Journal of Applied Sciences

ISSN 1812-5654

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Applied Sciences 8 (17): 2949-2957, 2008
ISSN 1812-5654

Suitability of Artificial Neural Network in Daily Flow Forecasting

[1]Karim Solaimani and [2]Zahra Darvari
[1]GIS and RS Centre, University of Agricultural and Natural Resources Sciences, Sari, Iran
[2]Department of Watershed Management,
University of Agricultural and Natural Resources Sciences, Sari, Iran

Abstract: This study aims to development of the Kasilian indicator river flow forecasting system using Artificial Neural Network (ANN). In this study the performance of multi-layer perceptrons or MLPs, the most frequently used artificial neural network algorithm in the water resources literature, in daily flow estimation and forecasting was investigated. Kasilian watershed in Northern Iran, representing a continuous rain-fall with a predictable stream flow events. Division of yearly data into four seasons and development of separate networks accordingly was found to be more useful than a single network applicable for the entire year. The used data in ANN was hydrometric and climatic daily data with 10 years duration from 1991 to 2000. For the mentioned model 8 years data were used for its development but for the validation/testing of the model 2 years data was applied. Based on the results, the L-M algorithm is more efficient than the CG algorithm, so it is used to train 6 ANNs models for rain fall-runoff prediction at time step t+1 from time step t input. The used network in this study was MLP with BP (back propagation) algorithm.

Key words: Neural networks, daily river flows, L-M and CG algorithm

INTRODUCTION

Real time forecasting of surface runoff is needed in planning and operation of water resources. Two basic approaches exist for this purpose, namely, conceptual or physical modeling and system-theoretic or black box modeling. Conceptual models attempt to model the flow physics and are generally non-linear, time-invariant and deterministic, involving parameters that represent watershed characteristics. Black box models on the other hand do not investigate the physical process and are of input-output type. One of the black box techniques, which has caught attention of hydrologists in recent past is Artificial Neural Networks (ANN), or shortly, Neural Networks (NN). NN models, which follow the cognition process of a human brain, are useful and efficient in problems for which the characteristics of the processes are difficult to describe using deterministic or stochastic equations. Neural networks neither require a priori detailed understanding of catchment's physical characteristics nor extensive data pre-processing and can handle incomplete, noisy and ambiguous data (Maier and Dandy, 2000). Many times they are computationally simpler than physically based schemes. They are also well suited to dynamic problems and are parsimonious in terms of information storage within the trained model. One of the initial applications of ANN in hydrology to forecast 1 h ahead rainfall intensity in space and time domain was due to French *et al.* (1992). A series of applications have been reported since then and they range from previous (Halff *et al.*, 1993; Garrote and Bras, 1995) to the recent study (Kisi, 2004; Olsson *et al.*, 2004). A good review of concepts and applications related to ANN to water resources can be seen in. ANN can model the highly nonlinear phenomenon in a simple way. Two different types of ANN are frequently used to model the rainfall-runoff process. One is the static networks including the Multilayer Perceptron (MLP) and the Radial Basis Function network (RBF) (Lin and Chen, 2004; Rajurkara *et al.*, 2004). The other one is the dynamic network such as the real-time recurrent network (Chang *et al.*, 2002, 2004; Chiang *et al.*, 2004). In the rainfall modeling context used synthetically generated rainfall storms to calibrate an ANN model and generated plausible rainfall events. Luk *et al.* (2001) identified an optimal set of spatio-temporal inputs for an ANN rainfall forecasting model using 15-min-intervals rainfall records. Recently, Abebe *et al.* (2000) compared fuzzy rule-based and ANN models for reconstructing missing precipitation events. In the last 15 years, ANNs have shown promising potential in various hydrologic applications (e.g., ASCE Task Committee, 2000a, b; Maier and Dandy, 2000, for a review). More recent hydrologic applications of ANNs are summarized in Cigizoglu (2005), Cigizoglu and Kisi

Corresponding Author: Karim Solaimani, GIS and RS Centre, University of Agricultural and Natural Resources Sciences, Sari, Iran Tel: (+98) 9111521858

(2006), Coulibaly *et al.* (2005) and Dibike and Coulibaly (2006). By considering complexity of the phenomena involved there is a strong need to explore alternative solutions through modeling direct relationship between the input and output data without having the complete physical understanding of the system. While data-driven models do not provide any information on the physics of the hydrologic processes, they are very useful for river flow forecasting where the main concern is accurate predictions of runoff (Nayak *et al*., 2005). Due to the ability of ANNs in modeling complex nonlinear systems, successful applications of these method in water resources modeling have been widely reported, such as for rainfall-runoff simulation (Sudheer *et al*., 2002; Chang *et al*., 2002), river flow forecasting (Imrie *et al*., 2000; Hu *et al*., 2001; Kumar *et al*., 2004) and rainfall forecasting (Luk *et al*., 2001; Rami'rez *et al*., 2005). And Comparison of neural nor infilling missing daily weather records (Coulibaly and Evora, 2007). The study focused on Comparative analysis of training methods and different data's for rainfall-runoff predication with use ANNs. In this respect, computational efficiencies measured in terms of better achieved accuracy and convergence speed of the Feed Forward Back Propagation (FFBP), Radial Basis Function (RBF) and Generalized Regression Neural Network (GRNN) in network training are evaluated and compared. A watershed system with continuous regime of stream flow in Northern Iran is used as a case study. In the presented study, the Levenberg-Marquardt and Conjugate gradient technique is employed. Levenberg-Marquardt optimization technique is more powerful than the conventional Conjugate gradient techniques (El-Bakyr, 2003; Cigizoglu and Kisi, 2006).

