International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Advanced ANN Hydrological Models for Rainfall-Runoff Modeling

Seema Narain[1], Ashu Jain[2]

[1]Civil Academic Officer, College of Military Engineering, Pune, India,
[2]Professor, Civil Engineering, Indian Institute of Technology-Kanpur, Kanpur, India

ABSTRACT

Hydrological modeling is a tool for the investigation of hydrologic system for both the hydrologists and practicing water resources engineers involved in the planning, development, and management of water resources systems. The Artificial Neural Network (ANN) solutions have been found promising in modeling the complex hydrological systems as compared to the traditional conceptual or empirical approaches. The basic building block of such ANN models used in hydrology employs an artificial neuron called McCulloch and Pitts Artificial Neuron (MPAN) proposed by McCulloch and Pitts in the early 1940s. Recently, some researchers have proposed the use of Generalized Neuron (GN) models in other branches of engineering and sciences but such attempts have been limited in hydrology so far. Neural system (NS) models presented here include: (a) a traditional feed-forward multi-layer perceptron (MLP) ANN model (employing MPAN) trained using back-propagation algorithm, and (b) three different GN models. This paper presents the results of an investigation aimed at developing NS models for the purpose of rainfall-runoff modeling. The performance of the developed GN models is compared with a traditional feed-forward neural network model developed using MPANs. The rainfall and flow data derived from the Kentucky River Basin, USA were used for the model development and validation. With their compact structure, less number of parameters, and, lesser training time, the GN models were found more promising for the rainfall-runoff modeling in the present study. The results obtained in this study indicate that the GN models have tremendous potential for application in hydrological development. It is hoped that future research efforts will focus on exploiting the strengths of such artificial neuron models for an effective and efficient operation and management of water resources and environmental systems.

Keywords: Hydrological Modeling, Generalized Neuron, Advanced Neuron Models, Artificial Neural Network, Neural System in Hydrology

1. INTRODUCTION

Water is a natural resource that is essential to all kinds of lives on the earth. Out of the total water available on earth, only about one percent is available as fresh water on land, and the rest is contained either in the oceans, or in the form of frozen ice on mountain tops and glaciers (Subramanya, 1994). An important factor in the water resources planning, development, design operation, or management project is the accurate estimation of the available water at a local source such as a river. If the estimated runoff at a location in a river is inaccurate, it may lead to inaccurate policy being implemented resulting in not only the loss of revenue but also the loss of life and property in extreme cases. Runoff forecast models are useful in many water resources applications such as flood control, drought management, operation of water supply utilities, optimal reservoir operation, and design of various hydraulic structures such as dams, bridges, and culverts, etc. Runoff forecasts are normally made through the development of runoff forecast models that use only hydrologic data, or through rainfall-runoff models that use both hydrologic and climatic data. 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). Most of the ANN applications to hydrology employ the 'McCulloch and Pitts' Artificial Neuron (MPAN) that was proposed in the early 1940s. The MPAN has been found to function very well in most engineering applications; however, determination of an optimal ANN architecture has remained a trial and error procedure over the years. Such trial and error procedures are often inefficient in terms of computational effort and may not be able to ensure optimal solutions. There are no standard guidelines that can be employed uniformly for determining the architecture i.e. the number of hidden layers as well as the number of hidden neurons of the ANN models. Therefore, there appears to be a strong need to explore another artificial neuron structures to overcome such limitations. The increased demand on the more and more accurate estimations of future variables has forced the researchers to look beyond MPAN. Recently, some researchers have proposed the use of Generalized Neuron (GN) models in other branches of engineering and sciences but such attempts have been limited in hydrology. The GN differs from the traditional MPAN in many ways including its capability to have non-linear discriminant function. The major limitation of the MPAN has been the linear nature of the discriminant function employed in the aggregation of inputs. There is no need to determine the number of hidden layers and consequently the number of hidden neurons as a single generalized neuron is capable of modeling a complex physical system. 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. Some of the reasons for the ANNs not been adopted as operational tools are the reluctance shown by water resources policy makers, ignorance about their strength, ANNs being data-driven models and the perception that they are not able to explain the physical behavior of the catchment, etc. There are certain issues that need the attention of researchers in order to develop solutions that are capable of providing improved model performance in modeling and forecasting of the complex physical systems such as hydrologic and environmental systems. The motivation of the present thesis work mainly stems from these issues which are either unexplored or are partially explored.

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). The statistical properties of the training and testing input data set have been presented in Table 1, which show that both the data set represent the overall catchment characteristics well.

