**ABSTRACT**

Water supplies are crucial for the development of living beings. However, change in the hydrological process i.e. climate and land usage are the key issues. Sustaining water level and accurate estimating for dynamic conditions is a critical job for hydrologists, but predicting hydrological extremes is an open issue. In this paper, we proposed two deep learning techniques and three machine learning algorithms to predict stream flow, given the present climate conditions. The results showed that the Recurrent Neural Network (RNN) or Long Short-term Memory (LSTM), an artificial neural network based method, outperform other conventional and machine-learning algorithms for predicting stream flow. Furthermore, we analyzed that stream flow is directly affected by precipitation, land usage, and temperature. These indexes are critical, which can be used by hydrologists to identify the potential for stream flow. We make the dataset publicly available (https://github.com/sadaqat007/Dataset) so that others should be able to replicate and build upon the results published.

**Keywords** Deep learning · Machine learning · Streamflow estimation · Surface water hydrology · River/stream
1 Introduction

Water is an important element for all human beings as well as all living things on the earth. Most of the water for domestic, industrial and agriculture use is obtained from the streamflow. The researchers concluded that streamflow has changed across the world [1, 2].

Streamflow is the main component of the hydrological cycle and it is essential for the stability of our ecosystem. Streamflow has a vital impact on the quality of water, living beings, and habitat in the stream [3]. Streamflow controls the kinds of organisms that can survive in the stream, for example, some organisms are susceptible to fast-flowing areas, while others need stagnant pools. Moreover, it also controls the quantity of residue and sediment carried by the stream. Large, rapidly flowing rivers are less affected by pollution, whereas narrow small streams have less capability to dilute and reduce wastes [4].

Automating and management of streamflow have the potential to significantly reduce the withdrawals of water for irrigation purposes, as well as to improve the quality of agriculture. The amount of water moving off the watershed into the stream channel is highly related to the flow of a stream. It is affected by the variation in weather, under different conditions such as rainstorm and dry periods. Moreover, different seasons also fluctuate stream flow level, for example, when evaporation is high and the ground is fertile thus removing water from the ground due to which streamflow level

*Corresponding author.
decreases and vice versa. Another serious problem of streamflow depletion is water withdrawals for irrigation purposes, for example, industrial water withdrawals [5, 6].

Streamflow estimation and forecasting are challenging and essential for better provision and management of water resources [7]. The streamflow prediction can be useful for hydropower generation. Hydrologists have developed different models and methods to estimate the streamflow [8]. Recently with the development of the computer, programming attempts have been made to estimate streamflow. The artificial neural network (ANN) has been applied successfully for rainfall-runoff modeling, discharge prediction, sediment quantification, and inflow forecasting [9, 10, 11].

In this paper, we applied deep learning techniques for high-level semantic understanding for simulating the daily streamflow. Specifically, two deep learning models: CNN and LSTM are adapted. This work is of great significance and has aroused a wide spread of concern [12]. In the current era, deep learning techniques have made significant development in fields such as artificial intelligence [13, 14, 15] and natural language processing [16, 17]. These advancements of deep learning techniques can help the merging of different technologies and inventions in other fields. Many applications are witness that deep learning techniques have benchmark ability in feature extractions and
predictions [13, 18], which is free from conventional handy feature extraction methods but carries out self-learning through data. This can greatly leverage hydrological extremes, and alleviate the problem of prediction for stream flow. In a nutshell, we don’t need to implement a complex mathematical model for information learning and decision making, but the deep learning models itself can automatically extract useful information from the hydrological condition records by self-learning, and then conduct streamflow prediction based on these information. This makes our model efficient and robust compared to hand-crafted models. The complete overview of the proposed framework is illustrated in Fig. 1.

Thus, we selected a tropical basin, the Soan River basin, which is the left bank tributary of the Indus River basin, for streamflow assessment. The selected area is very crucial as it has high geographical importance. It is the chief drinking water source to Islamabad city, Pakistan, and the key source of irrigation water for Potohar plateau. In the past few decades, it undergoes a significant climatic and land use [19, 20]. To our best knowledge, we took the first effort to commence the trend analysis of hydrological variables using deep learning algorithms for predicting streamflow change in the Soan river basin. Consequently, in this paper, we present and compare results from two different deep learning models, such as CNN and RNN, and some statistical machine learning models to assess the streamflow of the Soan River basin. The main aims of this work are evaluating the performance of different deep learning and machine learning techniques in predicting the stream flow in Soan river basin (Punjab, Pakistan) based on the existing hydrological variables, and to decide an efficient and robust model for this task.

