International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2024: 7.101

Sensitivity Analysis of ANN and Generalized Neuron Models for Hydrological Modeling

Seema Narain[1], Ashu Jain[2]

[1]Civil Academic Officer, Department of Civil Engineering, College of Military Engineering, Pune,411001, India
Email: seemanv[at]gmail.com

[2]Professor, Department of Civil Engineering, Indian Institute of Technology Kanpur, Kanpur-208 016, India
Email: *ashujain[at]iitk.ac.in*

Abstract: ***Artificial Neural Networks (ANNs) have gained significant attention for hydrological modeling due to their ability to handle complex, nonlinear relationships. Traditionally, the Multi-Layer Perceptron (MLP) model has been the most commonly employed ANN structure, although it lacks systematic guidelines for architecture selection and often requires extensive trial and error. To address these limitations, this study explores the application of a Generalized Neuron (GN) model, which offers a simplified architecture and improved flexibility. Three distinct GN configurations were tested using daily rainfall and streamflow data from the Kentucky River basin. The models were assessed for performance under varying initial weights and limited training data. Results indicate that GN models demonstrate higher resilience to initialization, improved generalization, and competitive accuracy with fewer parameters compared to MLPs. This suggests GN models are a promising alternative for efficient rainfall-runoff modeling.***

Keywords: Artificial neural networks, hydrologic models, generalized neurons, water resources, rainfall runoff process, hydrology.

1. Introduction

Modeling the rainfall-runoff (RR) process is fundamental for effective water resources planning, design, and operational management. The RR process is inherently complex, nonlinear, and dynamic, making it challenging to model with conventional deterministic or conceptual techniques. Traditional models, based on physical laws such as mass andmomentum conservation, often require extensive calibration and may struggle to capture intricate real-world behaviors, especially under data-limited conditions. In recent decades, data-driven modeling approaches, particularly Artificial Neural Networks (ANNs), have emerged as powerful tools for RR modeling. These models are not constrained by assumptions about the underlying physical processes and have shown promising performance in diverse hydrological contexts. Numerous studies havehighlighted the capability of ANNs to outperform classical in capturing nonlinear relationships between rainfall and runoff. Among the various ANN architectures, feed-forward Multi-Layer Perceptrons (MLPs), typically trained using backpropagation algorithms, have been extensively utilized. MLPs consist of input, hidden, and output layers, with neurons connected through weighted pathways. However, MLPs suffer from certain drawbacks: they often require a large number of hidden neurons, involve laborious trial-and-error procedures to determine optimal architecture, and are sensitive to the choice of initial weights. Additionally, the linear summation approach used in traditional neurons may limit their ability to model highly nonlinear dynamics. To overcome these limitations, recent research has introduced the Generalized Neuron (GN) model, which enhances flexibility by incorporating multiple discriminant and activation functions within a single neuron structure. Unlike MLPs, GN models do not require multiple hidden layers and can efficiently capture complex system behaviors with fewer parameters. This study aims to investigate the effectiveness of GN models in simulating the RR process and compares their performance with traditional MLP models. Specifically, we assess (a) the sensitivity of both models to different initial weight configurations, and (b) their robustness when trained on progressively reduced datasets.

2. Generalized neuron model

The Generalized Neuron (GN) model, introduced by Chaturvedi et al. [2004], represents a significant advancement over traditional artificial neuron models such as the McCulloch-Pitts Artificial Neuron (MPAN). Unlike the MPAN—which relies on a single linear summation function followed by a nonlinear activation function—the GN model incorporates a more flexible structure comprising five functional components. Figure 1 shows the structure of a GN model. The five components of a GN model are (a) first discriminant function (f1), (b) second discriminant function (f2), (c) activation function (g1) corresponding to the first discriminant function, (d) activation function (g2) corresponding to the second discriminant function, and (e) an assimilation function (f3) that aggregates outputs from the components (c) and (d) above.