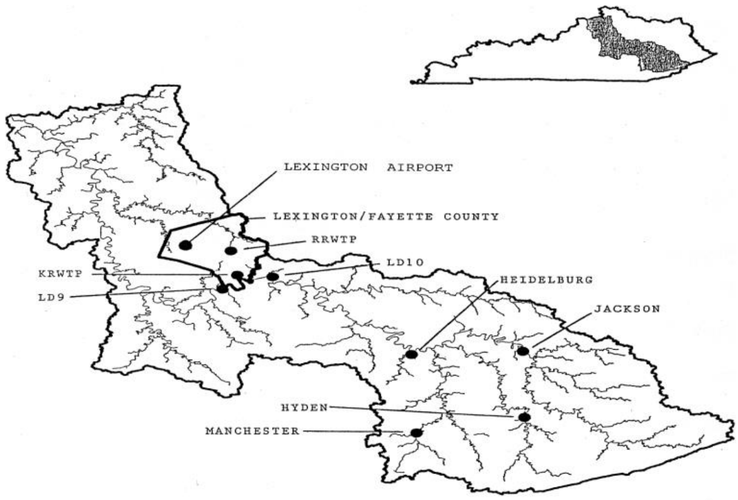


Figure 1: Kentucky River Basin

Table 1: Statistical properties of the Kentucky River data

	Flow (cubic meters/sec)		Rainfall (mm)	
	Training	**Testing**	**Training**	**Testing**
Minimum	3.60	3.28	0.000	0.000
Maximum	2528.69	2806.19	77.40	81.99
Average	149.94	144.78	3.24	3.16
Std. Deviation	243.30	232.81	6.20	6.18
Skewness	3.67	4.00	3.49	3.94

Model Performance: The performance of all the models developed in this study was evaluated using five different standard statistical measures. These are Normalized root mean square error (NRMSE), Nash-Sutcliffe efficiency (E), Pearson coefficient of correlation (R), average absolute relative error

International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

(AARE) and threshold statistic (TS). The equations to compute these statistics are provided below.

$$NRMSE = \frac{\left(\frac{1}{N}\sum_{t=1}^{N}\left(XE\ (t) - XO\ (t)\right)^2\right)^{1/2}}{\overline{XO}} \quad (1)$$

$$AARE = \frac{1}{N}\sum_{t=1}^{N}\left|\frac{XO(t) - XE(t)}{XO(t)}\right| X\ 100\% \quad (2)$$

$$E = 1 - \frac{\sum (XE - XO)^2}{\sum (XO - \overline{XO})^2} \quad (3)$$

$$R = \frac{\sum (XO - \overline{XO})(XE - \overline{XE})}{\sqrt{\sum (XO - \overline{XO})^2 \sum (XE - \overline{XE})^2}} \quad (4)$$

$$TS_x = \frac{n_x}{N} \times 100\% \quad (5)$$

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 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 25%, 50%, and, 100% were considered in this study to compute threshold statistics. Lower value of NRMSE would indicate better model performance and vice-versa; however the NRMSE, can still be biased towards high magnitude of the output variable as the numerator involves square of deviation. The AARE, which is relative with respect to individual output variable values, is better indicator as it will not be biased towards either high or low magnitude values. Values of 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 –∞ and +1.0 with values close to 1.0 being very good.

2. 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. The GN model consists of five components as opposed to two in the McCulloch and Pitts artificial neuron (MPAN). The typical structure of a GN model is shown in Figure 2. The five components of a GN model are (1) first discriminant function (f1), (2) second discriminant function (f2), (3) activation function (g1) corresponding to the first discriminant function, (4) activation function (g2) corresponding to the second discriminant function, and (5) an assimilation function (f3) that aggregates outputs from the components (3) and (4) above. The training of the GN model is carried out in a manner similar to the training of a traditional ANN using gradient descent method. A GN model receives inputs from an external source and gives output to an external receiver like in a conventional ANN model. As shown in Figure 2, a GN receives the inputs through its first two components and then computes the net input signal depending on the discriminant functions employed. A bias element is added to simulate the threshold characteristic of an artificial neuron.

The net input signal can be calculated as follows:

$$NetD1 = f_1\,(WD1_i, X_i, Bias\,b1) \qquad (6)$$

$$NetD2 = f_2\,(WD2_i, X_i, Bias\,b2) \qquad (7)$$

Where NetD1 and NetD2 are the net input signals to the GN model corresponding to the first and second discriminant functions, respectively; f_1 and f_2 are the first and second discriminant functions; $WD1_i$ and $WD2_i$ are the weights corresponding to the first and second discriminant functions, respectively, connecting to the inputs X_i's; i is an index representing the elements of the input vector; and Bias b1 and Bias b2 are the bias weights corresponding to the two components of the GN model.