2 Dataset Evaluation

2.1 Soan river basin.

In this paper, our main focus is on the Soan river basin. The people of Islamabad (capital territory of Pakistan) obtain 95% drinking water from a simly dam, which is the chief reservoir for the Soan river basin. Soan river initiating from Murree Mountains and runs through the Chirah hydrological station, a major branch of Indus River and a significant hydrological unit of Potohar plateau Pakistan. Figure 2 shows the visual description of the whole site. After that, it is joined by different branches and at Dhoke Pathan hydrological station it falls into the Indus River. Sub-basin1 is the area above the Chirah hydrological station called Chirah sub-basin, and sub-basin 2 is the area above Dhoke Pathan hydrological station called Dhoke Pathan sub-basin. The Soan basin covered area is $6842 km^2$ having elevation in the range of 265m – 2274m. Statistical data obtained from Pak govt. for the period of 1983 – 2012 shows that the annual precipitation of the basin is 1465mm and the mean annual temperature range from 8$^\circ$C to 18$^\circ$C. The Soan basin has both gentle and steep slope, stream flow in the basin mostly comes from monsoon rainfall. About 60% of the population associated is rural and cultivation is the main source of economy. During last few years, the population is greatly affected by the unpredictable variation in stream flow of soan river basin, therefore, the implementation of a highly predictable performance learning system is the need of the day to assess the variation of streamflow for agriculture purpose [21, 22].

2.2 Data description.

Pakistan meteorological department (PMD) and Soil and Water Conservation Research Institute (SAWCRI) provides the climatological data for this work. The key hydroclimatic parameters, such as temperature and precipitation, were obtained from six different meteorological stations. Water and Power Development Authority (WAPDA) and Capital Development Authority (CDA) provide the daily discharge data of two hydrological stations. Mwangi [23] et al., elaborate the discharge of Soan basin in detail and computed the runoff for the basin, sub-basin1 and sub-basin2 using a conventional method. We obtain statistical data for thirty years (1983-2012). The survey of Pakistan provides the topographic information of the study area. Punjab bureau of statistics and the Department of Agriculture (Marketing Wing) provided the agriculture production information. The complete data descriptions of hydro-meteorological variables are summarized in Table 1.

In this work, the predictors are precipitation and temperature, whereas streamflow level is the response variable. These parameters are selected based on a wealth of case studies. Specifically, streamflow level is highly sensitive to precipitation. For similar reasons, as in Ficchi [24] et al., we also include only two meteorological variables, such as precipitation and temperature to provide a challenging scenario for learning algorithms to leverage streamflow.

3 Models Design and Analysis.

In this study, we propose two different learning algorithms to extract features from hydroclimatic variables and predict streamflow levels. Specifically, CNN and RNN. As we discussed before, the key aim of selecting these different types
Table 1: Hydrological Characteristics of the Soan basin and its sub-basins, 1983-2012.

| Area of Study | Area/km² | Precipitation (mm/year) | Potential Evapotranspiration (mm/year) | Runoff (mm/year) | Evaporation (mm/year) |
|---------------|----------|-------------------------|---------------------------------------|------------------|-----------------------|
| Soan Basin    | 6675     | 972.1                   | 1382.5                                | 241.8            | 730.3                 |
| Sub-basin 1   | 335      | 1649.4                  | 934.5                                 | 695.8            | 953.6                 |
| Sub-basin 2   | 6340     | 988.3                   | 1394.1                                | 190.5            | 797.8                 |

The function of learning algorithms is to validate streamflow prediction on a variety of algorithms. The overall system frameworks are shown in Figure 1. Details descriptions and mathematical models for the individual algorithm are as follows.

### 3.1 CNN-based prediction model.

Convolutional Neural Networks are the class of deep learning and is very similar to ordinary Neural Networks. They are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network expresses a single differentiable score function from the raw image pixels on one end to class scores at the other [25, 26]. For the CNN-based prediction model, a convolutional neural network is introduced to extract features. The input to the model is the daily temperature and rainfall information for the upstream cities, and the output is the predicted river flow. The architecture of the proposed CNN for streamflow prediction is as follows:

It consists of a convolutional layer with three various sizes of convolutional kernels, followed by an average pooling layer and a fully-connected (fc) layer with a supervision signal called softmax loss.

The function of the convolutional layer is to extract features map from the input data matrix and different sizes convolutional kernels can learn various low-level features. The pooling layer sufficed as a downsampling layer to boost up system efficiency and robustness [13, 27]. Moreover, all these features are fused in the fully-connected layer, which further passes them to a supervision signal called softmax classifier for streamflow prediction. The purpose of the softmax classifier is to compute the similarity measurement between input feature vectors and finally display the streamflow probability. Details description of parameter tuning is given in the “Experimental Setting” section.

