was initially designed for natural language processing (NLP). Still, currently, many studies have indicated that Transformer is faster and stronger than CNNs and RNNs in dealing with time series [34]–[36]. However, like the previous models, Transformer also has trouble in predicting flood peak discharge.

There are three deficiencies in the previous works: 1. They can not forecast the flood peak discharge precisely. The R^2 in flood years has been lower than 0.9. 2. They cannot make long-term stable predictions with high accuracy. When making long-term predictions (e.g., more than a decade of predictions), the average R^2 may only be around 0.85. 3. Previous models do not have structures for multidimensional data processing and combine all the dimensions into feature vectors as input to make multidimensional predictions for river flow, limiting the models' capability. The previous multidimensional data prediction models all use this method of feature vectors [26], [37], [38]. In this study, a restructured Transformer model combined with VMD is proposed, namely double-encoder Transformer. Variational mode decomposition (VMD) is widely used to preprocess complex and nonlinear data [39]. It can decompose a signal into several different modes to transform the signal in the time domain into the frequency domain. The inherent features of the original signal can be better reflected, making the model fit well, no matter in normal or flood years. The restructured Transformer is still an encoder–decoder architecture with two encoders and one decoder. The structure of the multiencoder can support multivariate prediction. The inputs of the two encoders are observed river flow data and observed ENSO data, respectively. To better obtain the correlation between El Nino and flow, the decoder receives the outputs of the two encoders as inputs and then learns and computes the correlation between them.

The contributions of the present paper are as follows:

- The double-encoder Transformer is proposed to enhance the prediction capability successfully, which significantly improves the flow prediction accuracy of flood years with an R^2 higher than 0.95.
- A reliable long-term (21 years) flow prediction of the Yangtze River had been performed, achieving high accuracy.

- The "cross-attention" mechanism is proposed to solve the bivariate prediction problem, which validates the attention-based model's potential value for making a multivariate prediction.
- A heuristic method is applied to determine the K value in the VMD algorithm.

II. METHODOLOGY

The specific research methods can be divided into data preprocessing and deep learning. The critical step of data preprocessing is the VMD algorithm. In section II-A, starting with data preprocessing, the focus is on the details of the VMD algorithm. Deep learning techniques generally develop an encoder–decoder paradigm [38], mainly using a CNN and RNN and their variants. The double-encoder Transformer holds the encoder–decoder architecture with attention blocks. In section II-B, the principles and processes of attention operation are presented and described. Section II-C provides the core part of the work. The model's architecture and network layers will be presented in detail. Please refer to Fig. 1 for an overview and the sections for more information.

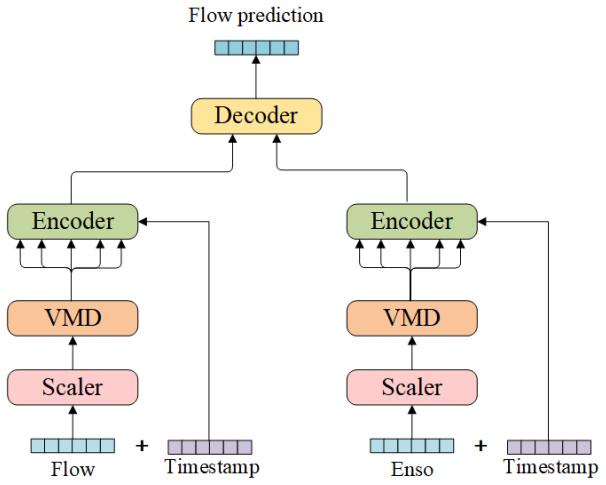


FIGURE 1. The overview of the work.

A. DATA PREPROCESSING

This section presents the process of preprocessing original observed flow data and ENSO data. There are three steps: 1. data normalization; 2. Adding time stamps; 3. applying the VMD algorithm to decompose the signal. The key to the third step is the appropriate amount of frequency components decomposed by the VMD. Here, we will demonstrate how to determine the right amount. There are two steps before employing the VMD.

1. Data normalization. The flow of the Yangtze River changes dramatically, ranging from 5000 m^3/month to nearly 70000 m^3/month. Suppose the dimensional (scaler) difference of the data is too large. In this case, the data with a large dimension (scale) will take the leading position, which will reduce the accuracy of the model. In addition, data with a large dimension will have a longer axis when updating the model using a gradient descent, which means that the convergence of the model will be slow [40]. In this study, we used min–max scaling to perform data normalization.

2. Adding time stamps. For time series, it is essential to add time information while training [41], [42]. Classic Transformer only encodes the input data with position and does not contain more fine-grained time information such as year, month, day, or hour [38]. The dataset used has a monthly dimension, so we added time stamps of years and months to each piece of data. Like positional encoding [33], these time stamps would be embedded into higher dimensions and added to the input data as sequential information.

VMD works efficiently with nonlinear and nonstationary data like streamflow by decomposing a signal into different modes of distinct spectral bands. In the original description, a model is defined as a signal whose number of local extrema and zero-crossings differ almost by one. Now, the definition is slightly changed into the so-called intrinsic mode function (IMF) [43], [44]. The complete VMD algorithm [39] can be summarized in Algorithm 1, where $u_k := \{u_1, u_2, u_3, \ldots, u_k\}$ are the shorthand notations for the set of all modes (IMFs) and $w_k := \{w_1, w_2, w_3, \ldots, w_k\}$ are the shorthand notations for their center frequencies, respectively. The role of Lagrangian multipliers λ is to enforce the constraint. The goal of VMD is to decompose the original signal into subsignals u_k.

VMD has proven to be a better algorithm than empirical mode decomposition (EMD) [45]. VMD is much more

Algorithm 1 Complete Optimization of VMD [39]

Initialize $\{\widehat{u}_k^1\}$, $\{\omega_k^1\}$, $\widehat{\lambda}^1$,$n = 0$

repeat

$n = n + 1$

for $k = 1 : K$ ***do***

Update $\widehat{u}_k$ for all $\omega \geqslant 0$:

$$\hat{u}_k^{n+1}(\omega) \leftarrow \frac{\hat{f}(\omega) - \sum_{i<k} \hat{u}_i^{n+1}(\omega) - \sum_{i>k} \hat{u}_i^n(\omega) + \frac{\hat{\lambda}^n(\omega)}{2}}{1 + 2\alpha\left(\omega - \omega_k^n\right)^2}$$

Update ω_k:

$$\omega_k^{n+1} \leftarrow \frac{\int_0^\infty \omega \left|\hat{u}_k^{n+1}(\omega)\right|^2 d\omega}{\int_0^\infty \left|\hat{u}_k^{n+1}(\omega)\right|^2 d\omega}$$

end for

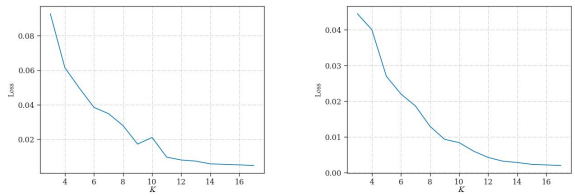
Dual ascent for all $\omega \geqslant 0$:

$$\hat{\lambda}^{n+1}(\omega) \leftarrow \hat{\lambda}^n(\omega) + \tau\left(\hat{f}(\omega) - \sum_k \hat{u}_k^{n+1}(\omega)\right)$$

until convergence:

$$\sum_k \left\|\hat{u}_k^{n+1} - \hat{u}_k^n\right\|_2^2 / \left\|\hat{u}_k^n\right\|_2^2 < \epsilon$$

robust to sampling and noise and supported by mathematical theory [39]. However, the amount of IMF decomposed by VMD is not fixed, and the effect of VMD is mainly affected by this amount. K will be used as a shorthand notation for this amount in the following presentation. Applying VMD to decompose the original signal will produce losses. When K is too small, a lot of important information in the original signal will be filtered, affecting the accuracy of subsequent predictions. The loss can be observed by decomposing the original signal into IMFs and then recomposing them and comparing them with the original signal. The loss can be defined as the difference between the recomposed signal and the original signal. To measure the loss, the coefficient of determination (R^2) is introduced. A R^2 between 0 and 1 can reflect the difference between the two distributions, and the smaller the R^2 the greater the difference. The loss can be measured by R^2 :$Loss = 1 - R^2$. Fig. 2(a) and Fig. 2(b) show the influence of K to loss on the flow and ENSO dataset, respectively. The larger K is, the smaller the loss. However, When K is too large, it will cause a few problems: 1. There will be gaps in high-frequency IMF, and the high-frequency signal becomes intermittent. 2. The center frequencies of the adjacent IMFs will be close to each other, resulting in modal repetition and extra noise. 3. More IMFs mean more computing resources and more time.