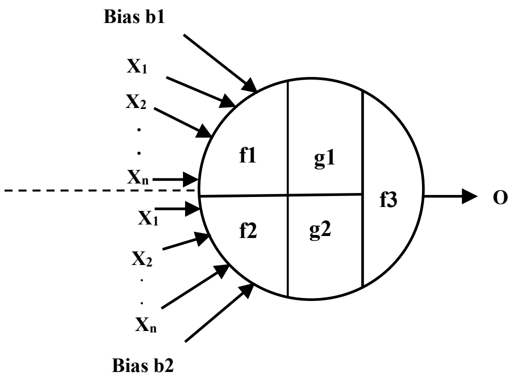


Figure 2: Generalized Neuron model

The outputs from the two components are calculated using the respective activation functions, which can be a Sigmoid, a Gaussian, a Spline, a linear function, or any other mathematical function satisfying the conditions of being an activation function in the traditional ANNs employing MPANs. The two outputs can be calculated as follows:

$$O1 = g_1\,(NetD1) \qquad (8)$$

$$O2 = g_2\,(NetD2) \qquad (9)$$

Where, g_1 and g_2 are the first and second activation functions associated with the first and second discriminant functions, respectively. The overall output from the GN model is then calculated using a linear aggregation of the two outputs calculated above. This can mathematically be represented as follows:

$$O = f_3\,(O1, O2) = W\,O1 + (1\text{-}W)\,O2 \qquad (10)$$

Where O is the overall output from the GN model; f_3 is the assimilation function that calculates output from the GN model; W is the weight corresponding to the output O1; and (1-W) is the weight corresponding to the output O2. The training of the GN model is carried out in a manner similar to the training of a traditional ANN using gradient descent method. It can be described by the following equations:

$$WD1_i(k+1) = WD1_i(k) + \Delta WD1_i(k+1) \qquad (11)$$

$$\Delta WD1_i(k+1) = \eta\,\delta_{D1}\,X_i + \alpha\,WD1_i(k) \qquad (12)$$

Where k and k +1 are the training steps; $WD1_i$ (k +1) is the updated weight connecting i^{th} input to the first discriminant function of the GN model at training step (k +1); $\Delta WD1_i$ (k +1) is the amount of weight adjustment to be made at the training step (k +1); η is the learning rate; δ_{D1} is the error signal corresponding to the first discriminant function; and α is the momentum factor. The adjustment of the weights corresponding to the second discriminant function ($\Delta WD2_i$ (k +1)) is carried out in an exactly similar manner. The method of calculating the error signals, δ_{D1} and δ_{D2}, depends upon the kind of discriminant and activation functions employed in the GN model. More details of the training of a GN model can be found in Chaturvedi et al. [2004]. The total number of weights to be optimized in a GN model is (2N+3) where N is the total number of inputs received by the GN model from an external source. The overall structure of the GN model described above provides a very compact ANN model as compared to the traditional MLP model having many times more weights due to the number of hidden neurons involved in them. The GN model described above has been employed in electrical systems but, in the knowledge of the authors, has not been employed in hydrology. Two different discriminant functions and two different activation functions were employed in this study after investigating many functions. The use linear and non-linear discriminant function was investigated for developing the GN models. Sigmoid and Gaussian functions were employed as activation functions.

3. MODEL DEVELOPMENT

Four types of neuron models were developed in the present study. The first model is MLP model with a traditional neuron structure, the other three models are Generalized Neuron models based on the concept of GN (discussed earlier).

MLP Models

The MLP model 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 3). The inputs to the ANN are average rainfall at various time steps (P_t, P_{t-1}, and P_{t-2}); and flow in Kentucky River at LD10 in the past ($Q10_{t-1}$, and $Q10_{t-2}$). The neuron models thus developed would require forecasts of one key inputs P_t, and the output from the neuron models is $Q10_t$ being modeled. 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. The architecture of MLP models in the form of 5-N-1 was investigated in which N represents the number of hidden neurons. The number of neurons in the hidden layer was varied from 1 to 15 to select the best possible architecture of the ANN model. For each value of hidden neurons, the BP algorithm was used to minimize the SSE at the output layer. Various standard error statistics during training data-set were plotted against the number of hidden neurons to determine optimal MLP architecture. Using this method, MLP architecture of 5-4-1 was found the best. Trial and error method was employed to find out the suitable learning rate coefficient and momentum correction factor to develop each of the models. The stopping criteria for the training of the models were kept as 50,000 as the maximum number of iterations or sum square error as 0.0005, whichever is achieved first. Once the MLP models were trained, they were used to calculate various error statistics on both training and testing data sets. The results in terms of various error statistics during training and testing are presented in Table 2.