For individual input data matrix, we describe it with a matrix $X \in \mathbb{R}^{N \times D}$, as illustrated in Eq. 1. Whereas the $i$-th row suggests the $i$-th day, and $D$ represents the dimension of the input feature. Mathematically it can be represented as:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,D} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,D} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N,1} & a_{N,2} & \cdots & a_{N,D} \end{bmatrix}$$

Generally, let $X_{i:j}$ refer to the matrix which feature vectors from the $i$-th day to the $j$-th day, mathematically:

$$X_{i:j} = \begin{bmatrix} x_i \\ x_{i+1} \\ \vdots \\ x_j \end{bmatrix} = \begin{bmatrix} a_{i,1} & a_{i,2} & \cdots & a_{i,D} \\ a_{i+1,1} & a_{i+1,2} & \cdots & a_{i+1,D} \\ \vdots & \vdots & \ddots & \vdots \\ a_{j,1} & a_{j,2} & \cdots & a_{j,D} \end{bmatrix}$$

Each convolution layer is made up of several convolution kernels of various sizes, and every size further contains many convolution kernels. The width of the convolution kernel is aligned with the width of the input data matrix. For example, if the height of $k$-th convolution kernel is $H$, then it can be described as $W^k \in \mathbb{R}^{H \times D}$, mathematically:

$$W^k = \begin{bmatrix} w_{1,1}^k & w_{1,2}^k & \cdots & w_{1,D}^k \\ w_{2,1}^k & w_{2,2}^k & \cdots & w_{2,D}^k \\ \vdots & \vdots & \ddots & \vdots \\ w_{H,1}^k & w_{H,2}^k & \cdots & w_{H,D}^k \end{bmatrix}$$
The function of convolution operation is the features extraction for local region elements of the input data matrix. For instance, when $w_{i,j}^k$ and $a_{i,j}$ co-occur, then the feature $c_{i}^k$ extracted from $X_{i,:}$ by the convolutional kernel can be expressed as:

$$c_{i}^k = f \left( \sum_{i=1}^{H} \sum_{j=1}^{D} w_{i,j}^k \cdot a_{i,j} + b_{i,j}^k \right), \quad (4)$$

weight $w_{i,j}^k$ represents the significance of the $j$-th value in the $i$-th row vector, $b_{i,j}^k$ denote the bias term and $f$ is a nonlinear function, as described in [13]. The ReLu function is described as nonlinear function, which is mathematically illustrated as:

$$z = ReLu(i) = max(0, i). \quad (5)$$

The convolution process is obtained by doing the convolution between the kernels $W_k$ and input data matrix $X$ with a certain step of $T_c$, and computes the features of individual local region. In conclusion, the feature extracted by the convolution kernel $W_k$ can be expressed as:

$$C_k = [c_{1}^k, c_{2}^k, ..., c_{N_p}^k]^T. \quad (6)$$

The pooling layer served to lessen the amount of neural network parameter and maintain the overall data distribution, which eventually prevent the system from over-fitting problem and enhance system robustness [13, 27]. Both the pooling and convolution operations have similar function, with a difference that pooling operation computes the maximum or average value of the local area. Here, after obtaining the feature $C_k$ from convolution operation we apply a max pooling operation. Let, $H_p$ represent the height of a pooling kernel and $T_p$ represent its step size, then mathematically we compute the output as:

$$M_k = [m_{1}^k, m_{2}^k, ..., m_{N_p}^k]^T, \quad (7)$$

where

$$m_{i}^k = max(c_{i}^k, c_{i+1}^k, ..., c_{i+H_p-1}^k). \quad (8)$$

$$N_p = \frac{N-H+1}{T_p}, \quad (9)$$

The aforementioned process described that a single convolution kernel $W_k$ produces single feature map $M_k$. Finally, after the successful completion of convolution and pooling operations, the entire extracted features are combined in an end-to-end fashion to achieve a single feature vector of the whole input data matrix, which can be mathematically expressed as:

$$F^T = [F_1^T; F_2^T; ...; F_l^T], \quad (10)$$

where $F_i = M_i$, $l$ represents the total number of features.

Finally, for high-level features extraction fully-connected layer is used. Let, $W_F$ represent the weight matrix, in order to calculate the weighted sum of the individual element to get the final feature representation of the input data matrix $S$, mathematically:

$$y = W_F \cdot F + b_f, \quad (11)$$

where $W_F$ represents the learned weight matrix and $b_f$ represents the bias term, however, the weight matrix $W_F$ highlights the significance of every individual feature.

