River Discharge Modeling Using Generalized Artificial Neuron Model

Seema Narain[1] and Ashu Jain[2]
[1]*Civil Academic Officer, College of Military Engineering, Pune, India*
[2]*Professor, Indian Institute of Technology Kanpur, India*

***Abstract*—Most applications of neural networks in hydrology employ the McCulloch and Pitts' Artificial Neuron (MPAN) that was proposed in early 1940s. This paper presents the results of a preliminary investigation of the use of a new artificial neuron called *Generalized Neuron* (GN) for hydrological modeling. The GN model offers many advantages over the traditional MPAN including but not limited to: (a) capability to model the non-linearity in a physical system through non-linear discriminant function, and (b) there is no need to determine the number of hidden layers and consequently the number of hidden neurons as a single GN is capable of modeling a complex physical system. Two neural network models are presented here: (a) a traditional feed-forward neural network model trained using back-propagation algorithm, and (b) a GN model. The rainfall and flow data from Kentucky River catchment were employed to develop the neural network models. A wide range of performance statistics was used to evaluate the ANN model performance. The results of the study present here indicate that the GN model has tremendous potential for application in hydrological modeling.**

I. INTRODUCTION

Hydrological modeling is important in planning, design, and operation of water resources systems. The physical processes describing a hydrological system are very complex, dynamic, and non-linear in nature that are difficult to understand and model. Historically, hydrologists have employed conceptual methods that incorporate the physics of the system in modeling, or empirical approaches that do not consider the underlying physics while modeling. There has been a tremendous growth in the use of artificial neural networks (ANNs) for the modeling hydrological systems in the last fifteen years or so. The ANN solutions have been found promising in modeling the complex hydrological systems as compared to the traditional conceptual or empirical approaches. The ANN applications to hydrological modeling range from simple application of ANNs (Mins and Hall, 1996; Shamseldin et al, 1997; Campolo et al., 1999; Jain and Indurthy, 2003) to complex ANN models involving specialized efforts such as the use of genetic algorithms for training of neural networks (Jain and Srinivasulu, 2004); developing hybrid neural networks (Chen and Adams, 2006); and data-decomposition and integration of techniques (Abrahart and See, 2000; and Jain and Srinivasulu, 2006). Some recent studies in hydrology focusing on the integration of conceptual and ANN methods or using different training methods (viz. genetic algorithms) emphasize the need of developing more robust and efficient hydrological models capable of producing more accurate flow forecasts.

Most of the ANN applications to hydrology employ the McCulloch and Pitts' Artificial Neuron (MPAN) that was proposed in early 1940s. Although the MPAN has been found to function very well in most engineering applications, the increased demands on the more and more accurate estimations of future variables requires the researchers to possibly look beyond MPAN in the search of more efficient hydrological neural network models. The conventional neural networks using MPANs as building blocks suffer from several short-comings: (a) the training time for the conventional neural network is too long, which results in the slower response of the system, (b) the number of hidden layers and hidden neurons make the model too complex apart from being determined through a trial and error basis, (c) the existing ANN models perform only the operation of summation of its weighted inputs leading to linear discriminant functions, and (d) the training methods employed, usually back-propagation (BP) algorithm due to Rumelhart et al. (1986), are slow, vulnerable of getting stuck in local minima, and can be biased towards a particular magnitude (low, medium, or high flows). This paper presents the results of a preliminary investigation of the use of a new artificial neuron called Generalized

Neuron (GN). The GN differs from the traditional MPAN in many ways including its capability to have non-linear discriminant function. An ANN model based on GN is developed using the rainfall and flow data derived from the Kentucky River Basin in USA. A feed-forward type neural network model trained using BP method is also developed. The performance of the GN model is compared with the traditional ANN model developed using MPANs as building blocks in terms of a variety of error statistics.