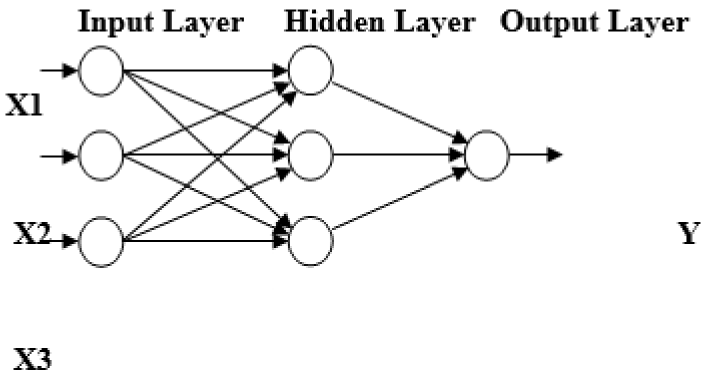


Figure 3: Structure of a feed-forward ANN

Generalized Neuron Models

Three types of generalized neuron (GN) models were developed to model flow at LD10 in this study. The three GN models consist of different combinations of aggregation and activation functions. Two different aggregation functions and two different activation functions were employed after investigating many functions in this study. The use of one linear and one non-linear aggregation function was investigated for developing GN models. Sigmoid and Gaussian functions were employed as activation functions. These are described in the following equations:

Linear Discriminant Function ($\sum$):

$$Net = \sum_{i=1}^{N} WD_i\, X_i + Bias \qquad (13)$$

Non-linear-Discriminant Function(Π):

$$Net = \prod_{i=1}^{N} WD_i\, X_i * Bias \qquad (14)$$

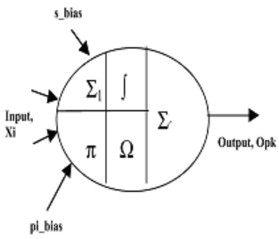
Sigmoid Activation Function (∫):

$$O = \frac{1}{1+e^{-Net}} \qquad (15)$$

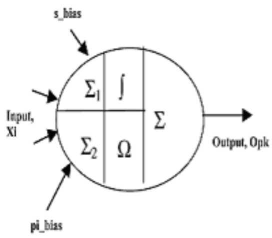
Gaussian Activation Function (Ω):

$$O = e^{-(Net)^2} \qquad (16)$$

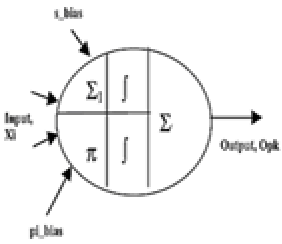
The three GN models are referred to as GNA, GNB and GNC models in this study. In the first GN model (GNA), linear aggregation function (Σ), and Sigmoid activation function (∫) are used in the first part; and Pie aggregation function (Π) and Gaussian activation function (Ω) are used in the second part (see Figure 4). The GNB model employed Σ and ∫ in the first part, and Σ and Ω in the second part. The GNC model employed Σ and Ω in the first part, and Π and Ω in the second part. The three GN models are shown in Figure 4. A gradient descent method similar to the back-propagation algorithm with momentum correction factor described earlier was employed for training of the three GN models. The inputs to GN models are same as those for the MLP models. The input and output data set were normalized to [0.1 0.9] for the development of GN models also. For each GN model, a suitable combination of training parameters (learning rate and momentum correction factor) was determined after a few trials. The stopping criteria for the training of the GN models were kept same as those for the MLP models.



(GNA Model)



(GNB Model)



(GNC Model)

Figure 4: Generalized Neuron models

4. RESULTS AND DISCUSSIONS

The results in terms of various error statistics from neural system models with five inputs are presented in Table 2(a) and Table 2(b). The best value of an error statistic from a model has been represented in bold font in the table. It may be noted that all the neural system models performed very well as shown by the E and R values in excess of 0.90 from all the models both during training and testing. Looking at the results from Table 2(a) and Table 2(b), it is clear that the GNB model performed the best in terms of most of the error statistics. It obtained the highest E and R values of 0.926 and 0.962 during training data set; and 0.918 and 0.959 during testing data set, respectively, which is very good. It also obtained the least AARE values of 23.13% and 22.84% during training and testing, respectively. It obtained a TS25 value of 50.7% during testing indicating that in almost half of the testing data points estimated, the ARE was less than 25%. The GNB model obtained the least NRMSE values both during training and testing. Comparing the performance of all the GN models, it may be said that the performance of GN models was comparable. Further, the performance of all the GN models was significantly better than the MLP model.