The parameters of the network are updated using backpropagation algorithm in the training process, and the loss function is computed through Smooth-L1 loss called Huber loss:

$$loss(\hat{y}, y) = \frac{1}{n} \sum_{i} z_i, \quad (12)$$
where \( z_i \) can be calculated as:
\[
z_i = \begin{cases} 
0.5(\hat{y} - y)^2, & \text{if } |\hat{y} - y| < 1 \\
|\hat{y} - y| - 0.5, & \text{otherwise.}
\end{cases}
\]

The main choice of selecting Smooth-\( L_1 \) loss is to diminish data sensibility, as outliers greatly affect \( L_2 \) loss. In this paper, we used the minimization of the \( \text{LOSS} \) function over a batch size number of samples for CNN training. The optimization algorithm is ADA-DELTA with 1.0 initial learning rate.

3.2 LSTM-based prediction model

Recurrent Neural Network (RNN) is an artificial neural network architecture, which is very suitable for sequential signals modeling. Mathematically, we described the RNN model in Eq. (8), which can deal with randomly size sequential signals. Though, the system is not able to handle long-range problem efficiently due to the gradient vanishing problem [28]. LSTM model [29], a variant of RNN efficiently solve this problem through rich designed unit nodes. The key architecture refinement of LSTM is the hidden layer unit, which contained four fundamental parts: a cell, an input gate, an output gate and a forget gate. The basic LSTM mathematical model is illustrated as follows:

\[
\begin{align*}
I_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) , \\
F_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) , \\
C_t &= F_t \cdot C_{t-1} + I_t \cdot \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) , \\
O_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) , \\
h_t &= O_t \cdot \tanh(C_t). \\
\end{align*}
\]

where \( I_t \) represents the input gate, which controls the quantity of novel information stored in the memory cell. The function of forget gate \( F_t \) is to discard redundant stored information from the memory cell. Hence, memory cell \( C_t \) is a summation of upcoming and previous information modulated by the input gate and the forget gate, respectively. The output gate \( O_t \) allows the memory cell to have an effect on the current hidden state and output or block its influence. Specifically, we represent the transfer function of LSTM units by \( f_{\text{LSTM}}(\cdot) \). Notice that, while calculating the output at time step \( t \), the information used is based on the input vector at time step \( t \), also includes the information stored in the cells at the previous time step \( t - 1 \). Hence, the output at a given time steps \( t \) can be described as:

\[
y_t = f_{\text{LSTM}}(x_t \mid x_1, x_2, ..., x_{t-1}).
\]

As we have mentioned before, the input matrix \( X \) can be regarded as a sequential signal and the \( i \)-th row \( x_i \) can be viewed as the signal at time step \( i \). In order to model this task on LSTM, we input the \( i \)-th row of \( X \) at time step \( i \). For individual sentence \( S \), we can describe it with a matrix \( S \in \mathbb{R}^{N \times D} \) as described in Eq. [1].

Overall, an RNN contains several network layers, each of which with multiple LSTM units. Let, \( n_j \) represent the number of LSTM units \( U^j \) of \( j \)-th hidden layer, thus the units of \( j \)-th layer can be described as:

\[
U^j = \{ u^j_1, u^j_2, \cdots, u^j_{n_j} \}.
\]

In LSTM first hidden layer, the input of each unit \( u^1_1 \) at time step \( t \) is the weighted sum of the elements in \( x_t \), mathematically as:

\[
u^1_{1,t} = \sum_{k=1}^{d} w^1_{1,k} \cdot a_{j,k} + b^1_{i,t},
\]

where \( w^1 \) represents the learned weights and \( b^1 \) represents the biases. Hence, the output value of \( u^1_1 \) at time step \( t \) is:

\[
o^1_{1,t} = f_{\text{LSTM}}(u^1_{1,t}) = f_{\text{LSTM}}(\sum_{k=1}^{d} w^1_{1,k} \cdot a_{j,k} + b^1_{i,t}).
\]

The vector \( Out^d_t \) is used to represent the output of the \( j \)-th hidden layer at time step \( t \), and each element in \( Out^d_t \) represents the output value of each unit in the \( j \)-th hidden layer at time step \( t \), mathematically:

\[
Out^d_t = f_{\text{LSTM}}(x_t) = [o^d_{1,t}, o^d_{2,t}, ..., o^d_{n,j,t}].
\]
Literature showed that increasing the number of neural network layers, increasing the ability of system to extract features\cite{30}. Therefore, in this paper, the network is stack with multiple layers of LSTM units. The transfer matrix is used to connect the adjacent hidden layers. For instance, the transfer matrix between \(l\)-th layer and \((l+1)\)-th layer can be described as a matrix \(W^l \in \mathbb{R}^{n_l \times n_l+1}\):

\[
W^l = \begin{bmatrix}
w_{1,1}^l & w_{1,2}^l & \cdots & w_{1,n_l+1}^l \\
w_{2,1}^l & w_{2,2}^l & \cdots & w_{2,n_l+1}^l \\
\vdots & \vdots & \ddots & \vdots \\
w_{n_l,1}^l & w_{n_l,2}^l & \cdots & w_{n_l,n_l+1}^l 
\end{bmatrix}
\]  