(a) Flow (b) ENSO

FIGURE 2. The correlation between K and Loss on flow and ENSO datasets.

In this study, a heuristic approach was adopted to select the appropriate K value. We apply Hilbert transform (HT) [46] to each IMF. The HT can be described as (1):

$$\begin{aligned} H[x(t)] &= \hat{x}(t) \\ &= x(t) * \frac{1}{\pi t} \\ &= \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{x(\tau)}{t-\tau} d\tau \end{aligned} \tag{1}$$

where π is the Pi. $*$ and τ are the convolution operator and the integration variable, respectively. The integral is considered to be Cauchy principal value, which avoids singularities at $\tau = t$ and $\tau = \pm\infty$. The instantaneous amplitude $A(t)$ and the instantaneous phase $\psi(t)$ can be calculated using $x(t)$ and $\hat{x}$ (t):

$$A(t) = \pm\sqrt{x^2(t) + \hat{x}^2(t)} \tag{2}$$

$$\psi(t) = \arctan \frac{\hat{x}(t)}{x(t)} \tag{3}$$

Then, calculate the instantaneous frequency $IF(t)$ for all points except for two endpoints:

$$\begin{aligned} IF(t) &= \psi'(t) \\ &= \frac{x(t)\hat{x}'(t) - x'(t)\hat{x}(t)}{A^2(t)} \end{aligned} \tag{4}$$

Finally, the mean of all the instantaneous frequencies of each IMF is the central frequency of that IMF:

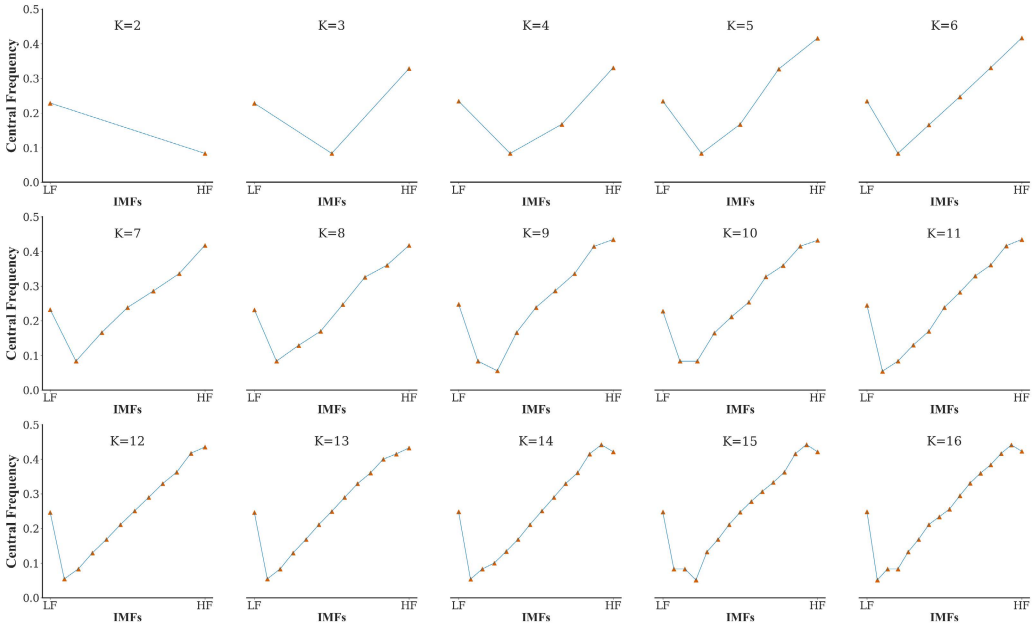
$$CF = \frac{\sum_{t=2}^{n-1} IF(t)}{n-2} \tag{5}$$

Fig. 3(a) and Fig. 3(b) show the central frequency distributions of the two datasets under different K values, respectively. Here, we focus on the high frequencies. When the K value is in a reasonable range, it is meaningful that the higher the frequency, the higher the central frequency. When K is too large, there is an obvious inflection point where the central frequency goes down as K goes up. This inflection point means that K is already large enough. When the K value is too high, the high-frequency part of the signal will break off, and there is no instantaneous frequency at the breakpoint; hence, after averaging, the center frequency will decrease instead. Fig. 4(a) and Fig. 4(b) show the variation of the highest center frequency with K , where the inflection points can be seen. The inflection point means the highest center frequency, and the K should be selected at the point. In this study, $K = 9$ was selected for the flow dataset and $K = 8$ for the ENSO dataset.

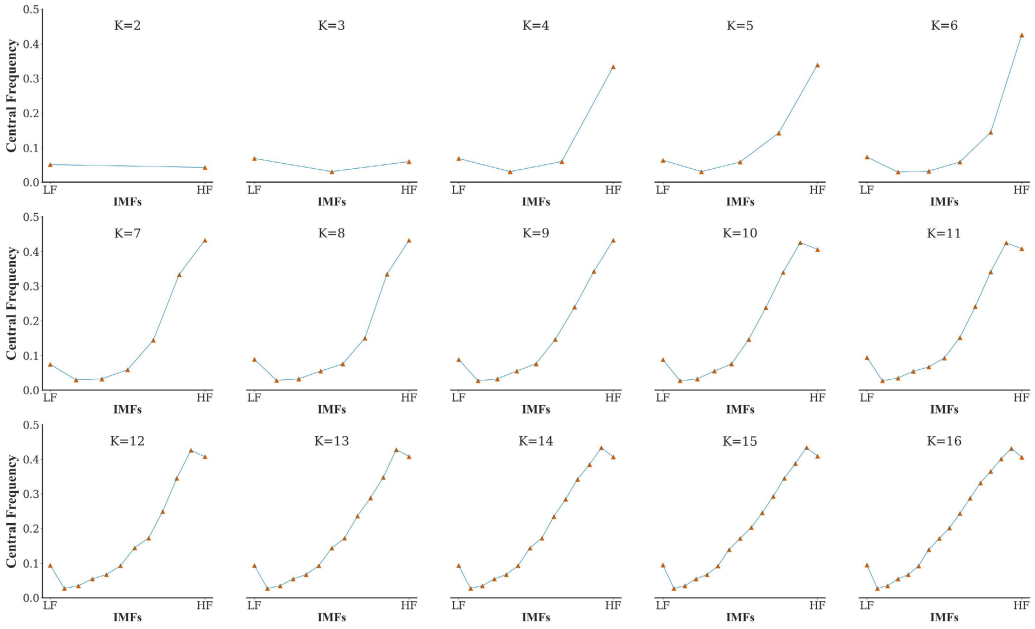
B. ATTENTION MECHANISM

Self-attention is the core part of Transformer. The attention mechanism is a heuristic method that refers to the process of human attention, hence enabling neural networks to focus more on what is important [32]. Therefore Transformer can also be regarded as a feature extractor. As the name suggests, self-attention is about inner attention, which excels in dealing with time series. For a time series sequence, self-attention can capture the correlations between each time step and all other time steps so that the prediction results will be more accurate.

Deep learning techniques mainly develop an encoder–decoder architecture by using RNNs and CNNs. However, both the RNN and CNN models are much slower than attention-based models. Compared with RNN and CNN architectures, the self-attention mechanism is faster [33] and is almost unaffected by the length of the sequence. In general, the longer the sequence, the slower the processing. With this in mind, an experiment was performed to test the speed of the three architectures. Three time series models, including vanilla TCN, vanilla LSTM, and vanilla Transformer, were selected. By recording the time consumed by three models while training, the speed can be measured. We used the flow time series data to train these models, respectively. The length of the input sequence ranges from 12 to 96 and the models were trained for 10 and 50 epochs. This experiment and all the following experiments were performed in the environment given in Table 1. Fig. 5 shows the time