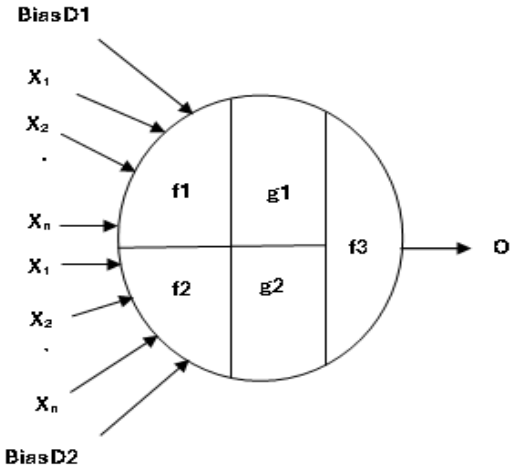


Figure 1: A Generalized Neuron

The GN model derives its power from the flexibility in being able to select different discriminant and activation functions. Since a single artificial generalized neuron is capable of capturing the complexity and non-linearity in the physical system being modeled, it is not necessary to have several hidden layers and corresponding hidden neurons. This reduces the complexity and dimensionality of the overall ANN model in a GN model.

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:

$$NetD1 = f_1(WD1_i, X_i, BiasD1) \quad (1)$$

$$NetD2 = f_2(WD2_i, X_i, BiasD2) \quad (2)$$

Where *NetD*1 and *NetD*2 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 *BiasD*1 and *BiasD*2 are the bias weights corresponding to the two components of the GN model. The outputs 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:

$$O1 = g_1(NetD1) \quad (3)$$

$$O2 = g_2(NetD2) \quad (4)$$

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:

$$O = f_3(O1, O2) = W\,O1 + (1 - W)\,O2 \quad (5)$$

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 $(1\text{-}W)$ 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. 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.

3. Study Area and Data

The study utilizes hydrological data from the Kentucky River basin in the United States. This river system serves as the primary water source for several municipalities in the region. The analysis focuses on data collected at Lock and Dam 10 (LD10) near Winchester, Kentucky, which drains a watershed area of approximately 10,240 km².The data employed include average daily stream flow (m^3/s) from Kentucky River at LD10 and daily total rainfall (mm) from five rain gauges, Manchester, Hyden, Jackson, Heidelberg, and Lexington Airport. The daily total rainfalls from the five rain gauges were spatially aggregated using simple mean approach. The rainfall and flow data of 26 years were available, which were divided into two sets: a training data set of thirteen years (1960-1972), and a testing data set of thirteen years (1977-1989). The performance of 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 (AARE), and threshold statistics (TS). In this study, TS statistics at ARE levels of 25%, 50%, and 100% have been considered. All of these are commonly employed error statistics to evaluate ANN model performance and their detailed description can be found in *Jain et al.* [2001] and *Jain and Kumar* [2009].

4. Model development

This section outlines the development and implementation of two categories of neural network models: a conventional Multi-Layer Perceptron (MLP) and three variants of the Generalized Neuron (GN) model. The models were trained and tested using rainfall and data from the Kentucky River basin. Details of the study area, data preprocessing, and model structures are presented below.

MLP Model Development

A feed-forward MLP architecture trained with the backpropagation algorithm and momentum factor was developed. The model comprises three layers: an input layer, one hidden layer, and an output layer. Both the hidden and output neurons used the sigmoid activation function. A computer program written in C was developed for ANN model simulation. The output from the MLP model was flow at time t, Q(t). Inputs to the MLP model were determined using auto- and cross-correlation analyses. The significant inputs were determined to be daily rainfalls P(t), P(t-1), and P(t-2) and daily average observed flows Q(t-1) and Q(t-2). Thus, an MLP structure of 5-N-1was explored. The input and output data were normalized in the range of 0.1 and 0.9. The optimum ANN structure is normally determined using a trial and error procedure. The number of hidden neurons was varied from 1 to 20 and the architecture giving the best performance in terms of E and AARE was selected as the optimum MLP model. An architecture of 5-4-1 was found suitable.