(20)

Then the input of each unit \(u_t^l\) in the \(l\)-th hidden layer at time step \(t\) is the weighted sum of the output values of the units in the previous layer, mathematically:

\[
u_{i,t}^l = Out_{t-1}^l \cdot W^l \cdot o_{i,t}^{l-1} + b_{i,t}^l.
\]

Furthermore, the output of the \(l\)-th layer at time interval \(t\) is:

\[
Out_t^l = f_{LSTM}(Out_t^l) = [o_{1,t}^l, o_{2,t}^l, \ldots, o_{n_l,t}^l],
\]

\[
o_{i,t}^l = f_{LSTM}(u_{i,t}^l) = f_{LSTM}(\sum_{k=1}^{n_l} w_{i,k}^l \cdot o_{k,t}^l + b_{i,t}^l).
\]

(22)

The aforementioned Eq.\cite{22} illustrated that the output at time step \(t\) is not only based on the input vector of the current time \(x_t\), but also the information stored in the cells at the previous \((t-1)\) intervals. Thus, \(l\)-th hidden layer output at time interval \(t\) can be obtained as a summation of all former \(t\) intervals, such as information fusion of former intervals \(\{x_1, x_2, \ldots, x_t\}\). Moreover, to compute the probability distribution of the \((t+1)\) day, a softmax layer is added after all the hidden layers of the LSTM. Particularly, we introduce the Prediction Weight (PW) as matrix \(W_P \in \mathbb{R}^{n_l \times N}\), mathematically as:

\[
W_P = \begin{bmatrix}
w_{1,1}^{p} & w_{1,2}^{p} & \cdots & w_{1,N}^{p} \\
w_{2,1}^{p} & w_{2,2}^{p} & \cdots & w_{2,N}^{p} \\
\vdots & \vdots & \ddots & \vdots \\
w_{n_l,1}^{p} & w_{n_l,2}^{p} & \cdots & w_{n_l,N}^{p} 
\end{bmatrix}
\]

(23)

where \(N\) represents the total number of days in the dictionary \(D\). Then the learned matrix \(W_P\) is used to compute the score for individual day in the dictionary \(D\) for a time span of 1983-2012. Mathematically, we can define as:

\[
y_i = \sum_{k=1}^{n_l} w_{k,i}^{p} \cdot o_{i,t}^l + b_{i,t}^{p},
\]

(24)

where \(W_P\) represents the learned weight and \(b^{p}\) represents the bias term, the values in weight matrix \(W_P\) reflect the significance of individual feature in \(o^l\). Then we can describe the dimension of output vector \(y\) as \(N\).

Both CNN and LSTM based prediction models use the same Loss function and training algorithm.

4 Experimental Results and Discussion

4.1 Experiment Setting

In this paper, we performed all the experiments on two deep learning models, such as CNN and LSTM, and three machine learning algorithms, such as RF, GBR, and LR. For the CNN model, the input data is fed to a convolution layer with three different kernel height: 3, 5, 7. The width of kernels is equal to the feature number. Each kernel initially generates 100 channels. After the convolution operation, max pooling is applied to each channel. Then the outputs are
concatenated together to the last layer, which is a fully-connected layer with a dimension of $300 \times 1$. The output of which is the predicted flow.

For the LSTM model, we use a bidirectional LSTM with cell number equal to the features number and 300 hidden sizes. Moreover, the max-pooling layer and fully-connected layer follow the same architecture as CNN.

As for the traditional machine learning methods, such as Logistic Regression (LR) [31, 32], Gradient Boosting Tree (GBR) [33] and Random Forest (RF) [34], we used the default setting from package scikit-learn.

4.2 Dataset Split

We divided the dataset into three subsets, such as training set, validation set, and testing set, with a proportion of 7 : 1 : 2. The final result is obtained using a 10 fold cross validation with different random dataset division. Normalization is applied to each feature in the preprocessing.

Mini-batch is used in the training process of CNN and LSTM, with 90 batch size. The optimization algorithm is ADA-DELTA with 1.0 initial learning rate.