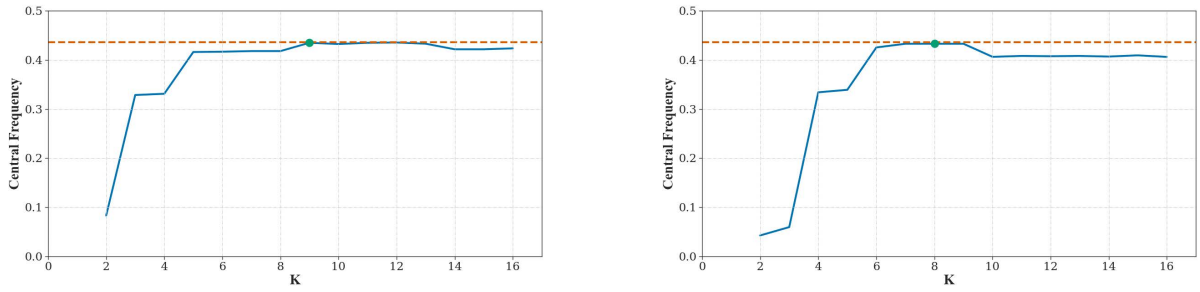
(a) Flow



(b) ENSO

FIGURE 3. The original signal was decomposed into IMFs of K amount (e.g. $K = 9$ means the original signal was decomposed into nine IMFs). The range of K was from 2 to 16. The subfigure shows the central frequency of every IMF under particular K. LF: low frequency; HF: high frequency.

consumed to train the models of three architectures for 10 and 50 epochs under different lengths of the input sequence. It is supposed that the longer the sequence, the slower the processing. For the LSTM, with the length of the sequence



(a) Flow (b) ENSO

FIGURE 4. The figure shows the correlation of K and central frequencies of high-frequency IMFs. Because the IMFs of a high frequency was discontinued, the central frequencies would decrease. It is supposed to select a K value to maximize the center frequency. The green dot in the figure represents the highest center frequency. For the flow dataset, when $K = 9$, there is the highest center frequency; and for the ENSO dataset, $K = 8$.

TABLE 1. The environment.

CPU	GPU	RAM
AMD Ryzen 7 5800H	NVIDIA RTX 3070 8GB	16GB
OS	**Compiler**	**Library**
Windows 10 x64	Python 3.7.8	Pytorch-gpu 1.10.1

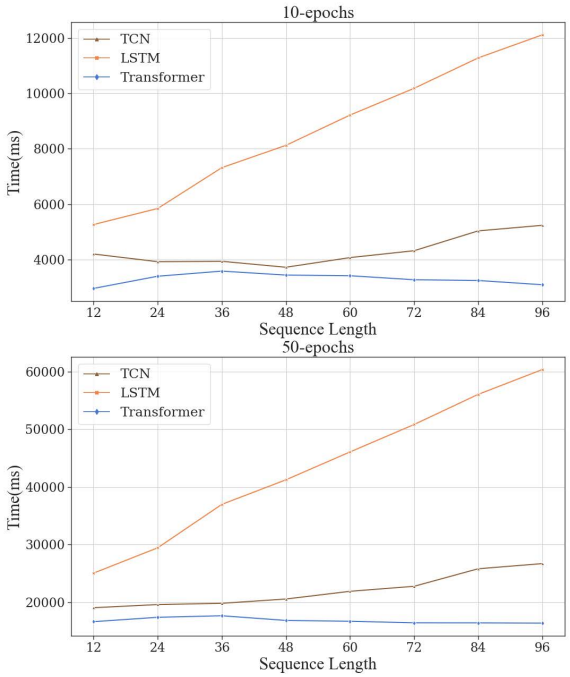


FIGURE 5. Time consumed during training. The models were trained for 10 and 50 epochs. The length of input sequence is (12, 24, 36, 48, 60, 72, 84, 96).

increasing, it consumes significantly more time. However, for CNN and Transformer, the time consumption changes gently and is less affected by the sequence length. The reason for this phenomenon is that the sequence length is not long enough. In the settings of this study, the sequence length ranging from 12 to 96 is reasonable and meaningful. It can be seen that the time consumption of TCN has an obvious increase when the sequence length is longer than 48. As for the Transformer, there will be a significant increase, when the sequence length is longer than 400. In a word, the time consumed by the Transformer is the least of the three and is hardly affected by the sequence length.

On the one hand, the RNN and CNN architectures are slower than the attention architecture. On the other hand, the CNN and RNN models are powerless to capture long-term dependency as efficiently as attention-based models. The reasons for this are as follows:

1. RNN is a linear architecture, and to obtain the correlations between time steps, the operation of each time step depends strictly on the previous steps [21]. RNN is a step-by-step architecture that limits its parallelism. As a result, the RNN is slow. The CNN uses convolution kernels to extract features. The size of the kernels and step size of the convolution affect its speed. Moreover, to better extract features, a multilayer convolutional network is required. Although a CNN is much faster than an RNN, it is still not as fast as self-attention. Self-attention completely abandons RNN structures and instead introduces matrix operations. Matrix operations are parallel, which means that each time step is computed simultaneously instead of one by one, which significantly increases the speed. Because it is parallel, the length of the sequence has little effect on the speed.

2. Although the structure of the ''gates'' mechanism such as LSTM and its variant gated recurrent unit (GRU) [47] alleviate the problem of long-term dependence, RNN is still powerless in dealing with the exceedingly long-term dependence [33]. If the length of inputs is L, the RNN obtains a length of dependence that is shorter than L. The length of dependence that a CNN could get completely depends on the size of convolution kernels, which are generally shorter than L. On the other hand, because the attention value between any two-time steps is computed, it can obtain arbitrarily long-term dependence before then focusing on the high weight and ignoring the low weight. Self-attention obtains a length of dependence that can be L.

The input sequence is transformed into three vectors through three matrices in the self-attention mechanism. These are query matrix (Q), key matrix (K) and value matrix (V).

The canonical self-attention is defined based on the tuple inputs (Q, K, V). Self-attention performs the scaled dot product that can be summarized as follows [33]:

$$\text{Self-Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{6}$$

where $Q \in \mathbb{R}^{d_x \times d_k}$, $K \in \mathbb{R}^{d_x \times d_k}$, $V \in \mathbb{R}^{d_x \times d_v}$ and d_x is the input dimension. d_k stands for the dimension of the Q martrix and K matrix, and d_v stands for the dimension of the V matrix. We use $\frac{1}{\sqrt{d_k}}$ as the scaling factor.

The concepts of query, key, and value are derived from information retrieval systems. This matches the key to the query to get the matching value. The value is the similarity between the query and the key. In matrix operations, the dot product is a method to compute the similarity of two matrices, and operation $Q \cdot K^T$ computes the similarity of each pair of time steps. Weighted matching is then performed based on similarity, and the weights are the similarities.

In this study, based on attention mechanism, we propose the "cross-attention" mechanism (Fig. 6). The cross-attention mechanism combines both the dot product attention and the additive attention to efficiently capture the dependency between two time series sequences. Like traditional attention mechanisms, there are one query, one key, and one value. The cross-attention block comes immediately after the two encoders. The outputs of the flow encoder were transformed into the query and the value matrices, and the outputs of the ENSO encoder were transformed into the key matrix. Let q_i, k_i, and v_i stand for the i-th row in Q, K, and V respectively.The process of cross-attention can be described as follows:

First, use additive attention to weighted average the key matrix into a global key vector (k):

$$k = \sum_{i=1}^{L} \alpha_i \cdot k_i \tag{7}$$

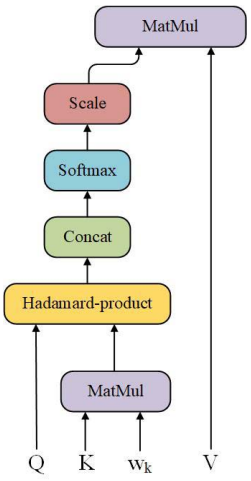


FIGURE 6. The Cross-Attention.

The weight α_i is calculated as follows:

$$\alpha_i = \frac{\exp\left(\mathbf{k}_i \mathbf{w}_k^T / \sqrt{d}\right)}{\sum_{j=1}^{L} \exp\left(\mathbf{k}_j \mathbf{w}_k^T / \sqrt{d}\right)} \tag{8}$$

where $w_k \in \mathbb{R}^L$ is a trainable vector. Use Hadamard product to simulate the nonlinear relation between q_i and k and get a score (s_i):

$$s_i = q_i \circ k \tag{9}$$

The i-th query's cross-attention is defined as:

$$\text{Cross-attention}(q_i, K, V) = \text{Softmax}(s_i/\sqrt{d}) \cdot V \tag{10}$$

It has been proven that the additive attention has a lower computational complexity than dot product attention. The main purpose of using the additive attention is that it can quickly summarize important information in a sequence with linear complexity, which greatly improves the efficiency of multivariate forecasting.

