Application of Artificial Neural Network in Streamflow Forecasting

Mian Xiao
Civil & Environmental Engineering
Stanford University
mx18@stanford.edu

Shanni You
Civil & Environmental Engineering
Stanford University
shanni30@stanford.edu

Abstract

Streamflow forecast is a complicated but highly useful technique for water resources planning and development. In this study, two types of the popular Artificial Neural Networks(ANNs) models, the standard feed-forward and the LSTM model, are implemented to validate the feasibility of modern machine learning techniques for complicated hydrologic forecast problems. The results confirmed that ANN is an important alternative to conceptual models and it can be used when the range of collected dataset is short of diverse feature, and only consider precipitation and historical runoff.

1. Introduction and related works

In the field of hydraulic forecast, streamflow forecast is one of the most important elements, with which we can further determine the occurrence of a flood or the transportation of pollutants. However, streamflow process is very hard to predict because it is influenced by various factors, such as precipitation, historical flow and sewage outlets. To investigate the runoff amount in streamflow, scientists developed many physical models decades ago. They usually claim that the current runoff is a function of historical runoff and precipitation. One popular physical model is the probability-distributed model [1], which incorporates the exponential distribution to calculate the incremental runoff induced by precipitation. The total runoff is then assumed to be a sum of the incremental runoff and the propagated historical runoff.

In addition to physical models, hydrologic scientists utilize empirical statistical models, which produce satisfactory results without explicitly representing the intrinsic mechanism. In fact, the accuracy of empirical models is dominated by the assumption. However, it is nearly impossible to make a correct assumption due to the intrinsic complexity of the this problem. This means that empirical models may fail under extreme conditions that invalidate their assumptions.

Modern machine learning techniques, like artificial neural networks (ANN) with embedded multi-layer structure, can represent much more sophisticated nonlinear functions than traditional empirical models can do. As a result, ANN exhibits a great potential in capturing the complicated behaviors of the streamflow series. In 1995, Hsu et al. implemented a three-layer standard feed-forward neural network for runoff forecast [2]. And their work demonstrated that ANN achieves better performance than physical models. Minns et al. tested ANN models with both one and two hidden layers for runoff prediction [3], and they concluded that ANN with more than 1 hidden layer may overestimate the degree of nonlinearity in the streamflow process. Recently, researchers introduced sequence dependency into neural network. Kratzert et al. applied the state-of-art "Long Short-Term Memory" (LSTM) network to address the sequential order in the inputs [4]. Their LSTM implementation makes satisfactory prediction except at the occurrences of runoff peaks. **In this study, both standard feed-forward NN and LSTM are implemented and tested for runoff forecast.**

2. Methodology

2.1. Input feature identification

Derived from physical models, the current runoff amount $Q(t)$ is assumed to be a function of a historical rainfall series and the historical runoff amount [2], i.e.:

$$Q(t) = f\left\{R(t-1), ..., R(t-m), Q(t-1), ..., Q(t-n)\right\} \tag{1}$$

where $Q(t)$, $R(t)$ are the runoff amount and the rainfall at a specific time t. In this study, we will adopt this assumption, and use $R(t-1), ..., R(t-m)$ and $Q(t-1), ..., Q(t-n)$ as input features.

m, n are customized parameters, and they illustrate the dependency on historical data series for our current prediction, which definitely affect the predicting accuracy. A reasonable choice for m and n is $m, n \leq 10$ to avoid over-dependency. This study choose both m and n from the range of 2 to 7. In the numerical experiments, multiple combinations of m and n are tested to determine an optimal input structure.