4.3 Evaluation Method

The index we adapt to evaluate results is the relative error, which is defined as:

$$re = \frac{1}{n} \sum_{i} \frac{|predict - real|}{real}.$$  \hspace{1cm} (25)

Notice that we have normalized the raw data in preprocessing, so before calculating relative error, the output will be recovered to the initial distribution.
4.4 Hydro-climatic Trends in Soan Basin

Fig. 3 and 4 shows the hydro-climatic trends in sub-basin 1 and sub-basin 2 (all units in mm). It can be observed that precipitation and streamflow of both sub-basin have a strong correlation. The catchment area of the sub-basin 1 is smaller as compared to the sub-basin 2, therefore the streamflow of the sub-basin 1 is more as compared to sub-basin 2. The sub-basin 1 has hilly areas and due to high mountains and slope differences, most of the precipitation is converted into streamflow and also has small water losses. On the other hand, sub-basin 2 has less precipitation as compared to sub-basin 1, which eventually causes less streamflow in sub-basin 2. There is more water loss in sub-basin 2 and has high agriculture and water consumption. Moreover, the water storages and small check dams in sub-basin 2 can cause a reduction in the streamflow. While, on the other hand, sub-basin 1 has no small check dams or retention, which can stop the water and topography is favorable for high streamflow and less water loss.

4.5 Predictive performance of different models.

Supervised learning algorithms are the major benchmark paradigms found in many applications, such as classification, detection and hydrological modeling [35,36,37,38]. Supervised learning algorithms contain three fundamental parts: an input object (vector), a processing unit (hidden layers) and an output value (supervision signal). These learning algorithms produce an inferred function by evaluating the training data, which can be further employed for representing new examples. An optimal setting allows these algorithms to describe the class labels for hidden examples, which help the learning algorithms to generalize for unseen conditions from the training data. Let, given a set of \( n \) training examples \( (x_i, y_i), ..., (x_N, y_N) \) such that \( x_i \) represent the feature vector of \( i^{th} \) example and \( y_j \) is its class label, a supervised learning algorithm search for a function \( g: X \rightarrow Y \), where \( X \) and \( Y \) is the input and output space respectively. The function \( g \) is an element of the hypothesis space of possible functions \( \Theta \). Under certain circumstances \( g \) is represented as a scoring function \( f: X \times Y \rightarrow \mathbb{R} \), such that, it return the value of \( y \) that gives the maximum score: \( g(x) = \arg \max_y f(x, y) \).

Suppose, \( F \) represents the scoring functions.

Table 2: Relative Error of different Deep learning and Machine learning methods.

| Model | Mean Relative Error(%) | Standard Deviation |
|-------|-------------------------|--------------------|
| LR    | 28.0377                 | 1.9769             |
| GBR   | 27.4681                 | 1.8108             |
| RF    | 31.5304                 | 1.9430             |
| CNN   | 26.8145                 | 1.6070             |
| LSTM  | 25.0984                 | 1.9182             |

Even though, \( G \) and \( F \) can be any space of functions, many learning algorithms such as CNN and LSTM are probabilistic models where \( g \) takes the form of a joint probability model \( g(x, y) = P(x, y) \), and logistic regression is a conditional probability model where \( f \) takes the form of a conditional probability model \( f(x) = P(y|x) \).

In this paper, we evaluated five different learning models for measuring predictive performance – CNN, LSTM, RF, GBR, and LR - to predict the stream flow based on hydrological variables. These different learning models were chosen to test the performance of a wide variety of learning algorithms on streamflow prediction problem. The inclusion of both statistical models, such as LR, GBR, and RF, and deep learning models, such as LSTM and CNN, help to ensure that the predictive performance is tested on a wide variety of learning algorithms. Statistical models are used in a variety of fields, such as medical disease diagnosis, machine learning and social sciences for prediction. Moreover, these models itself simply model the probability of output in terms of input and do not perform statistical classification. CNN and LSTM models can provide robust predictions, due to the transformations of the input space in the inner layers [39]. The outcome of each algorithm was summarized using Mean Relative error (MRE) of successful prediction (error rate %), as shown in Table 2. Moreover, the advantage of deep learning models (CNN and LSTM) over statistical models is that deep learning models continue to improve efficiency as the number of datasets grows. Whereas, statistical models plateau at a certain level of performance when you add more examples and training data to the network.

Table 2 shows the parameter settings and numerical results obtained using different algorithms including the proposed algorithms. It is clear from the results that the LSTM surpasses other benchmark algorithms in solving streamflow prediction. It demonstrates that the Random Forest (RF) approach achieved the highest MRE (31.53%) on the streamflow prediction dataset. Followed by Improved Linear Regression (LR) with mean relative error (MRE) of
Figure 5: Prediction results using LSTM. The horizontal axis shows the real stream flow data, whereas the vertical axis shows the predicted results obtained through LSTM.