II. STUDY AREA AND PERFORMANCE STATISTICS

Study Area and Data

The data derived from the Kentucky River Basin were employed to train and test all the models developed in this study. The Kentucky River Basin (see Figure 1), encompasses over 4.4 million acres (17,820 km^2) of the state of Kentucky. Forty separate counties lie either completely or partially within the boundaries of the catchment. The Kentucky River is the sole source for the several water supply companies of the state. The drainage area of the Kentucky River at Lock and Dam 10 (LD10) near Winchester, Kentucky is approximately 10,240 km^2 and the time of concentration of the catchment is approximately two days. The data used in this study include the average daily streamflow (m^3/s) from Kentucky River at LD10 and LD11 (near Heidelberg), and the daily average rainfall (mm) from the five rain gauges (Manchester, Hyden, Jackson, Heidelberg, and Lexington Airport) scattered throughout the Kentucky River catchment. A total length of the data of 26-years (1960-1989 with data in some years missing) was available. The data were divided into two sets: a training data set consisting of the daily rainfall and flow data for thirteen years (1960-1972), and a testing data set consisting of the daily rainfall and flow data of thirteen years (1977-1989).

Figure 1: Kentucky River Basin

Model Performance

The performance of all the models developed in this study was evaluated using eight different standard statistical measures. These are summing square error (SSE), Nash-Sutcliffe efficiency (E), Pearson coefficient of correlation (R), average absolute relative error (AARE) and threshold statistic (TS). The equations to compute these statistics are provided below. Where XO is the observed value of the variable, XE is the estimated value of the variable from a model, $\overline{XO}$ is the average observed value of the variable, $\overline{XE}$ is the average estimated value of the variable, n_x is the number of data points estimated for which the absolute relative

$$SSE = \sum (XE - XO)^2 \quad (1)$$

$$E = 1 - \frac{\sum (XE - XO)^2}{\sum (XO - \overline{XO})^2} \quad (2)$$

$$R = \frac{\sum (XO - \overline{XO})(XE - \overline{XE})}{\sqrt{\sum (XO - \overline{XO})^2 \sum (XE - \overline{XE})^2}} \quad (3)$$

$$TS_x = \frac{n_x}{N} \times 100\% \quad (4)$$

$$AARE = \frac{1}{N} \sum_{t=1}^{N} \left| \frac{XO(t) - XE(t)}{XO(t)} \right| X\ 100\% \quad (5)$$

error (ARE) is less than x%, N is the total number of data points estimated, and all the summations run from 1 to N. The value of x of 1%, 10%, 50%, and, 100% were considered in this study to compute threshold statistics. Values of SSE and AARE close to 0.0 represent good model performance. The TS can range between 0% and 100% with higher values representing good model performance. Coefficient of correlation can range between -1.0 and +1.0 with magnitudes close to 1.0 meaning good linear dependence between observed and modeled outputs. The Values of Nash efficiency can range between $-\infty$ and +1.0 with values close to 1.0 being very good. The values of E equal to 0.0 means the model is as good as the mean values.

III. MODEL DEVELOPMENT

Two types of neural network models are developed. The first is the feed-forward type neural network model trained using BP algorithm. It consists of three layers: an input layer, a hidden layer, and an output layer (see Figure 2). The inputs to the ANN are average rainfall at various time steps (P_t, P_{t-1}, and P_{t-2}); flow in Kentucky River at LD10 in the past ($Q10_{t-1}$, and $Q10_{t-2}$); and flow in Kentucky River at an upstream gauging station LD11 at various time steps ($Q11_t$, $Q11_{t-1}$, and $Q11_{t-2}$). The ANN model thus developed would require forecasts of two key inputs P_t, and $Q11_t$. The output from the ANN is $Q10_t$ being modeled.