C. DOUBLE-ENCODER TRANSFORMER

It has been proven that El Nino has a close correlation with rainfall, which affects the flow of the river, so El Nino has a significant impact on streamflow [1]. Classic Transformer only has the self-attention block, which is only suitable for dealing with a single time series. When dealing with the prediction issues with multivariate data, it cannot capture the attention among different kinds of variates effectively. The aim of the attention mechanism is to obtain the correlation of one point to all the other points and then weight the average of them so that the attention operation could be more than just self-attention. Because the self-attention process can obtain a correlation among the time steps of one sequence, the correlation between different sequences can be obtained, too. In this study, we improved the classic Transformer by using two encoders and one decoder with a cross-attention block to make streamflow predictions with El Nino covariate. The whole double-encoder architecture is proposed and illustrated in Fig. 7. Because Transformer abandoned the RNN linear sequence structure to support parallel computing, the positional information in time series will be lost. Therefore, adding order signals to vectors is necessary to help the model learn this positional information, so positional encoding [33] is used to solve this problem. Positional encoding works by combining order information and vectors to form a new representation input to the model to learn order information. Positional encoding is itself a vector with order information. In this study, order information and date information are added to the input sequence. The datasets are monthly, so the year and month information were embed into vectors and added to original input vectors with positional encoding.

The original hydrological time series are nonlinear and nonstationary. Some features will be ignored if used directly, causing decreases in the prediction accuracy, especially in flood years. The VMD could decompose original data into

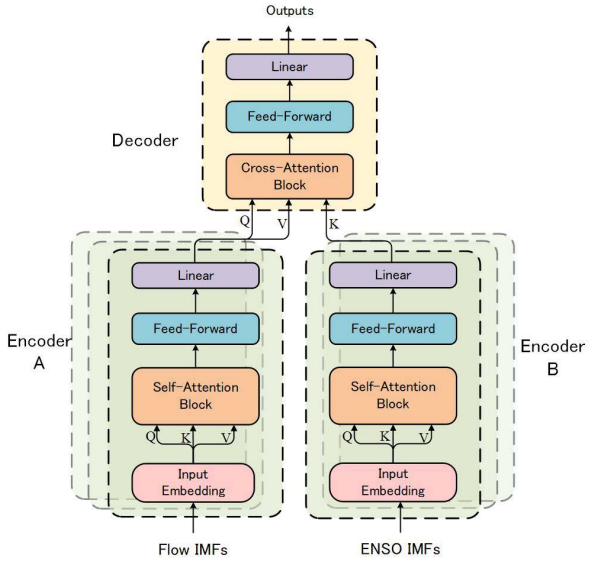


FIGURE 7. The architecture of double-encoder Transformer.

several IMFs. Compared with the other frequency decomposition technique, the fundamental components decomposed by VMD have physical significance and are much more robust to sampling and noise. In this study, the flow sequence was decomposed into nine IMFs and ENSO sequence into eight IMFs. The IMFs were sent into encoders to perform self-attention. Each IMF should have its own encoder block, so there are nine encoder blocks for the flow sequence and eight for the ENSO sequence. The present study adopted the transfer learning method: flow IMFs share a common input embedding block and self-attention block, and so do ENSO IMFs, but the layers behind them are different. To sum up, there are two input embedding blocks, two self-attention blocks, 9+8 feed-forward blocks and 9+8 linear blocks. The introduction of transfer learning can significantly save the space occupied by the model and accelerate the training speed.

All the IMFs of flow and IMFs of ENSO were re-composed as one, respectiovely, after being processed by the encoders. The recomposed flow and ENSO data were sent into the decoder to perform cross-attention. In river flow forecasting, the ENSO dataset works as the covariate. The main purpose of cross-attention is to capture the impact of ENSO on flow. flow was convert into Q and V, and ENSO was convert into K. The details of cross-attention were presented in section II-B.

Let L_in and L_out stand for the input window size and output window size. The proposed method can be summarized as Algorithm 2: where Q_{self_f}, K_{self_f}, V_{self_f}, Q_{self_e}, K_{self_e}, and V_{self_e} are conversion matrices in self-attention. All the IMFs of flow were converted into Q, K, and V using the same Q_{self_f}, K_{self_f}, and V_{self_f} matrices, and all the IMFs of ENSO were converted into Q, K, V using the same Q_{self_e}, K_{self_e}, and V_{self_e} matrices. The w_f and w_e are trainable vectors used to recompose the IMFs.

Algorithm 2 The Process of Proposed Method

Input: The flow data: $F = \{f_1, \ldots, f_{L_{in}}\}$; The ENSO data: $E = \{e_1, \ldots, e_{L_{in}}\}$; The time stamps: $ST = \{st_1, \ldots, st_{L_{in}}\}$.

Output: The predictions of river flow: $F_{pred} = \{f_{L_{in}+1}, \ldots, f_{L_{in}+L_{out}}\}$.

Initialize Q_{self_f}, K_{self_f}, V_{self_f}, Q_{self_e}, K_{self_e}, V_{self_e}, w_f, w_e, w_k, Q_{cross}, K_{cross}, V_{cross};

Use the VMD to decompose the original data:

$$\text{VMD}(F) = \{F_{IMF\_1}, \ldots, F_{IMF\_9}\};$$
$$\text{VMD}(E) = \{E_{IMF\_1}, \ldots, E_{IMF\_8}\};$$

foreach *modal in* VMD(F)*and* VMD(E) **do**

Perform data embedding:

$$EMB = Conv1d(modal) + PE + Conv1d(ST);$$

Convert EMB into Q, K, V matrices usingQ_{self_f}, K_{self_f}, V_{self_f}, Q_{self_e}, K_{self_e}, and V_{self_e}, and compute the Self-attention:

$$\text{Self-attention}(modal) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V;$$

The outputs of the two encoders are:

$$F_{self} = \{F_{self\_1}, \ldots, F_{self\_9}\};$$
$$E_{self} = \{E_{self\_1}, \ldots, E_{self\_8}\};$$

Re-compose the F_{self} and E_{self} into one:

$$F' = \text{concat}(F_{self\_1}, \ldots, F_{self\_9}) \cdot w_f;$$
$$E' = \text{concat}(E_{self\_1}, \ldots, E_{self\_8}) \cdot w_e;$$

Convert F' into Q, and V matrices using Q_{cross} and V_{cross}. Convert E' into K using K_{cross}. Compute the Cross-attention:

foreach q_i *in* Q **do**

$s_i = q_i \circ k = q_i \circ \sum_{i=1}^{L_{in}} \alpha_i \cdot k_i$;

$$\text{Cross-attention}(F', E') = \text{Softmax}(\text{concat}(s_1, \ldots, s_{L_{in}})/\sqrt{d}) \cdot V;$$

The prediction of river flow is:

$$F_{pred} = \text{Feedforward}(\text{Cross-attention}(F', E'))$$

III. EXPERIMENT AND RESULTS

This study focuses on the streamflow predictions on the Hankou Hydrological Station. The goal is to predict river flow with the flow and ENSO datasets. The flow dataset was collected by the Yangtze River Water Resources Commission, and the ENSO dataset was collected by the National Oceanic and Atmospheric Administration. Both datasets were collected monthly from January 1952 to December 2018, hence

totaling 67 years. The datasets were divided into two parts: one from 1952 to 1997 and another from 1998 to 2018. The former was used to train the model, and the latter was used to make predictions. The Yangtze River has experienced several floods from 1952 to 2018, most recently in 2016. In 1998, the Yangtze River experienced a devastating flood because of strong subtropical highs, resulting in the most significant surge over the past 50 years. In this study, the proposed model was proven to work well in both flood years and normal years by making predictions from 1998 to 2018. We have chosen streamflow data in 1998, 2016, and 2018 (two flood years and one normal year) to make predictions and then made 21 years of rolling predictions from January 1998 to December 2018 to further verify the overall reliability.