2.2. Neural network structures

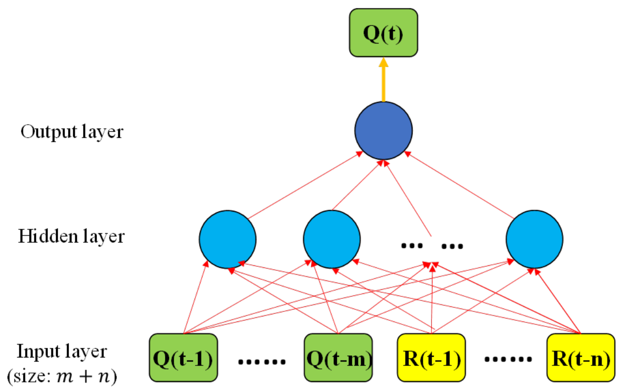

Figure 1. Standard network model design

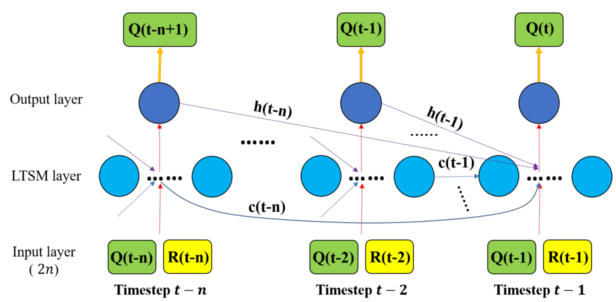

Figure 2. LSTM network model design

Adopting the work of Hsu et al., this study primarily uses a **three layer, fully connected feed-forward neural network**, as shown in Fig 1. The input layer size is determined by the choice of m, n; the output layer size is always 1; the hidden layer size is one of the hyper-parameters to optimize. Since m, n are fairly small numbers, there is no point setting up a huge hidden layer. Naturally, the hidden layer size should be comparable with the input layer size for a small network, and we will use a hidden size smaller than 32. As a convention, the hidden layer activation is chosen to be the sigmoid function, while no activation is applied on the output layer [2],[3]. The network architecture can be describe with the following formula:

$$h_t = \sigma(W^{[1]}x_t + b^{[1]})$$

$$y_t = W^{[2]}h_t + b^{[2]}$$

where h_t is the hidden layer activation and y_t is the prediction of the entire network. $\sigma(z) = \frac{1}{1+\exp(-z)}$ is the mathematical expression for the sigmoid function.

The LSTM network maintains the three-layer structure of the standard version, with the hidden layer replaced by an LSTM layer. But the input for the LTSM layer must be organized by time sequence, then the valid input at each step is just the runoff and rainfall at the corresponding time, as shown in Fig 2. This implies $m = n$.

The multi-timestep dependence is introduced by instructing the LSTM layer to accept the latent feedback and cell status variables from several previous timesteps, and process them with the same four-gate LSTM architechture as is usually used in linguistic learning. The following section focuses on the illustration of the three-gate mechanism of individual LSTM cell.

First, the four gate variables: $f_{t,k}$ (forget), $\tilde{c}_t$ (input), i_t (block), o_t (output) are computed as follows:

$$f_{t,k} = \sigma(W_{f,k}x_t + V_{f,k}c_{t-k} + b_{f,k}) \tag{2}$$

$$i_t = \sigma(W_i x_t + \sum_{k=1}^{n} V_{i,k}c_{t-k} + b_i) \tag{3}$$

$$o_t = \sigma(W_o x_t + \sum_{k=1}^{n} V_{o,k}c_{t-k} + b_o) \tag{4}$$

$$\tilde{c}_t = \tanh(W_{\tilde{c}}x_t + \sum_{k=1}^{n} V_{\tilde{c},k}c_{t-k} + b_{\tilde{c}}) \tag{5}$$

where $W_{f,k}$, $W_{\tilde{c}}$, W_i, W_o are the weights to the inputs for the four gates, $V_{f,k}$, $V_{\tilde{c},k}$, $V_{i,k}$, $V_{o,k}$, are the weights to the kth latent cell state, $b_{f,k}$, $b_{\tilde{c}}$, b_i, b_o are the gate bias.

Next, obtain the current cell state as:

$$c_t = i_t \odot \tilde{c}_t + \sum_{k=1}^{n} f_{t,k} \odot c_{t-k} \tag{6}$$

Finally, with the cell state and the output gate parameters in eqn (4), determine the final hidden layer output h_t as follows:

$$h_t = \tanh(c_t) \odot o_t \tag{7}$$

h_t will goes into the dense output layer and be utilized to compute the final prediction with a linear combination: $y_t = W_d h_t + b_d$.

2.3. Dataset and Features

Ten years of the daily runoff is collected at the **leaf river basin near Collins, LA**, and corresponding rainfall at Collins is also collected. The time period is 2003 - 2012. Since some of the data have invalid value such as None, the raw runoff and rainfall data is cleaned and separated into a training set and a validation set. The data of year 2003 - 2007 is utilized for training purpose, while the rest is categorized as validation data.