MATERIALS AND METHODS

Study site description: The aim of this study is to briefly describe the study area and the structure of the utilized ANN. The proposed methodologies were applied to the upper sub-watershed of the Kasilian watershed system (Fig. 1) for the evaluation of the predication daily flow forecasting Modeling capabilities like FFBP in terms of their abilities in network training. The influences of daily rainfall, stream flow and air temperature data as different input dimensions were also evaluated. The study area of Kasilian watershed with a drainage area of 3360 km^2 is located in Mazandaran province, northern part of Iran, with an important role for the agricultural and industrial activities.

The sources of water are mostly originated from snowmelt, springs and also individual rain events. In spite of a permanent flow system, the rainy season is limited to 9 months duration (October-April) with a mean annual precipitation of near to 800 mm. Rainfall events are usually occurred during the last days but rainfall durations of 1 day or less is also recorded. Daily optimized precipitation occurs in August to September and daily minimum precipitation recorded in February to March. This watershed is included two sub-basin of Talarsarband and Tilehsiah which are following to the North direction.

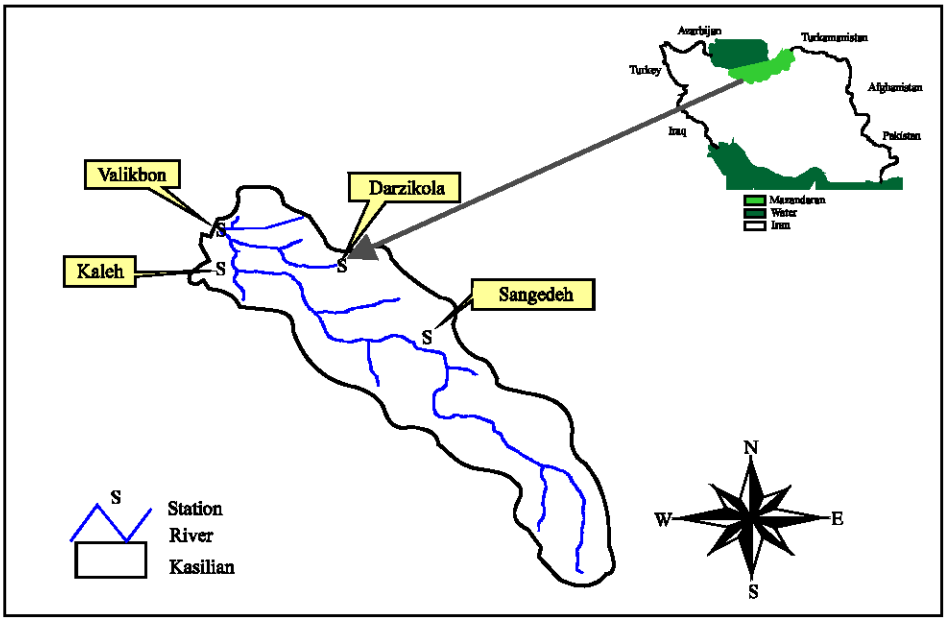

Fig. 1: Geographical location of the study area in Iran

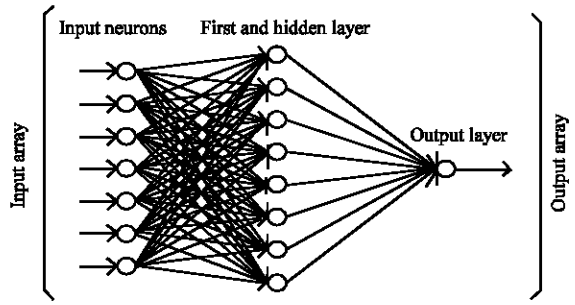

Fig. 2: A typical multi-layer feed forward neural network architecture showing seven neurons in input layer, eight neuron in a hidden layer and a single neuron in the output layer

Figure 2 shows the location of the hydrometric stations of the whole basin. The used data with 10 years duration (1991-2001) for this study included daily rainfall and discharge of Sangedeh, Darzikola and Keleh stations. In this study, finally decide to carry out separate network training by dividing total data into four seasons, namely, spring, summer, fall and winter where smaller number of training patterns was more found to be helpful. A large variation in the statistical parameters across the four seasons may be noted. Data division in this way reduced the chance of unsatisfactory training caused by irregular variations in the input. The divided data were normalized and separate training was done for the four seasons