Some representative models were selected as comparisons, including the traditional statistical analysis method: autoregressive integrated moving average model (ARIMA), the convolutional neural network: TCN [27], the representative of the RNN: LSTMa [48], and the classic Transformer [33]. For CNNs, RNNs, and classic Transformer, there is no structure specifically designed to support multidimensional prediction, so it is general to combine all dimensions into feature vectors as input when dealing with multidimensional time series. All the models have a 2d correlated input to make multidimensional time series forecasting. Under the fixed-size window forecasting setting, for double-encoder Transformer, the input flow sequence is $F_t = \{f_{t-L}, \ldots, f_t \mid f_i \in \mathbb{R}^{d_x}\}$ from time $t - L$ to time t, and the input ENSO sequence is $E_t = \{e_{t-L}, \ldots, e_t \mid e_i \in \mathbb{R}^{d_x}\}$. And the input sequence for TCN, LSTMa, and classic Transformer is $FE_t = \{(f_{t-L}, e_{t-L}), \ldots, (f_t, e_t) \mid f_i, e_i \in \mathbb{R}^{d_x}\}$. The output predicts the corresponding sequence $F' = \{f'_t, \ldots, f'_{t+l} \mid f'_i \in \mathbb{R}^{d_x}\}$. L is the length of the inputs. It has been proven that El Nino is cyclical, with a cycle of about 5.5 years [49] and through experiments, we have verified that there will be better results when the input length (L) is 72 (6 years). l is the length of the prediction steps. In this study, l is set to 12 so that the decision makers can take action to prepare for the flood a year in advance. For the TCN model, the kernel size was 24, and the stride was 1. The MSE-Loss was selected as the loss function, here using AdamW as the optimizer. To measure the performance of the results, two evaluation metrics were introduced, including root mean squared error (*RMSE*) and coefficient of determination (R^2). The RMSE was defined as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2} \tag{11}$$

and the R^2 was defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y}_i)^2} \tag{12}$$

A. ORIGINAL DATA DECOMPOSE

To improve the accuracy and speed up the convergence of the model, data normalization is necessary. Min–max scaling was used to perform data normalization. For data $X = \{x_1, x_2, x_3, \ldots, x_n\}$, the min-max scaling is defined as:

$$x_{i(scaled)} = \frac{x_i - \min(X)}{\max(X) - \min(X)} \cdot (max - min) + min \tag{13}$$

where the $[min, max]$ is a scaling interval and all the data is scaled into the interval. In this study, both the flow and ENSO data were scaled into $[-1, 1]$. The scaled data from January 1952 to December 2018 were decomposed into nine IMFs and eight IMFs, respectively. The decomposed results are shown in Fig. 8(a) and Fig. 8(b).

B. THE FLOOD YEARS AND NORMAL YEAR PREDICTIONS

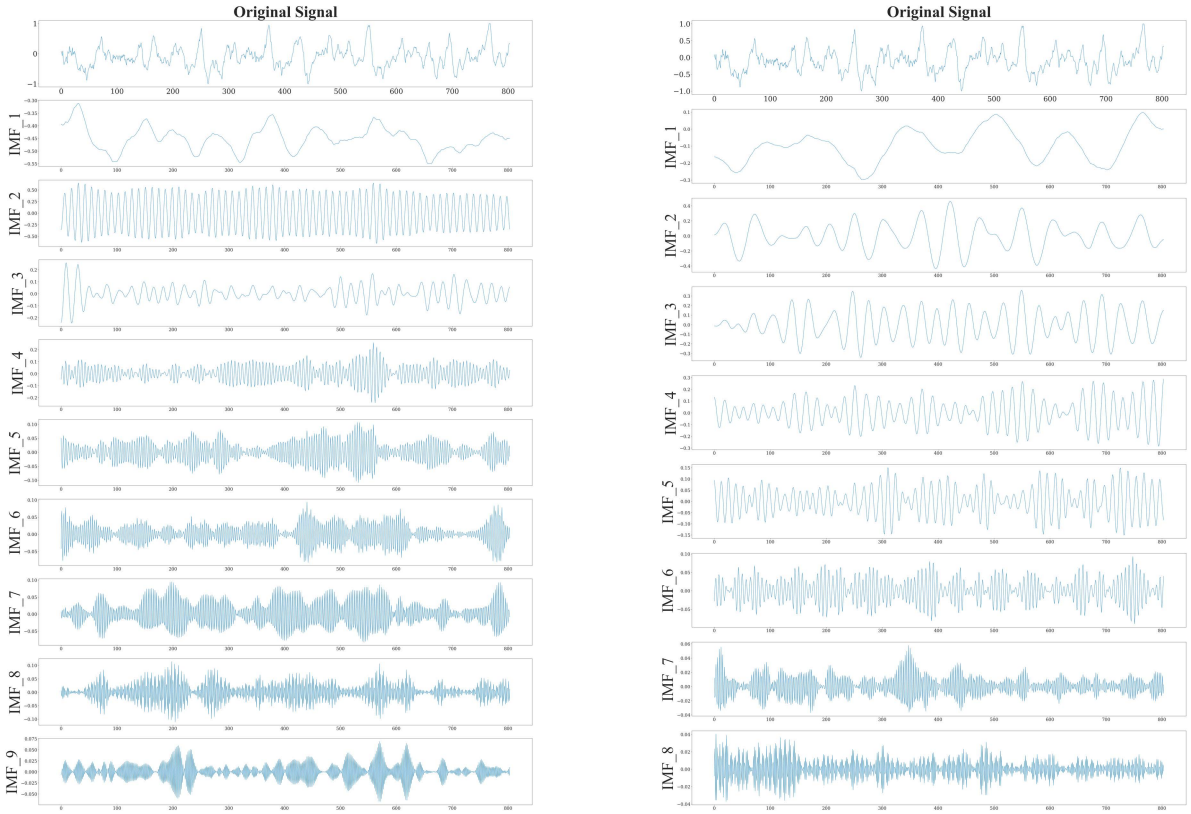
The main purpose of river flow forecasting is to predict floods, especially the flood peak discharge, helping those affected prepare to fight the flood ahead of time. So it is crucial to predict floods accurately, which previous models have not been able to do. Traditional models could obtain decent performance in normal years but did not work well in flood years. In this section, the time series from January 1952 to December 1997 were used to train the model, and 1998, 2016, and 2018 were selected to make predictions to measure the model's performance. Taking 1998 predictions as an example, the input data is the flow data and ENSO data from January 1992 to December 1997 (a total of 72 months). The output is a 12-months prediction for 1998.

ARIMA, TCN, LSTMa, and classic Transformer were selected as comparisons that made the same predictions on these models. Fig. 9(a) and Fig. 9(b) show the whole 12 months of predictions for 1998 and 2016, respectively. Additionally, data in 2018 were selected as a representative of normal years to measure the model's performance in normal years. Fig. 9(c) shows the whole 12-month prediction of 2018. Finally, the metrics including R^2 and *RMSE* were used to measure the performances of all the models in 1998, 2016, and 2018. Table 2 shows the *RMSE* and R^2 of all the models on these three years of predictions.

All three models could fit the streamflow change trend that increases first and then decreases. They all performed well at the beginning and end of the year (when the streamflow is low), and the gap among their forecasts was widest in June, July, and August (when the streamflow is at its peak of the year). The double-encoder Transformer had a higher R^2 and lower *RMSE*. In the flood years (1998, 2016), it is evident that double-encoder Transformer fits better than other models with the actual values, especially the flood peak, and the R^2 could reach more than 0.95. In a normal year (2018), all three models performed well, but double-encoder Transformer was more accurate than the others, here with a 0.94 R^2.

C. THE 21 YEARS OF ROLLING PREDICTIONS

In section III-B, three years were selected to perform predictions, and the results showed that the double-encoder Transformer had significant advantages in flood years and normal years. However, the results of the three-year pre-



(a) Flow (b) ENSO

FIGURE 8. **The IMFs of the original data.**

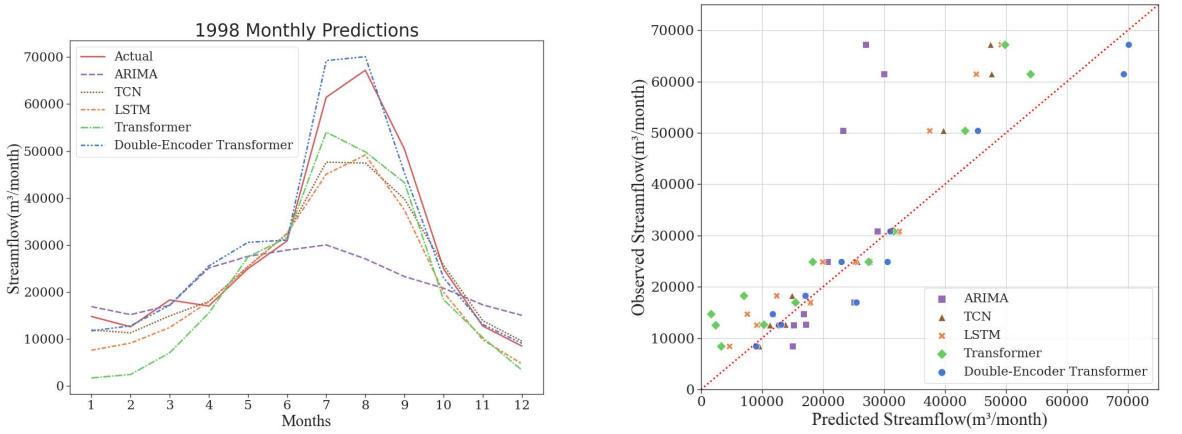
TABLE 2. **The streamflow forecasting results on three years (five models).**