The goal is to predict the runoff amount, so this type of problem should be categorized as regression problem. For regression problems, it is useful to normalized the input into $[0, 1]$ to avoid numerical issues caused by different scales over input features. The runoff and rainfall series are normalized with (divided by) their maximum value, which is 29700 ft^3/s and 6.8 inch, respectively. While predicting, the output is amplified with its corresponding scale to retrieve the true runoff amount.

3. Experiments and Results

3.1. Experimental Setting

The Root Mean Square Error(RMSE) is selected as the primary metric for performance evaluation in all experiments:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(r_i - r_i')^2}{n}}$$

where r_i denotes the real runoff value, r_i' denotes its predicted value, and n is the total number of dates to be predicted.

For hyperparameters, we fixed maximum epoch at 200, learning rate at 0.009 for Standard feed-forward network and 0.001 for LTSM model, which we find it is the best for the model respectively.

We then retrain the models with the selected hyperparameters. We then use the "best model" with the minimum RMSE on the validation set.

The general framework of our code is based on tensorflow. We also utilized the $mean - squared - error$ functions from scikit-learn in our code.

3.2. Standard feed-forward network and LSTM network Results

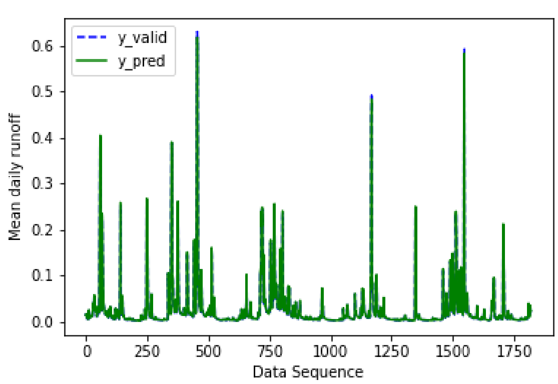

Figure 3. Best model of Standard feed-forward

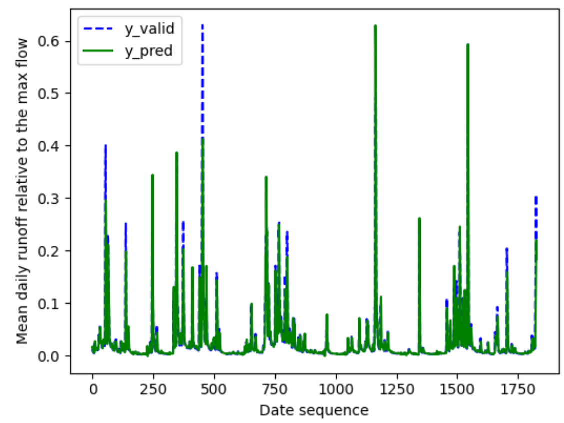

Figure 4. Best model of LSTM

The best standard network and the best LSTM simulation is presented here. Additionally, to elaborate the differential behavior at peak runoff and non-peak runoff prediction, the validation series is further divided into two subsets with threshold = 1500. Then the RMSE on peak runoff predictions ($Q_{valid} > 1500$) and the RMSE on non-peak runoff predictions ($Q_{valid} \leq 1500$) are evaluated separately.

Table 1. RMSE relative to the max flow on best models

RMSE on validation	Standard	LSTM
at peaks	0.63%	6.78%
at non-peaks	0.07%	0.54%
overall	0.23%	2.43%

Results show that the performances of both models are generally satisfactory. The RMSE (total) relative to the maximum flow presented in Table 1, 0.23%, is smaller than the RMSE available in other references, e.g., 1.65% from the experiments conducted by Hsu et al. [2]. For most dates of the validation set where the runoff amount is small, the predicted runoff is very close to the actual one. While at peak values, the error is more significant, particularly for the LSTM model. Table 1 also indicates that the RMSE on validation mostly comes from the misprediction at peak runoff. **In terms of the general evaluation metrics (RMSE all over the validation set), the standard network performance is better than that of the LSTM model.**

From observation of the experiment results, the LSTM model behaivor is more un-robotic, since the maximum underestimation and overestimation is much higher than their standard counterpart. One possible reason for the unsatisfactory performance of LSTM might come from the LSTM cell architecture. The intrinsic learning mechanism of LSTM, which is developed from linguistic machine learning, is not capturing the essential sequential dependency in the rainfall-runoff series.