GN Model Development

Three different GN models were constructed, each employing a unique combination of discriminant and activation functions. The goal was to evaluate the influence

of these components on model accuracy and generalization. Sigmoid and Gaussian functions were employed as activation functions. These are described in the following equations:

Linear Discriminant Function (∑):

$$Net = \sum_{i=1}^{N} WD_i X_i + Bias \quad (6)$$

Non-linear Discriminant Function (Π):

$$Net = \prod_{i=1}^{N} WD_i X_i * Bias \quad (7)$$

Sigmoid Activation Function (∫):

$$O = \frac{1}{1+e^{-Net}} \quad (8)$$

Gaussian Activation Function (Ω):

$$O = e^{-(Net)^2} \quad (9)$$

Three different GN models are investigated in this study, which differ in the discriminant and activation functions employed. The GN models are referred to as GNA, GNB, and GNC models in this study. The details of the discriminant and activation functions employed in the three GN models are presented in Table 1. All the three GN models use a linear assimilation function as the fifth component as described earlier. A gradient descent method similar to back-propagation algorithm with momentum factor described earlier was employed for training of the three GN models. The stopping criteria were kept same as those for the MLP models.

Table 1: Details of the models developed

Model	*f*1	*f*2	*g*1	*g*2	No. of Parameters
MLP	Linear (∑)	---	Sigmoid	---	29
GNA	Linear (∑)	Non-linear (Π)	Sigmoid	Gaussian	13
GNB	Linear (∑)	Linear (∑)	Sigmoid	Gaussian	13
GNC	Linear (∑)	Non-linear (Π)	Sigmoid	Sigmoid	13

5. Sensitivity Analyses

To assess the robustness and reliability of the developed models, two separate sensitivity analyses were conducted. These focused on (a) the effect of different initial weight configurations, and (b) the impact of progressively reducing the training dataset size. Both analyses were carried out on all four models: MLP, GNA, GNB, and GNC.

Sensitivity to Initial Solutions

The feed-forward multi-layer perceptron ANN models are often criticized for their inability to provide global solution in terms of optimized weight matrix. This is due to several reasons including the gradient descent nature of the back-propagation training algorithm employed in MLP model building, the error surfaces being extremely complex consisting of many local solutions, and heavy dependence of the MLP models on the initial weights among others. One can use a higher order training algorithm for training of an MLP model, which involves computation of higher order derivatives at each training cycle increasing computational burden during training. The complex structure of the MLP models consisting of many hidden layers and associated hidden neurons leads to the error surfaces being complex consisting of many local peaks and troughs. Therefore, the principle of parsimony should be exercised during the MLP model building process. However, the number of hidden neurons at-least equal to the size of the input vector is normally required, which still causes the problem of local minima and the search algorithm getting stuck in one of them. Given that the BPA is the most popular training method employed and that there will be many local minima for the training algorithm to handle, it is a usual practice to shake the initial weights and retrain the MLP models. Often, one needs to attempt ten or twenty different initial weight sets in order to get close to the global optimum, which is still not guaranteed.

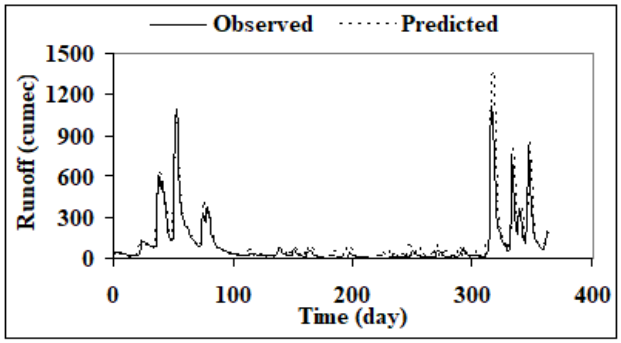
With these problems in mind, a sensitivity analysis was carried out to investigate the dependence of the models developed in this study on the initial solutions. For this, each model developed (MLP, GNA, GNB, and GNC) was trained using ten different initial weight vectors. Various performance evaluation measures were then calculated from all the models corresponding to each initial weight vector. The results in terms of average and standard deviation of various error statistics (over ten different models developed on ten different initial weights) are presented in Table 2. Looking at the average of the statistics from Table 2, it is clear that the GNB model performs the best in terms of all the error statistics. Its performance was marginally better than the next best model in terms of NRMSE, E, and R but was significantly better in terms of AARE and TS statistics. The performance of the GNC model was next to the best model. Further, the MLP model performed slightly better than the GNA and GNC models in terms of average NRMSE, E, and R statistics but its performance was much inferior in terms of the AARE and TS statistics.