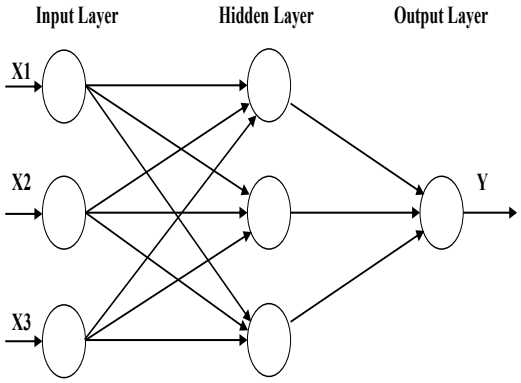

Figure 2: Structure of a feed-forward ANN

The number of neurons in the hidden layer was determined using a trial-and-error procedure. The BP method with momentum factor was used to train various ANN architectures (with hidden neurons varying from 1 to 20) and the best ANN architecture in terms of various error statistics during training was selected. Based on this method, the ANN architecture 8-5-1 was found suitable for the modeling of daily flow in Kentucky River at LD10. The results in terms of various error statistics during training and testing are presented in Table 1.

Generalized Neuron Model

The ANN model presented above uses MPAN as a building block. The ANN models using MPAN model suffers from certain weaknesses described earlier. In this paper, a new generalized neuron model is proposed, which overcomes some of the drawbacks of conventional neural network employing MPAN. The GN model incorporates non-linearities present in the system through the non-linear discriminant function. Also, there is no need of the selection of number of hidden layers and the number of hidden neurons. This reduces the complexity and dimensionality of the overall ANN model. A schematic of the GN model is presented in Figure 3.

The GN model receives inputs from external source and gives output to the external source like in an ANN model built using MPANs. The GN model differs in its structure that is actually responsible for capturing the complex input-output relationships. The GN model consists of five distinct components: an aggregation of weighted inputs (Σ_1) that is similar to MPAN (linear discriminant function), a product of weighted inputs (non-linear discriminant function, $\prod$), the corresponding outputs computed using independent activation functions ($\int$ and Ω, respectively), and an assimilation function (Σ_2) that provides output to the external source.

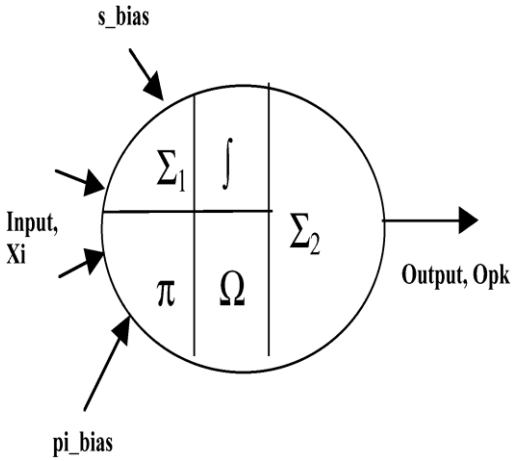

Figure 3: Generalized Neuron model

The bias terms are added to both linear and non-linear discriminant functions like in MPANs. The weighted inputs to the GN are represented as follows:

$$NetL_j = \sum WL_{ij} X_i + s\_bias \quad (6)$$

$$NetNL_j = \prod WNL_{ij} X_i * pi\_bias \quad (7)$$

Where $NetL_{ij}$ is the net input to the GN for the linear part; WL_{ij}'s are the weights for the linear discriminant function; X_i's are the inputs, s_bias is the bias weight corresponding to the linear part; $NetNL_{ij}$ is the net input to the GN for the non-linear part; WNL_{ij}'s are the weights corresponding to non-linear discriminant function, and pi_bias is the bias weight corresponding to the non-linear discriminant input. The output from the linear and non-linear discriminant portions is calculated using the respective activation functions (∫ and Ω), which can be either a sigmoid, a Gaussian, a Spline, or any other mathematical function satisfying the conditions of being an activation function in the traditional ANNs employing MPANs. In this study, sigmoid and Gaussian activation functions were used to calculate the outputs from the linear and non-linear components of the GN model. The assimilation function was a linear combination of the two outputs. The equation to calculate the overall output from the GN model can be represented as follows:

$$O_{pk} = w y_1 + (1-w) y_2 \quad (8)$$

Where O_{pk} is the overall output from the GN model; w is the weight corresponding to the linear output y_1; and y_2 is the non-linear output from the GN model. The training of the GN model is carried out in a similar fashion as a traditional ANN using gradient descent method. The weight parameter w is also optimized during training so that the GN model consists of a total of 2n+3 weight parameters for n inputs. The overall structure of the GN model described above provides a very compact ANN model as compared to the traditional ANN model having many times more weights due to the number of hidden neurons involved in them. The details of training of a GN model are not included here and can be found in Chaturvedi et al. (1999) and Yadav et al. (2006).