In practice, underestimation is highly unfavorable because it leads to insufficient preparation for the potential flood hazard. To deal with this in an engineering sense, the predicted flooding runoff is usually multiplied with a safety factor (> 1) to avoid underestimation. This works for the standard network where the prediction is satisfactory enough. However, similar approach is not suitable for the LSTM model because of its inaccuracy and instability at high flow amounts.

3.3. Parametric Study

In our case of study, we discovered that the learning rate have the dominate influence of our simulation. With large learning rate, the prediction is barely converge to the right result. And if the learning rate is too small, it does not catch the characteristic of the data set, and it also have trouble converging to the expectation within a limited number of epochs. For that it is not point in discussing the result, we hence decided to fixed the learning rate in this problem to further study other parameters. It is observed that the learning rate is better in range of (0.008,0.01) for the standard feed-forward network. For the LSTM network, the learning rate is expected to smaller (around 0.001).

The major parameters of interest are the lag-day features and the network hyper-parameters including: hidden size and learning rate. However, a better estimation of the peak-flow value or low-flow value does not necessarily imply a overall satisfactory prediction evaluated by the overall RMSE. The lag-days for rainfall (n) and runoff (m) are the first to optimize, since they determine the input features. Patterns with $n - m = 1, 2, 3$ are constructed. After several experiment attempted, the lag-days for runoff is usually preferred larger than that of rainfall to securely predict the extreme value. Four plots of RMSE w.r.t. the lagdays-rainfall and hidden layer size are presented in Fig 3.