Years	1998		2016		2018	
Metric	R^2	RMSE(m^3/month)	R^2	RMSE(m^3/month)	R^2	RMSE(m^3/month)
ARIMA	0.6853	14104.98	0.6855	8665.65	0.8196	3064.91
TCN	0.8643	6762.16	0.8241	4460.78	0.8604	2468.20
LSTM	0.8421	7722.26	0.7963	4952.89	0.8432	2839.69
Transformer	0.8469	7634.24	0.7465	6072.85	0.8910	2025.69
double-encoder Transformer	0.9541	3224.71	0.9503	1674.14	0.9401	1171.70

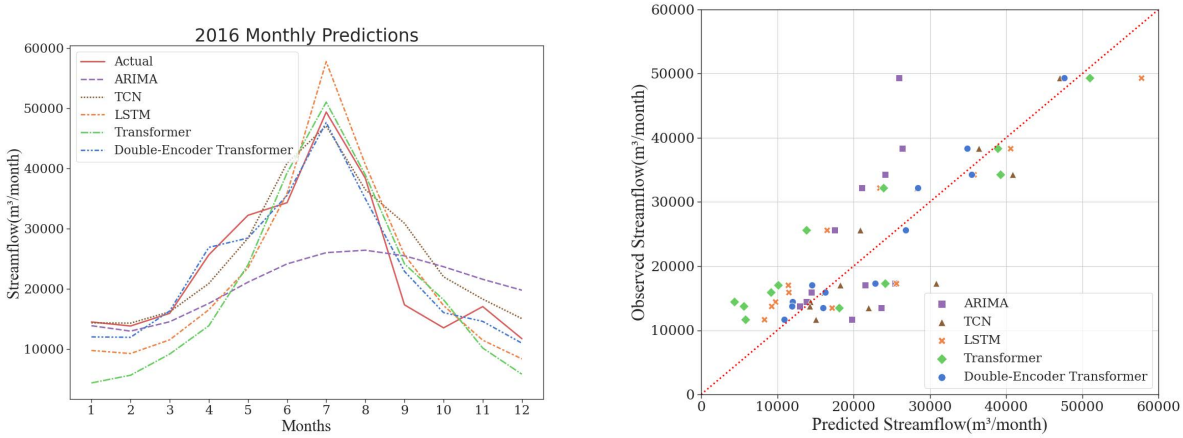
dictions were largely haphazard. In this section, a 21-year rolling forecast (from 1998 to 2018) was made to test the models' capacity for long-term prediction. Fig. 10(a) shows the 21 years of rolling predictions of all models and the actual value: it is a little confusing, so Fig. 10(b) shows the double-encoder Transformer and the actual value in isolation. Fig. 10(c) and Fig. 10(d) are corresponding scatter plots.

Annual R^2 and *RMSE* were calculated for each year of the five models and finally obtained the 21 years of the average R^2 and *RMSE*. Fig. 11(a) and Fig. 11(b) show the 21 years of the annual R^2 and *RMSE*, respectively. The results of the 21-year prediction show that the double-encoder Transformer was better than the traditional time series models, with an average R^2 of more than 0.91 and an average *RMSE* of lower than 2600 m^3/month.

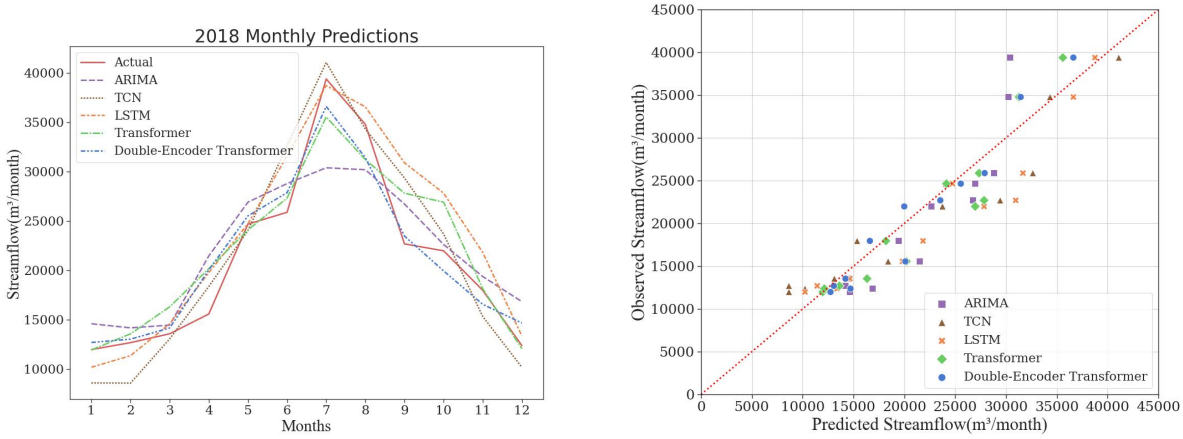
The experimental results show that double-encoder Transformer is superior to the current time series forecasting models. In 21 years of rolling predictions, the average R^2 of the double-encoder Transformer could reach more than 0.91, which is higher than the other models by about 0.1, and the *RMSE* was just 2579 m^3/month, nearly half of the other models. It can be seen from Fig. 10(c) that these points of the five models are very concentrated near the actual red line at first, which means that they all performed well when streamflow was low. As the flow continued to increase, the distribution of the points became more scattered, resulting in bad predictions. In this case, the



(a) The 1998 monthly predictions



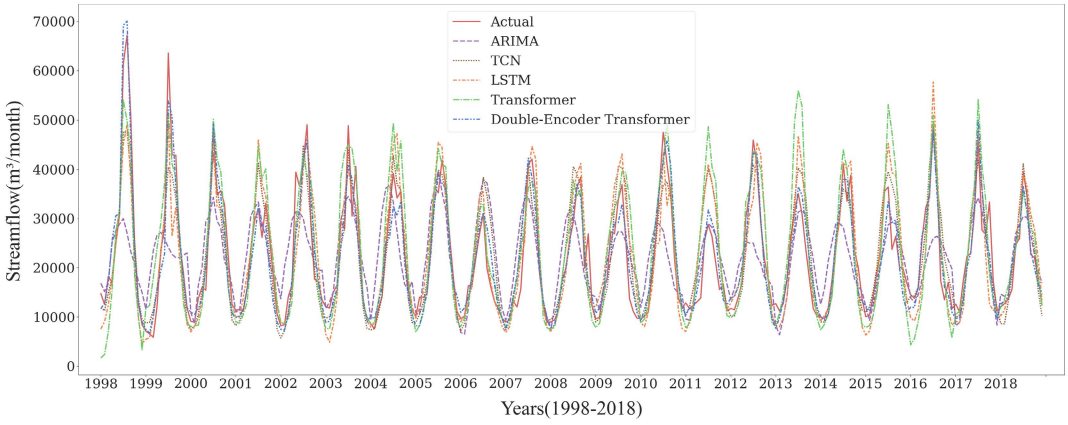
(b) The 2016 monthly predictions



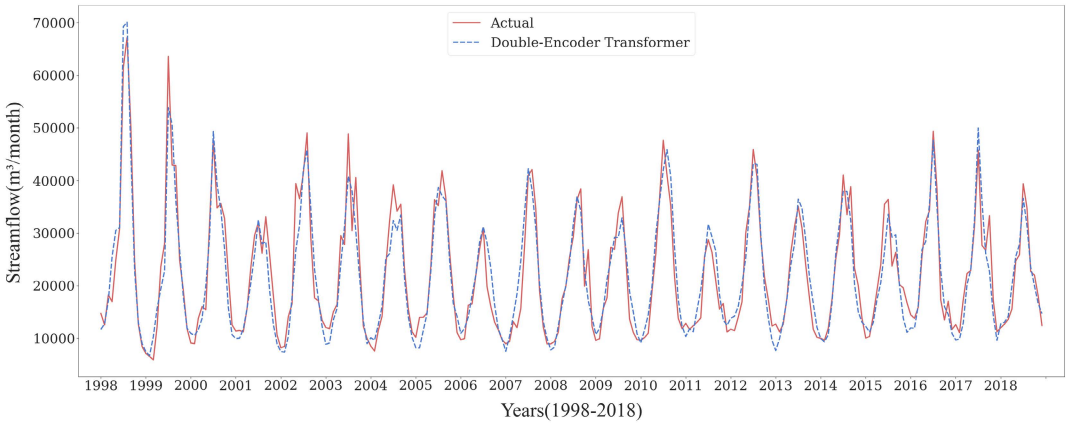
(c) The 2018 monthly predictions

FIGURE 9. The forecasted streamflow of models and the validation scatter plot in 1998, 2016, and 2018.