IV. RESULTS AND DISCUSSIONS

The results in terms of various statistical parameters from the two ANN models are presented in Table 1. The values of E and R in excess of 0.95 from both 8-5-1 and GN models both during training and testing indicate an excellent performance. The 8-5-1 ANN model achieved a slightly better SSE both during training and testing; however, the GN model obtained a slightly better AARE value during testing. The performance of the two models was comparable in terms of all the threshold statistics. The comparable performance of the two models indicates that the GN model has tremendous potential in hydrology. Also, it provided a very compact ANN structure consisting of a single artificial neuron. Looking at the overall results from Table 1, the GN model may be preferred over the 8-5-1 ANN model due to its comparable performance and based on the principle of parsimony. The results in terms of scatter plots from the two models are provided in Figures 4 and 5, respectively. Figure 4 indicates the 8-5-1 ANN model's inability of estimating flows greater than 2000 m^3/s. This problem is overcome by the GN model (see Figure 5). Also, the GN model appears to estimate the lower magnitude flows (up to 1000 m^3/s) much better than those from the 8-5-1 ANN model.

Table 1: Error Statistics of ANN Models

Model	SSE	E	R	AARE	TSAD	TS1	TS1
TS50	TS100						
During Training							
8-5-1	0.33	0.969	0.985	3.98			
29.38	15.82	94.10	99.95	99.99			
GN	0.46	0.958	0.979	4.17			
32.63	13.54	92.28	99.96	99.98			
During Testing							
8-5-1	0.44	0.956	0.978	4.25	32.05	15.82	
	92.73	99.95	99.99				
GN	0.50	0.950	0.975	4.13			
32.47	16.16	91.50	99.96	100.0			

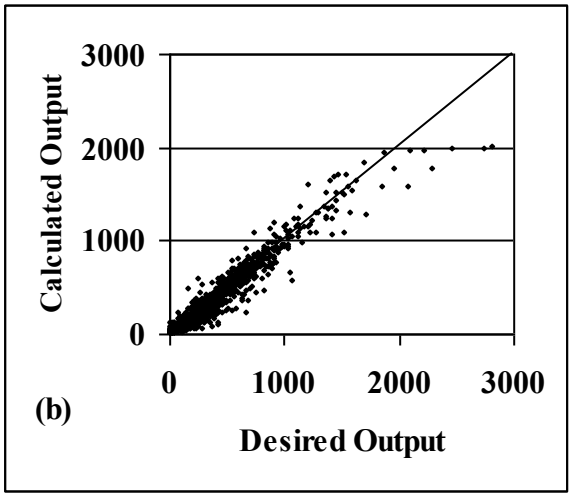

Figure 4: Scatter plot from 8-5-1 ANN model

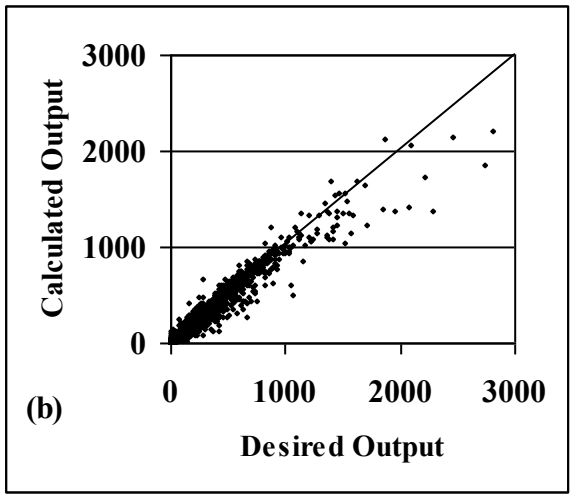

Figure 5: Scatter plot from GN model

V. CONCLUSION

In this study, the rainfall-runoff process has been modeled using two different ANN techniques: Three layered Feed forward Neural Network using MPAN and Generalized neural network using single generalized neuron.