Figure 6: Prediction results using CNN. The horizontal axis shows the real streamflow data, whereas the vertical axis shows the predicted results obtained through CNN.

Figure 7: Prediction error changes with days.
28.03%. The Gradient Boosted Regression (GBR) algorithm and CNN have similar MRE of 27.46% and 26.81% respectively (the overall difference was no more than 0.6%). The lowest MSE is achieved by the LSTM method, which is 25.09%. Furthermore, LSTM also shows high predictive accuracy despite having a small error, as shown in Fig. 5. It shows the real stream flow plotted against the predicted flow by LSTM with a 45° reference line. Followed by CNN, which also shows a good predictive accuracy as shown in Fig. 6. Hence, we concluded that the LSTM algorithm with the lowest MRE achieves the highest prediction rate, which needs less training time to obtain an optimum solution.

The accuracy of forecasting traffic is related to the forecast duration. For example, according to the data of the first five days or the first 10 days or the first month, the accuracy of the next day’s traffic forecast is different. Therefore, the abscissa of this graph is how many days before we forecast the next day. The ordinate is the accuracy of the prediction. The main purpose is to find the best forecast duration. Based on the data in this Fig. 7, we find that when the predicted number of days is 7, the predicted error is the smallest. So our model was finally determined to predict river flow of the next day based on data from the previous 7 days.

5 Conclusions

Two deep learning techniques and three machine learning algorithms are proposed to predict stream flow, given the present climate conditions. These different learning models were chosen to evaluate the performance of a wide variety of learning algorithms on streamflow prediction problem. The inclusion of both statistical models, such as LR, GBR, and RF, and deep learning models, such as LSTM and CNN, help to ensure that the predictive performance is tested on a wide variety of learning algorithms. The results showed that the LSTM, an artificial neural network based method, outperforms other conventional and machine-learning algorithms for predicting stream flow.

To our best knowledge, this is the first work to introduce five different deep learning and machine learning algorithms for streamflow prediction problem. Furthermore, we believe that the proposed predictive algorithms serve as a reference tool for researchers and engineers to help the design and implementation of improving evaluation protocols.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (No. 61801008, U1705261, U1405254, U1536115, U1536207), National Key R&D Program of China (No. 2018YFB0803600, 2016YFB0801301), Beijing Natural Science Foundation National (No. L172049), Scientific Research Common Program of Beijing Municipal Commission of Education (No. KM201910005025).

References

[1] David Labat, Yves Goddéris, Jean Luc Probst, and Jean Loup Guyot. Evidence for global runoff increase related to climate warming. *Advances in Water Resources*, 27(6):631–642, 2004.
[2] Nicola Gedney, PM Cox, RA Betts, O Boucher, C Huntingford, and PA Stott. Detection of a direct carbon dioxide effect in continental river runoff records. *Nature*, 439(7078):835, 2006.
[3] TR Petty and P Dhingra. Streamflow hydrology estimate using machine learning (shem). *JAWRA Journal of the American Water Resources Association*, 54(1):55–68, 2018.
[4] Gang Zhao, Huilin Gao, and Lan Cuo. Effects of urbanization and climate change on peak flows over the san antonio river basin, texas. *Journal of Hydrometeorology*, 17(9):2371–2389, 2016.
[5] Zhi Li and Jiming Jin. Evaluating climate change impacts on streamflow variability based on a multisite multivariate gcm downscaling method in the jing river of china. *Hydrology and Earth System Sciences*, 21(11):5531–5546, 2017.
[6] Bibi S Naz, Shih-Chieh Kao, Moetasm Ashfaq, Huilin Gao, Deeksha Rastogi, and Sudershnan Gangrade. Effects of climate change on streamflow extremes and implications for reservoir inflow in the united states. *Journal of Hydrology*, 556:359–370, 2018.
[7] Manish Kumar Goyal, CSP Ojha, RD Singh, PK Swamee, et al. Application of artificial neural network, fuzzy logic and decision tree algorithms for modelling of streamflow at kasol in india. *Water Science and Technology*, 68(12):2521–2526, 2013.
[8] Amir Mosavi, Pinar Ozturk, Shahab Shamshirband, Hai Thanh Nguyen, and Kwok-wing Chau. Flood prediction using machine learning. *Literature Review, Engineering Applications of Computational Fluid Mechanics*, 2018.
[9] Caihong Hu, Qiang Wu, Hui Li, Shengqi Jian, Nan Li, and Zhengzheng Lou. Deep learning with a long short-term memory networks approach for rainfall-runoff simulation. *Water*, 10(11):1543, 2018.