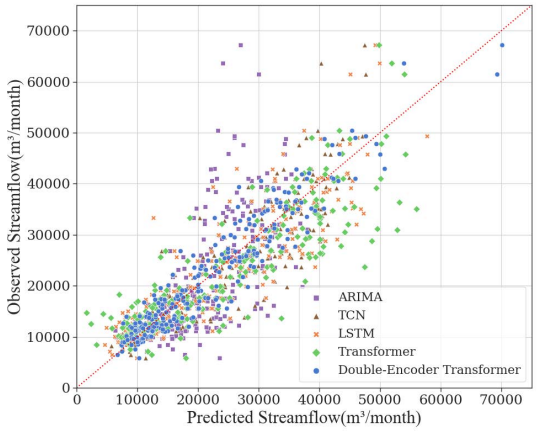
double-encoder Transformer had a considerable advantage of being more accurate in predictions of high streamflow. The results fully prove that the double-encoder Transformer could perform very well in normal and flood years and could be reliable enough in long-term predictions with high accuracy.



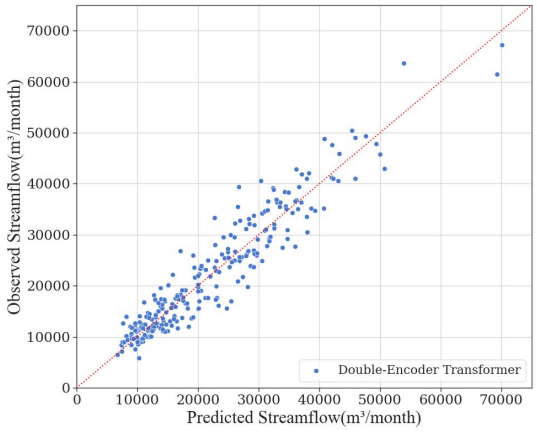
(a)



(b)

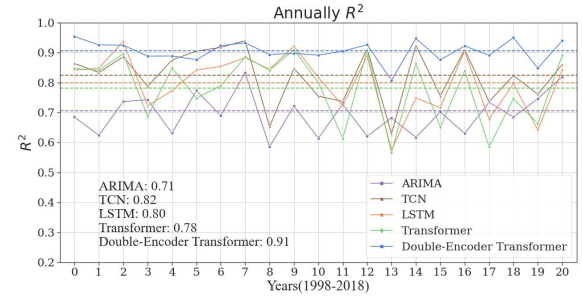


(c)

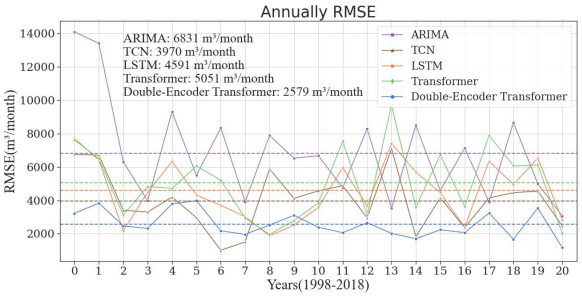


(d)

FIGURE 10. A 21-year rolling forecast (from January 1998 to December 2018) was made to test the models' capacity. Fig. 10(a) shows the actual value and the prediction results of all models in 21 years. To show the proposed model's performance more clearly, Fig. 10(b) shows the actual value and the result of the double-encoder Transformer in isolation. Fig. 10(c) and Fig. 10(d) are the corresponding scatter plots for model comparison. It can be seen from the scatter plots that when streamflow was in low value, all the models have great performance (scatters are concentrated near the red line), but when streamflow was in high value, the models except double-encoder Transformer perform worse (scatters are away from the red line).



(a) The R^2 of 21 years



(b) The RMSE of 21 years

FIGURE 11. The R^2 and RMSE of 21 years of rolling predictions.

IV. CONCLUSION

In this work, the streamflow forecasting problem of the Yangtze River was studied, and a new Transformer-based double-encoder-enabled model was proposed: double-encoder Transformer. This model, combined with the VMD algorithm, can effectively make precise long-term streamflow forecasting of the Yangtze River, especially in flood years. Although the model is still of an encoder–decoder architecture, it alleviates the limitation of a traditional encoder–decoder architecture. Specifically, we designed the cross-attention mechanism to handle the challenges of not supporting multivariate prediction in traditional time series forecasting models. Combining the additive attention with the dot product attention, the cross-attention mechanism can effectively capture the relation between flow and ENSO data.

The experiments on real-world data of the Hankou Hydrological Station demonstrated the effectiveness of the new model for enhancing the prediction capacity both in normal and flood years. A reliable long-term monthly prediction (from 1998 to 2018) was made. There were floods in two of these 21 years (1998 and 2016). The R^2 in both years is higher than 0.95, and the *RMSE* value is just 3224 m^3/month, which is when the flow reached nearly 70000 m^3/month in 1998. Other mainstream forecasting models, including ARIMA, TCN, LSTMa, and classic Transformer, were selected as comparisons to demonstrate the superiority of the double-encoder Transformer. In the 21 years of predictions, the average R^2 of double-encoder Transformer was about 0.91, which is higher than the other model by 0.1, and the *RMSE* was 2579 m^3/month, which is significantly lower than the other models. These experimental results show that the double-Encoder Transformer can be used in real-world streamflow predictions.

The main work of this study is to predict the streamflow of the Yangtze River, focusing on flood prediction. However, drought year is also important because water is a vital resource on earth and we depend heavily on river water. Accurate drought forecasting is promising for future research. The variation of river flow in the Yangtze River is related to many variables. In this work, we made a bivariate forecast with the ENSO data. The outcomes can be improved by adding more variables to make multivariate forecasting. The cross-attention mechanism is a promising method that can efficiently summarize the features of a sequence into a vector and compute the attention between the independent variable and covariates. This work is a good start in making the multivariate long sequence time series forecasting. We are excited about the future of models with the cross-attention mechanism.

REFERENCES

[1] J. Wei, W. Wang, Q. Shao, Y. Rong, W. Xing, and C. Liu, "Influence of mature El Niño–Southern oscillation phase on seasonal precipitation and streamflow in the Yangtze River Basin, China," *Int. J. Climatol.*, vol. 40, no. 8, pp. 3885–3905, Jun. 2020.

[2] J. Peng, X. Luo, F. Liu, and Z. Zhang, "Analysing the influences of ENSO and PDO on water discharge from the Yangtze River into the sea," *Hydrol. Processes*, vol. 32, no. 8, pp. 1090–1103, Apr. 2018.

[3] Q. Zhang, C.-Y. Xu, T. Jiang, and Y. Wu, "Possible influence of ENSO on annual maximum streamflow of the Yangtze River, China," *J. Hydrol.*, vol. 333, nos. 2–4, pp. 265–274, Feb. 2007.

[4] F. Huang, Z. Xia, N. Zhang, Y. Zhang, and J. Li, "Flow-complexity analysis of the upper reaches of the Yangtze River, China," *J. Hydrol. Eng.*, vol. 16, no. 11, pp. 914–919, Nov. 2011.

[5] M. Cheng, F. Fang, T. Kinouchi, I. M. Navon, and C. C. Pain, "Long lead-time daily and monthly streamflow forecasting using machine learning methods," *J. Hydrol.*, vol. 590, Nov. 2020, Art. no. 125376.

[6] V. K. Keteklahijani, S. Alimohammadi, and E. Fattahi, "Predicting changes in monthly streamflow to karaj dam reservoir, iran, in climate change condition and assessing its uncertainty," *Ain Shams Eng. J.*, vol. 10, no. 4, pp. 669–679, Dec. 2019.

[7] C. Ma and Y. Li, "Improving forecasting accuracy of annual runoff time series using RBFN based on EEMD decomposition," in *Proc. DEStech Trans. Eng. Technol. Res.*, 2017, pp. 211–216.

[8] G. Yu, H. Ye, Z. Xia, and X. Zhao, "Application of projection pursuit auto regression model in predicting runoff of Yangtze River," *J. Hohai Univ., Natural Sci.*, vol. 37, no. 3, pp. 263–266, 2009.

[9] N. Noori and L. Kalin, "Coupling SWAT and ANN models for enhanced daily streamflow prediction," *J. Hydrol.*, vol. 533, pp. 141–151, Feb. 2016.

[10] J. Arnold, "Swat-soil and water assessment tool," USDA Agricult. Res. Service, Grassland, Soil Water Res. Lab., Temple, TX, USA, Tech. Rep., 1994.