This paper presents the findings of a study aimed at developing Generalized Neuron model for river flow forecasting. The results from the GN model are compared with a traditional feed-forward ANN trained with back-propagation with momentum factor. The daily rainfall and flow data for a 26-year period from Kentucky River, USA were employed. The performances of the two models were evaluated using five different types of error statistics capable of assessing ANN model performance comprehensively.

The results obtained in this study indicate that a compact ANN model consisting of a single artificial generalized neuron is capable of modeling the complex, dynamic, and non-linear rainfall-runoff process in a large catchment. The GN model was able to achieve similar performance as compared to a fully connected feed-forward ANN developed on the same data set. The GN model overcomes some of the problems associated with traditional ANNs developed using MPAN as a building block. It offers a flexible structure wherein various alternative discriminant, activation, and assimilation functions can be used to model the specific nature inherent in

different types of problems. The GN model has tremendous potential for solving a variety of problems in hydrology. It is hoped that future efforts will focus on the use of GN model in hydrology to exploit their strengths to advantage in hydrological modeling.

REFERENCES

[1] Abrahart, R.J. and See, L. (2000). Comparing neural network and autoregressive moving average techniques for the provision of continuous river flow forecasts in two contrasting catchments, Hydrol. Processes, 14, 2157-2172.

[2] Campolo, M., Andreussi, P., and Soldati, A. (1999). River flood forecasting with neural network model. Water Resour. Res., 35(4), 1191-1197.

[3] Chaturvedi, D.K., Malik, O.P., and Kalra, P.K. (2004), Generalised neuron-based adaptive power system stabilizer, IEEE Proc-Genetation, Transmission and Distribution, 151(2), 213-218.

[4] Chaturvedi, D.K., Satsangi, P.S., and Kalra, P.K. (1999). New neuron models for simulating rotating electrical machines and load forecasting problems, Elec. Power Sys. Res., 52, 123-131.

[5] Chen, J. and Adams, B.J. (2006). Integration of artificial neural networks with conceptual models in rainfall-runoff modeling, J. Hydrol., 318, 232-249.

[6] Jain, A. and Srinivasulu, S. (2004). Development of effective and efficient rainfall-runoff models using integration of deterministic, real-coded genetic algorithms, and artificial neural network techniques, Water Resour. Res., 40(4), W04302, doi:10.1029/2003WR002355.

[7] Jain, A. and Srinivasulu, S. (2006). Integrated approach to modelling decomposed flow hydrograph using artificial neural network and conceptual techniques, J. Hydrol., 317(3-4), 291-306.

[8] Jain, A., and Indurthy, S.K.V.P. (2003). Comparative analysis of event-based rainfall-runoff modeling techniques-deterministic, statistical, and artificial neural networks, J. Hydrol. Engg., ASCE, 8(2), 93-98.

[9] Minns, A.W. and Hall, M.J. (1996). Artificial neural networks as rainfall runoff models, Hydrol. Sci. J., 41(3), 399-417.

[10] Rumelhart, D.E., Hinton, G.E. and Williams, R. J. (1986). Learning representations by back-propagating errors, Nature, 323, 533-536.

[11] Shamseldin, A. Y., (1997). Application of a neural network technique to rainfall-runoff modeling, J. Hydrol., 199, 272-294.

[12] Yadav, R.N., Kalra, P.K., and John, J. (2006). Neural network learning with generalized mean-based neuron model, Soft Comput., 10, 257-263.