[10] Dongwoo Jang, Hyoseon Park, and Gyeewoon Choi. Estimation of leakage ratio using principal component analysis and artificial neural network in water distribution systems. *Sustainability*, 10(3):750, 2018.

[11] Chen Liang, Hongqing Li, Mingjun Lei, and Qingyun Du. Dongting lake water level forecast and its relationship with the three gorges dam based on a long short-term memory network. *Water*, 10(10):1389, 2018.

[12] Sills J. et al. Nextgen voices: Unique identities. *Science*, 363(6428):702–702, 2019.

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[16] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

[17] Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

[18] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[19] Winston Yu, Yi-Chen Yang, Andre Savitsky, Donald Alford, Casey Brown, James Wescoat, Dario Debowicz, and Sherman Robinson. The Indus basin of Pakistan: The impacts of climate risks on water and agriculture. *The World Bank*, 2013.

[20] Muhammad Shahid, Hamza Farooq Gabriel, Amjad Nabi, Sajjad Haider, AS Khan, and AMS Shah. Evaluation of development and land use change effects on rainfall-runoff and runoff-sediment relations of catchment area of simly lake pakistan. *Life Science Journal*, 11(7s), 2014.

[21] A Ashfaq, M Ashraf, and A Bahzad. Spatial and temporal assessment of groundwater behaviour in the soan basin of pakistan. *University of Engineering and Technology Taxila. Technical Journal*, 19(1):12, 2014.

[22] Muhammad Shahid, Zhentao Cong, and Danwu Zhang. Understanding the impacts of climate change and human activities on streamflow: a case study of the soan river basin, pakistan. *Theoretical and Applied Climatology*, 134(1-2):205–219, 2018.

[23] Hosea M Mwangi, Stefan Julich, Sopan D Patil, Morag A McDonald, and Karl-Heinz Feger. Relative contribution of land use change and climate variability on discharge of upper mara river, kenya. *Journal of Hydrology: Regional Studies*, 5:244–260, 2016.

[24] Andrea Ficchi, Luciano Raso, Pierre-Olivier Malaterre, David Dorchies, Maxime Jay-Allemand, Francesca Pianosi, Peter-Jules van Overloop, and Guillaume Thirel. Short term reservoirs operation on the seine river: Performance analysis of tree-based model predictive control. 2014.

[25] Sadaqat Rehman, Shanshan Tu, Obaid Rehman, Yongfeng Huang, Chathura Magurawalage, Chin-Chen Chang, et al. Optimization of cnn through novel training strategy for visual classification problems. *Entropy*, 20(4):290, 2018.

[26] Sadaqat ur Rehman, Shanshan Tu, Yongfeng Huang, Guojie Liu, et al. Csf: A novel unsupervised convolution neural network approach for visual pattern classification. *AI Communications*, 30(5):311–324, 2017.

[27] Y-Lan Boureau, Jean Ponce, and Yann LeCun. A theoretical analysis of feature pooling in visual recognition. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 111–118, 2010.

[28] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

[29] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[30] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks (2013). *arXiv preprint arXiv:1311.2901*, 2013.

[31] Johan AK Suykens and Joos Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.

[32] Frank E Harrell. Ordinal logistic regression. In *Regression modeling strategies*, pages 311–325. Springer, 2015.
[33] Jerome H Friedman. Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367–378, 2002.

[34] Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[35] Roshanak Nateghi, Seth D Guikema, and Steven M Quiring. Comparison and validation of statistical methods for predicting power outage durations in the event of hurricanes. *Risk Analysis: An International Journal*, 31(12):1897–1906, 2011.

[36] Roshanak Nateghi, Jeremy D Bricker, Seth D Guikema, and Akane Bessho. Statistical analysis of the effectiveness of seawalls and coastal forests in mitigating tsunami impacts in iwate and miyagi prefectures. *PloS one*, 11(8):e0158375, 2016.

[37] B Grizzetti, A Pistocchi, C Liquete, A Udias, F Bouraoui, and W Van De Bund. Human pressures and ecological status of european rivers. *Scientific reports*, 7(1):205, 2017.

[38] Kazuya Nishina, Mirai Watanabe, Masami K Koshikawa, Takejiro Takamatsu, Yu Morino, Tatsuya Nagashima, Kunika Soma, and Seiji Hayashi. Varying sensitivity of mountainous streamwater base-flow no-3 concentrations to n deposition in the northern suburbs of tokyo. *Scientific reports*, 7(1):7701, 2017.

[39] Stefano Lancini, Marco Lazzari, Alberto Masera, and Paolo Salvaneschi. Diagnosing ancient monuments with expert software. *Structural engineering international*, 7(4):288–291, 1997.