[11] M. Kirkby and K. Beven, "A physically based, variable contributing area model of basin hydrology," *Hydrol. Sci. J.*, vol. 24, no. 1, pp. 43–69, 1979.

[12] R.-J. Zhao, "The Xinanjiang model applied in China," *J. Hydrol.*, vol. 135, nos. 1–4, pp. 371–381, 1992.

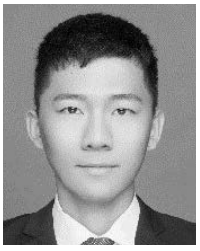
[13] Z. M. Yaseen, S. O. Sulaiman, R. C. Deo, and K.-W. Chau, "An enhanced extreme learning machine model for river flow forecasting: State-of-the-art, practical applications in water resource engineering area and future research direction," *J. Hydrol.*, vol. 569, pp. 387–408, Feb. 2019.

[14] W. Collischonn, R. Haas, I. Andreolli, and C. E. M. Tucci, "Forecasting river Uruguay flow using rainfall forecasts from a regional weather-prediction model," *J. Hydrol.*, vol. 305, nos. 1–4, pp. 87–98, Apr. 2005.

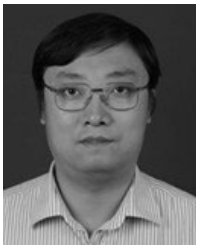
[15] Z. A. Al-Sudani, S. Q. Salih, A. Sharafati, and Z. M. Yaseen, "Development of multivariate adaptive regression spline integrated with differential evolution model for streamflow simulation," *J. Hydrol.*, vol. 573, pp. 1–12, Jun. 2019.

[16] W.-C. Wang, K.-W. Chau, D.-M. Xu, L. Qiu, and C.-C. Liu, "The annual maximum flood peak discharge forecasting using Hermite projection pursuit regression with SSO and LS method," *Water Resour. Manage.*, vol. 31, no. 1, pp. 461–477, Jan. 2017.
[17] Y. Sang, L. Shang, Z. Wang, C. Liu, and M. Yang, "Bayesian-combined wavelet regressive modeling for hydrologic time series forecasting," *Chin. Sci. Bull.*, vol. 58, no. 31, pp. 3796–3805, Nov. 2013.
[18] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.
[19] S. Zhu, J. Zhou, L. Ye, and C. Meng, "Streamflow estimation by support vector machine coupled with different methods of time series decomposition in the upper reaches of Yangtze River, China," *Environ. Earth Sci.*, vol. 75, no. 6, p. 531, Mar. 2016.
[20] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[21] J. L. Elman, "Finding structure in time," *Cogn. Sci.*, vol. 14, no. 2, pp. 179–211, Mar. 1990.
[22] K.-I. Funahashi, "On the approximate realization of continuous mappings by neural networks," *Neural Netw.*, vol. 2, no. 3, pp. 183–192, 1989.
[23] O. Kisi and H. K. Cigizoglu, "Comparison of different ANN techniques in river flow prediction," *Civil Eng. Environ. Syst.*, vol. 24, no. 3, pp. 211–231, Sep. 2007.
[24] M. Rezaeianzadeh, H. Tabari, A. A. Yazdi, S. Isik, and L. Kalin, "Flood flow forecasting using ANN, ANFIS and regression models," *Neural Comput. Appl.*, vol. 25, no. 1, pp. 25–37, Jul. 2014.
[25] M. C. Demirel, A. Venancio, and E. Kahya, "Flow forecast by SWAT model and ANN in Pracana basin, Portugal," *Adv. Eng. Softw.*, vol. 40, no. 7, pp. 467–473, Jul. 2009.
[26] S.-Y. Shih, F.-K. Sun, and H.-Y. Lee, "Temporal pattern attention for multivariate time series forecasting," *Mach. Learn.*, vol. 108, nos. 8–9, pp. 1421–1441, Sep. 2019.
[27] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, *arXiv:1803.01271*.
[28] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[29] Z. Xiang, J. Yan, and I. Demir, "A rainfall-runoff model with LSTM-based sequence-to-sequence learning," *Water Resour. Res.*, vol. 56, no. 1, Jan. 2020, Art. no. e2019WR025326.
[30] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 802–810.
[31] D. Liu, W. Jiang, L. Mu, and S. Wang, "Streamflow prediction using deep learning neural network: Case study of Yangtze River," *IEEE Access*, vol. 8, pp. 90069–90086, 2020.
[32] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*.
[33] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[34] P. Schwaller, T. Laino, T. Gaudin, P. Bolgar, C. A. Hunter, C. Bekas, and A. A. Lee, "Molecular transformer: A model for uncertainty-calibrated chemical reaction prediction," *ACS Central Sci.*, vol. 5, no. 9, pp. 1572–1583, Sep. 2019.
[35] L. Yang, T. L. J. Ng, B. Smyth, and R. Dong, "HTML: Hierarchical transformer-based multi-task learning for volatility prediction," in *Proc. Web Conf.*, Apr. 2020, pp. 441–451.
[36] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," 2021, *arXiv:2102.12122*.
[37] S. Ha, D. Liu, and L. Mu, "Prediction of Yangtze River streamflow based on deep learning neural network with El Niño–Southern oscillation," *Sci. Rep.*, vol. 11, no. 1, pp. 1–23, Dec. 2021.
[38] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proc. AAAI*, 2021, pp. 1–9.
[39] K. Dragomiretskiy and D. Zosso, "Variational mode decomposition," *IEEE Trans. Signal Process.*, vol. 62, no. 3, pp. 531–544, Feb. 2014.
[40] R. A. van den Berg, H. C. Hoefsloot, J. A. Westerhuis, A. K. Smilde, and M. J. van der Werf, "Centering, scaling, and transformations: Improving the biological information content of metabolomics data," *BMC Genomics*, vol. 7, no. 1, pp. 1–15, Dec. 2006.
[41] G. Sugihara and R. M. May, "Nonlinear forecasting as a way of distinguishing chaos from measurement error in time series," *Nature*, vol. 344, no. 6268, pp. 734–741, 1990.
[42] S. Mehran Kazemi, R. Goel, S. Eghbali, J. Ramanan, J. Sahota, S. Thakur, S. Wu, C. Smyth, P. Poupart, and M. Brubaker, "Time2Vec: Learning a vector representation of time," 2019, *arXiv:1907.05321*.
[43] I. Daubechies, J. Lu, and H. T. Wu, "Synchrosqueezed wavelet transforms: An empirical mode decomposition-like tool," *Appl. Comput. Harmon. Anal.*, vol. 30, no. 2, pp. 243–261, Mar. 2011.
[44] J. Gilles, "Empirical wavelet transform," *IEEE Trans. Signal Process.*, vol. 61, no. 16, pp. 3999–4010, Aug. 2013.
[45] N. E. Huang, Z. Shen, S. R. Long, M. C. Wu, H. H. Shih, Q. Zheng, N.-C. Yen, C. C. Tung, and H. H. Liu, "The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis," *Proc. Roy. Soc. London A, Math., Phys. Eng. Sci.*, vol. 454, no. 1971, pp. 903–995, 1998.
[46] D. Hilbert, "Mathematical problems," *Bull. Amer. Math. Soc.*, vol. 8, no. 10, pp. 437–479, 1902.
[47] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, *arXiv:1406.1078*.
[48] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," 2015, *arXiv:1506.07503*.
[49] E. Bryant, E. A. Bryant, and B. Edward, *Climate Process and Change*. Cambridge, U.K.: Cambridge Univ. Press, 1997.

CHUANFENG LIU received the B.S. degree in engineering from the China University of Geosciences, Wuhan, China, where he is currently pursuing the master's degree. His research interests include machine learning, neural networks, and physical oceanography.



DARONG LIU received the B.S. degree in engineering from the China University of Geosciences, Wuhan, China, where he is currently pursuing the Ph.D. degree. During the undergraduate study, his research interests include computer science, three-dimensional modeling, and numerical simulation in geoscience. His current research interests include numerical simulation in physical oceanography, machine learning, and neural networks.



LIN MU received the B.S., M.S., and Ph.D. degrees in physical oceanography from the Ocean University of China, Qingdao, China, in 2000, 2002, and 2007, respectively. He is currently a Professor of physical oceanography with Shenzhen University and the Shenzhen Research Institute, China University of Geosciences, Guangdong, China. He has authored or coauthored over 20 scientific articles and four books. His research interests include physical oceanography: prevention and mitigation of marine disasters, maritime search and rescue, and emergency response management of offshore oil spills.