Table 2 (a): Statistical results from MLP and GN models

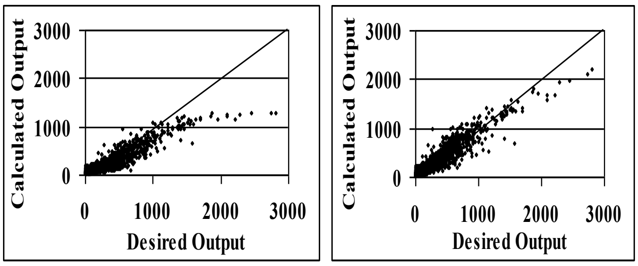
Model	NRMSE	E	R	AARE
During Training				
MLP	0.503	0.904	0.954	32.76
GNA	0.479	0.913	0.955	30.17
GNB	**0.441**	**0.926**	**0.962**	**23.13**

GNC	0.457	0.920	0.959	29.75
During Testing				
MLP	0.670	0.823	0.948	27.78
GNA	0.482	0.908	0.953	30.53
GNB	**0.454**	**0.918**	**0.959**	**22.84**
GNC	0.462	0.916	0.957	29.82

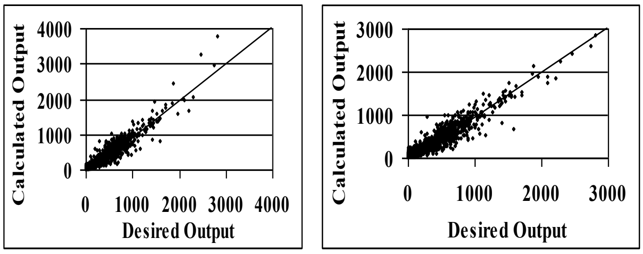
Table 2 (b): Statistical results from MLP and GN models

Model	TS25	TS50	TS100	Execution Time (CPU time)
During Training				
MLP	38.5	52.5	64.7	00:16:55.21
GNA	42.4	60.6	75.5	00:01:57.55
GNB	**49.1**	**69.3**	**84.4**	00:01:58.70
GNC	46.1	67.6	82.4	**00:01:52.68**
During Testing				
MLP	36.4	63.9	74.7	-
GNA	43.9	62.7	75.5	**-**
GNB	**50.7**	**69.4**	**82.8**	-
GNC	46.9	67.5	81.3	

Finally, looking at the execution times from Table 2(b), it can be noted that all the GN models take significantly less time for training as compared to the MLP model, which highlights their superiority over the traditional feed-forward neural networks. Observed and estimated flows from the four neural system models with five inputs are shown in the form of scatter plots in Figure 5 for testing data set. It is noted from Figure 5(a) that the MLP model was not able to capture the complex relationship between inputs and output data very well especially for high magnitude flows. The narrow scatter around the ideal line from the GNB model (see Figure 5(c)) further strengthens the conclusions about its superiority.



(a) MLP **(b) GNA**



(c) GNB **(d) GNC**

Figure 5: Scatter plot from the neuron models during testing

The GNC model estimated flows during testing exceedingly well (see Figure 5 (c)) and the points are narrowly scattered around the ideal line. This indicated that the combination of a non-linear aggregation function and sigmoid activation function in a GN model may be suitable for estimating the high magnitude flows.

5. SUMMARY AND CONCLUSIONS

Two different structures of the Neuron have been used in the present study, first is MPAN and the other is Generalized neuron. Three layered Feed forward Neural Network using MPAN and Generalized neural network using single generalized neuron have been used to develop different models in the present work. 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 models were evaluated using five different types of error statistics capable of assessing ANN model performance comprehensively. Considering the statistical and graphical results together, it may be concluded that the GN models offer an exciting alternative for modeling of the complex hydrological systems as they consist of a compact structure, are easy to develop, take less time to train, and exhibit better generalization ability than the traditional MLP type of ANN models. Further, out of the three GN models, GNB model was found to be the best while the GNC model was suitable for capturing timing and magnitude of peak flows. 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 results obtained in this study indicate that a compact neuron 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 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. Subramanya, K. (1994), Engineering Hydrology, Tata McGraw Hill Publishing Company Ltd., New Delhi.
13. Yadav, R.N., Kalra, P.K., and John, J. (2006). Neural network learning with generalized mean based neuron model, Soft Comput., 10, 257-263.