Analyzing at the results from Table 2 in terms of the standard deviations of the various error statistics during training and testing, it was observed that MLP model consistently obtained higher standard deviations for all the error statistics. Higher standard deviations indicate higher sensitivity of a model towards the initial weights. This may be because the search algorithm probably getting stuck in different local minima corresponding different initial weights and providing an inferior overall performance. The GNA model performed the best in terms of standard deviations of various statistics as it obtained the least standard deviations for all the error statistics. The GNB and GNC models were also able to perform at par with the GNA model in terms of some of the error statistics (see bold font statistics in Table 2). This means that the GN model employing a combination of linear and non-linear discriminant functions and a combination of Sigmoid and Gaussian functions as activation functions was the least sensitive to initial weights. This is a significant finding as such a combination of discriminant and activation functions can be employed for the modeling of complex physical systems without having to worry about getting stuck in local minima and/or shaking initial weights.

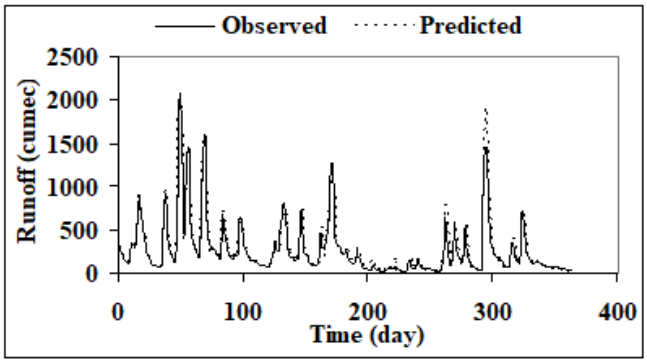
Table 2: Sensitivity analyses results with respect to initial weights

Model	NRMSE	E	R	AARE	TS25	TS50	TS100
Average During Training							
MLP	0.497	0.905	0.952	28.5	43.6	64.0	75.5
GNA	0.480	0.913	0.956	29.9	42.1	60.4	75.3
GNB	**0.421**	**0.933**	**0.966**	**23.3**	**53.0**	**72.7**	**86.6**
GNC	0.458	0.921	0.960	29.8	48.1	70.9	84.2
Average During Testing							
MLP	0.515	0.895	0.946	28.3	45.8	64.6	75.4
GNA	0.482	0.909	0.953	30.2	43.9	62.6	75.3
GNB	**0.453**	**0.920**	**0.960**	**23.0**	**54.1**	**72.4**	**85.6**
GNC	0.467	0.915	0.957	29.8	48.7	70.3	83.5
Standard Deviation During Training							
MLP	0.066	0.025	0.013	6.016	10.407	9.551	10.876
GNA	**0.001**	**0.000**	**0.000**	**0.058**	**0.123**	**0.045**	**0.070**
GNB	0.002	0.001	**0.000**	0.128	0.605	0.740	0.489
GNC	0.001	0.001	0.001	0.295	1.490	2.237	0.744
Standard Deviation During Testing							
MLP	0.056	0.022	0.012	6.548	9.076	7.796	9.449
GNA	**0.000**	**0.001**	**0.000**	**0.058**	**0.069**	**0.087**	**0.097**
GNB	0.001	**0.000**	**0.000**	0.132	0.438	0.692	0.649
GNC	0.005	0.002	0.001	0.345	1.610	1.906	1.378