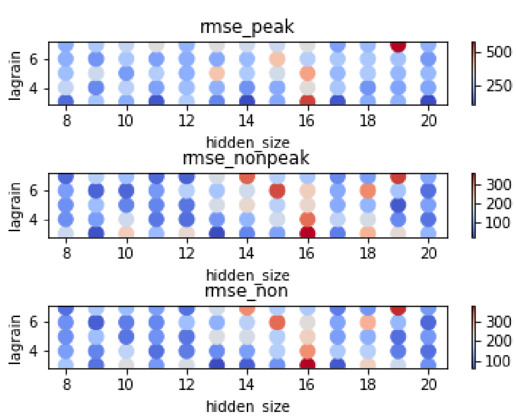

Figure 5. RMSE variation for the standard NN with $n - m = 1$

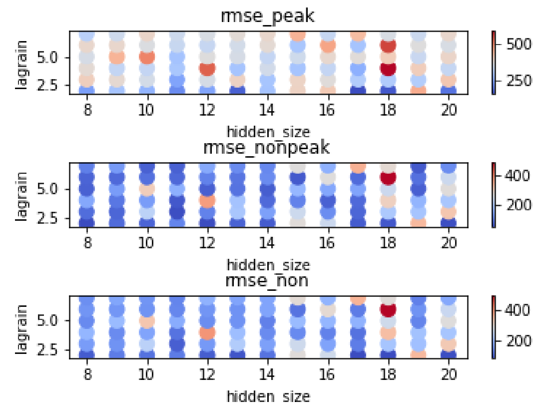

Figure 6. RMSE variation for the standard NN with $n - m = 2$

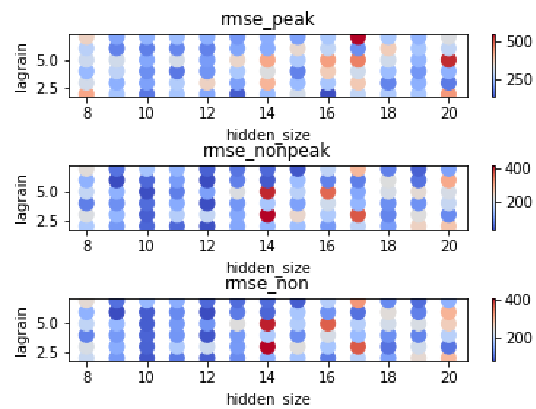

Figure 7. RMSE variation for the standard NN with $n - m = 3$

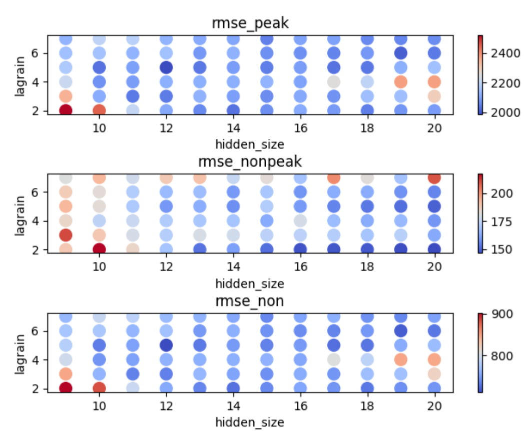

Figure 8. RMSE variation for the LSTM network

Due to the intrinsic high nonlinearity of the neural network model, the variation of RMSE in the parameter space suffers from significant oscillation. The occurrence of local maxima, around which the fluctuation is generally very violent, usually takes place at a large hidden size with a small input size, e.g. lag-rainfall = 3 and hidden size = 16 in Fig 5. This indicates the potential of overfitting using a large hidden size when the input feature size is relatively small. However, in the parameter space of the LSTM network, the overfitting effect is not very significant until the hidden reaches the upper limit of interest. And the variation of total RMSE is more stable around the center of the parameter space for LSTM model.

Table 2. parameters combination for the best models

Type of NN	Standard	LSTM
hidden size	12	12
learning rate	0.009	0.001
lagdays-rainfall	3	5
lagdays-runoff	4	5

The parameter combination that optimize the overall RMSE is presented in Table 3. The primary criterion for identifying the best model is the RMSE all over the validation set. The criterion with "non-peak" RMSE is also studied, but it is not satisfactory since a globally minimized "non-peak" RMSE does not necessarily implies the a globally minimized total RMSE. The best parameter combination for standard network use less input feature but equivalent hidden size. The difference in total input size is 3, which not very large. This suggests that the optimal network size parameters for the two models are relatively close. However, the optimal learning rate varies a lot due to the difference in activation and back-propagation algorithm.

4. Conclusions

In this study, the standard network produces very satisfactory prediction. Therefore it validates the feasibility of standard NN approach for streamflow forecasting problem.

It is amazing that the standard NN maintains a good accuracy with merely historical runoff and rainfall data. So it becomes particularly useful if the data for prediction is very limited. The standard NN model is not difficult to implement, too. Hence, this model can serve as an important alternative to other practical forecasting models, and the authors are confident about the potential application of ANN in other areas of hydrologic forecast.

In terms of peak runoff prediction, the performance of NN is not as accurate as that of non-peak runoff. But the deviation from actual runoff is very limited (roughly $< 10\%$) for the standard network. So an engineering remedial is available which multiply the predicted runoff by a safety factor to ensure security.

The accuracy and stability of of LSTM network prediction is not as satisfactory as their standard counterpart, and it fails at peak runoff values. A potential improvement is to customize the design of LSTM architecture to better capture the time-dependency mechanism for this type of problem.

5. Future works

At present, the study is limited in a specific area (Collins). However, in practice, it does not make sense to use the model developed at Collins to make prediction for other stations, because the weather and other conditions changes a lot. As a results, there is a rising demand for a generic model that works on a large region.

To make prediction on a large geographical scale, more features like geographical and environmental conditions should be included in the neural network input. Moreover, a more sophisticated neural network architecture design is required to deal with increasing nonlinearity introduced by new features.

6. Contribution

Both team members participate actively in this project. Shanni You performed the majority of the standard feed-forward implementation and training experiments, while Mian completed the formulation and implementation of the LSTM network. Both members write part of the final report, and Mian contributes to the modification and polish for the final version of the report.

References

[1] Moore, R. J. "The probability-distributed principle and runoff production at point and basin scales." Hydrological Sciences Journal 30.2 (1985): 273-297.

[2] Hsu, Kuo-lin, Hoshin Vijai Gupta, and Soroosh Sorooshian. "Artificial neural network modeling of the rainfall-runoff process." Water resources research 31.10 (1995): 2517-2530.

[3] Minns, A. W., and M. J. Hall. "Artificial neural networks as rainfall-runoff models." Hydrological sciences journal 41.3 (1996): 399-417.

[4] Kratzert, Frederik, et al. "Rainfall–runoff modelling using long short-term memory (LSTM) networks." Hydrol. Earth Syst. Sci 22.11 (2018): 6005-6022.

[5] A.R.Ghumman, Yousry M.Ghazaw, et al. "Runoff forecasting by artificial neural network and conventional model." Alexandria Engineering Journal 50.11(2011):345-350

[6] S.RIAD, J.MANIA, et al. "Rainfall-Runoff Model Using an Artificial Neural Network Approach." Mathematical and Computer Modelling 40(2004):839-846