Further, the difference in the performances from MLP and GN models in terms of standard deviations on AARE and TS statistics was quite significant indicating that different solutions obtained from MLP model differed significantly from each other as compared to the GN models. This demonstrates that the different solutions obtained from the GN models corresponding to different initial weights are very close to each other indicated by almost zero standard deviations for many of the error statistics. Considering the sensitivity analysis results in terms of both average and standard deviations taken together, the GNB model appeared to perform the best. The time series plots from the GNB model from the best initial weights for two sample years during testing (one wet and one dry year) are shown in Figure 2. Figure 2 demonstrates that the GNB model is able to estimate the magnitude and timing of all the peak flows very well, and is able to estimate low flows also very well.



(a) Time-series plot from the GNB model for sample dry year 1986



(b) Time-series plot from the GNB model for sample wet year 1989

Figure 2: Time series plots from the GNB model with best initial weights

Sensitivity to Reduced Training Data

The performance of the four models was evaluated when presented with gradually reduced training data with an objective of investigating their efficiency under scarce data conditions. As discussed earlier, the models were trained with a set of ten different initial weight in order to find the sensitivity of the models towards initial weights. The best model structure was selected on the basis of performance of the model during training as well as testing with ten different initial weights. The initial weight with which the performance of the models was found the best was considered as the best model structure. Table 3 presents the statistical results of the models with best model structure. In order to carry out the computational experiment, the four models with best model structure were retrained using reduced number of years of training data. The training data presented to the best model structures with best (MLP, GNA, GNB, and GNC) were reduced from thirteen to one year in a step of one year. After retraining of the models, performance statistics were calculated during training and testing data sets. The training data set consisted of reduced years but the testing data set was kept constant at thirteen years in order to test the model performance under scarce training data. The statistical results are not presented here due to their volume. Instead, the results for R, AARE, NRMSE, and TS50 are presented in Figure 3 and Figure 4 for training and testing, respectively.

Table 3: Statistical results from MLP and GN models (with best initial weights)

Model	NRMSE	E	R	AARE	TS25	TS50	TS100
During Training							
MLP	0.439	0.927	0.963	21.94	53.58	**74.28**	**87.51**
GNA	0.479	0.913	0.956	29.76	42.40	60.40	75.44
GNB	**0.423**	**0.933**	**0.966**	**23.43**	**53.79**	73.50	86.94
GNC	0.455	0.922	0.960	29.21	50.04	73.50	85.11
During Testing							
MLP	0.460	0.917	0.958	20.93	55.48	72.79	85.89
GNA	0.482	0.909	0.953	30.11	43.91	62.76	75.57
GNB	0.454	0.919	0.960	23.09	54.68	73.02	86.27
GNC	0.476	0.911	0.955	29.13	50.91	72.68	85.30

Looking at Figure 3 and Figure 4, it can be noted that the performance of all the models is unaffected when the number of training years is reduced from thirteen to about three or four years. Figure 3(a) and Figure 4(a) show that correlation coefficient is largely unaffected by the number of

training years. It appears that the performance of models GNA and GNC gets affected significantly beyond three training years and the performance of MLP and GNB models remains largely unaffected even when training years were reduced to a single year. This shows that both MLP and GNB models are robust when encountered with scarce data situation, which is a little surprising result as both of these models employed linear discriminant functions. Analyzing the differences in performances of the MLP and GNB models using TS50 (Figures 3(d) and 4(d)), it is noted that the performance of the GNB model is far superior to that of the MLP model up to six training years beyond which the performance of the two models is comparable with the MLP model performing marginally better. This trend was observed in other error statistics also although not as apparent as that for TS50.

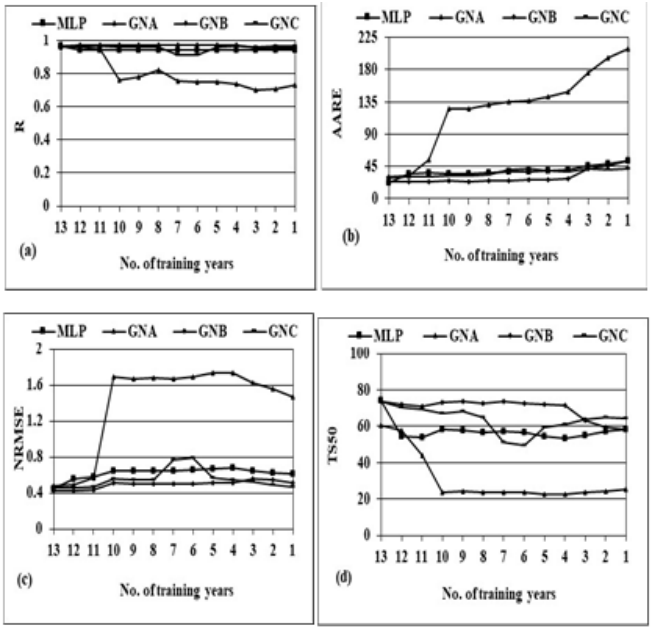


Figure 3: Error statistics during training under reduced training data

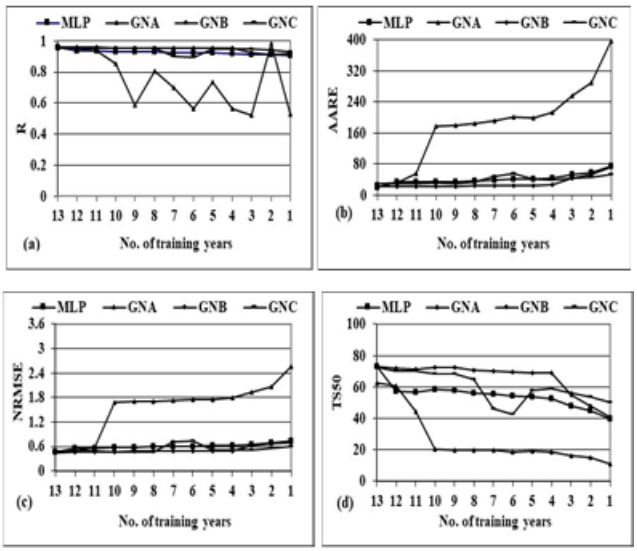


Figure 4: Error statistics during testing under reduced training data

6. Summary and Conclusions

This study investigated the application of Generalized Neuron (GN) models for simulating the nonlinear and complex rainfall-runoff (RR) relationship, and compared their performance against the conventional Multi-Layer Perceptron (MLP) model. Three GN variants—GNA, GNB, and GNC—were evaluated using daily rainfall and streamflow data from the Kentucky River basin. Five different error statistics were used to evaluate the model performance. Experiments were carried out to investigate the efficiency of the models in terms of their dependence on the initial weights and reduced training data.

It has been found that the GN models perform better than the conventional MLP model. The GN models were found to be insensitive to the initial weights. The GN models offer a promising alternative to model the complex and non-linear rainfall-runoff process as they have a very compact structure, take less time to train, consist of a very few parameters, are independent of the initial weights, and provide better generalization and extrapolation ability beyond the range of training data. The GNB and MLP models were found to perform well when presented with minimum amount of training data with the GNB model performing better when presented with seven year or more of training data. The GN models have tremendous potential to be employed in modeling of the complex engineering systems. They require significantly less time for training.

There exists a need and tremendous potential for carrying out research in the area of developing new artificial neuron models and it is recommended that the GN models proposed in this study are applied in other watersheds to have more confidence in the findings reported here. The method of training employed in this study was popular back-propagation training algorithm, which has been reported to have its own limitations. The new GN models trained using alternative training algorithms may prove to be still better tools for hydrological modeling; however, such aspects need attention by the water resources researchers.

References

[1] Campolo, M., Andreussi, P., and Soldati, A. (1999) River flood forecasting with neural network model, *Water Resour. Res.,* 35(4), 1191-1197.
[2] Chaturvedi, D.K., Malik, O.P., and Kalra, P.K. (2004), Experimental studies with a Generalized Neuron based power system stabilizer, *IEEE Trans. Power Systems,* 19(3), 1445–1453.
[3] Curry, B. and Morgan, P. (1997), Neural network: A need for caution, *Omega*, *Intl. J. Mgmt. Sci*., 25(1), 123-133.
[4] Dawson, D.W. and Wilby, R. (1998), An artificial neural network approach to rainfall-runoff modeling, *Hydrol. Sci. J.*, 43 (1), 47-65.
[5] Hsu, K.-L., Gupta, H.V. and Sorooshian, S. (1995), Artificial neural network modeling of the rainfall-runoff process. *Water Resour. Res*., 31(10), 2517-2530.
[6] Jain, A., and Indurthy, S.K.V.P. (2003), Comparative analysis of event based rainfall-runoff modeling

techniques-deterministic, statistical, and artificial neural networks, *ASCE J. Hydrol. Engg.*, 8(2), 93-98.

[7] Jain, A. and Kumar, S. (2009), Dissection of trained neural network hydrologic model architectures for knowledge extraction, *Water Resour. Res*.

[8] Jain, A. and Narain S. (2014), "Modeling rainfall magnitude using neural system models, Intl. J. Wat. Resour. & Env. Mgmt., 5(1-2), 117-126.

[9] 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.

[10] 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.

[11] Jain, A., Varshney, A.K., and Joshi, U.C. (2001), Short-term water demand forecast modeling at IIT Kanpur using artificial neural networks, *Water Resour. Mgmt.*, 15(5), 299-321.

[12] McCulloch, W. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity, *Bulletin of Mathematical Biophysics*, 7, 115-133.

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

[14] Narain S**.** and Jain, A. (2011), "Development of advanced neuron models for rainfall-runoff modeling", International Journal of Earth Sciences and Engineering, 4 (6): 216-222.

[15] Narain, S. and Jain, A. (2006), Hydrological modeling using generalized artificial neuron model, *Proc. ASCE-EWRI Intl. Conf. - An International Perspective on Environmental and Water Resources*, 18-20 December, 2006, New Delhi, India.

[16] Narain, S. and Jain, A. (2005), "Evaluation of the sensitivity of learning rate on the training of neural network hydrologic models", Session H35 – Soft computing tools for hydrologic modeling - Proc. AGU Fall Meeting 2005, December 5-9, 2005, San Francisco, California, USA.

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

[18] Sajikumar, N. and Thandaveswara, B.S. (1999), A non-linear rainfall-runoff model using an artificial neural network, *J. Hydrol*., 216, 32-55.

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

[20] Smith, J. and Eli, R.N. (1995), Neural network models of the rainfall runoff process, *ASCE J. Water Resour. Plng. Mgmt*., 121, 499-508.

[21] Srinivasulu, S. and Jain, A. (2009), River flow prediction using an integrated approach, *ASCE J. Hydrol. Engg.*, 14(1), 75-83.

[22] Tokar, A. S. and Markus, M. (2000), Precipitation runoff modeling using artificial neural network and conceptual models, *ASCE J. Hydrol. Engg.,* 5(2), 156-161.

[23] Wilby, R. L., Abrahart, R. J., and Dawson, C. W. (2003), Detection of conceptual model rainfall-runoff processes inside an artificial neural network, *Hydrol. Sci. J*., 48(2), 163–181.

[24] Zhang, B. and Govindaraju, S. (2000), Prediction of watershed runoff using Bayesian concepts and modular neural networks, *Water Resour. Res.*, 36 (3), 753-762.