Analysis of the data: The used discharges data of Kasilian basin were limited with a 10 year duration, which was used for a purposed model development to considering of different combinations of inputs variables, e.g., rainfall, stream flow and average air temperature. For all of the mentioned models available data were separated in 80% for training and 20% for validation. Since in this research the number of data points was more than the number of used parameters by the network (weights and biases), therefore additional analysis for the detection of network overtraining was seemed dispensability. So, data could be divided in two parts for use in the training and validation stages, i.e., cross-validation analysis to stop overtraining was not required (Salehi *et al*., 2000). Data usage by an ANN model typically requires data scaling. This may be due to the particular data behavior or limitations of the transfer functions. For example, as the outputs of the logistic transfer function are between 0 and 1, the data are generally scaled in the range 0.1-0.9 or 0.2-0.8, to avoid problems caused by the limits of the transfer function (Maier and Dandy, 2000). In this study, the data were scaled in the range of -1 to +1, based on the following equation:

$$p_n = 2\left(\frac{p_0 - p_{min}}{p_{max} - p_{min}}\right) - 1 \quad (1)$$

where, p_0 is the observed data, p_n is the scaled data and p_{max}, p_{min} are maximum and minimum observed data points. In order to provide the consistency of analysis, the Eq. 1 was used to scale the average air temperature, stream flow and rainfall data, then the unit of the scaled p_n would correspond to individual data series.

Overview of artificial neural networks: Artificial neural network comprises of a network of neurons and takes the cue from their biological counterparts, in the manner that neurons being capable of learning can be trained to find solutions, recognize patterns, classify data and even forecast future events (Clair and Ehrman, 1998). As a result of these, they have found wide application in simulating very complex relationships and as such found wide application in modeling many hydrological problems including rainfall-runoff modeling and stream flow forecasting (Hsu *et al*., 2002; Riad *et al*., 2004; Mazvimavi *et al*., 2005). Such a network usually comprises of many layers arranged in series, each layer containing one or a group of neurons each of which have the same pattern of connections to the neurons in the other layer(s). The 1st and last layers are used for input and output variables and the intermediate layers are usually connoted as the hidden layer which can be one or many, depending on the complexity of the problem. The weights to a neuron are automatically adjusted by training the network according to a specie learning rule until it properly simulates the past data or performs the desired task. Mathematical functions, known as neuron transfer functions, are used to transform the input to output for each neuron. Two separate stages are involved in the hidden (intermediate) and output layers in transformation of their respective inputs to output. Firstly, an input to a neuron is multiplied by its corresponding weight and the total sum of those input products, to such neuron, plus a constant term yields the neuron net input. The second stage entails transformation of the net input into output. The optimum network architecture (plant model) can then be used to predict future behaviour (forecasting) of the plant. An optimization algorithm is used to select the control input that optimizes future performance. Figure 2 shows a typical example of a one hidden layer feed forward neural network architecture.

Feed Forward Back Propagation Network (FFBP): One of the basic types of a neural network is Multi-Layered Perceptron (MLP), which consists of three layers of neurons namely, input, hidden and output and, where the information flows in the forward direction (Fig. 3). Such a

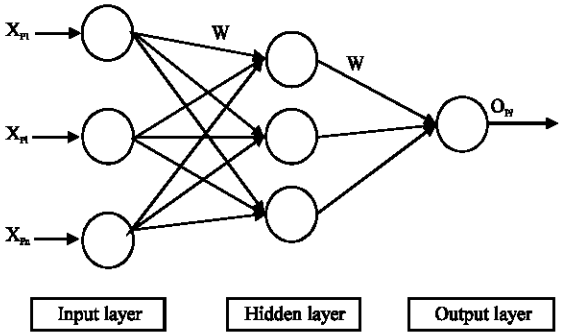

Fig. 3: The FFBP network

feed forward network can be conveniently trained using a simple Back Propagation (BP) algorithm. The resulting network is called Feed Forward Back Propagation (FFBP) and it involves minimization of the error between realized and target outputs according to the steepest or gradient descent approach. In such an approach the correction to a weight, Dw, is made proportional to the rate of change of global error, E, with respect to that weight, i.e.,

$$\Delta w \propto \frac{\partial E}{\partial w} \tag{2}$$

For FFBP, the number of hidden layers and the number of the nodes in the input and hidden layers were determined after trying various network structures. There are several methods to avoid overfitting in ANNs. These methods are summarized by Giustolisi and Laucelli (2005).

The Levenberg-Marquardt algorithm: Several training methods were used, but the Levenberg-Marquardt algorithm proved to be the fastest and the most robust. It is particularly adapted for networks of moderate size and has memory reduction feature for use when the training set is large (Charalambous, 1992). Like the quasi-Newton methods, the Levenberg-Marquardt algorithm was designed to approach second-order training speed without having to compute the Hessian matrix. When the performance function has the form of a sum of squares, then the Hessian matrix can be approximated as:

$$H = J^T J \tag{3}$$

The gradient can be computed as:

$$g = J^T e \tag{4}$$

where, J is the Jacobian matrix that contains first derivatives of the network errors with respect to the weights and biases and e is a vector of network errors. The Jacobian matrix can be computed through a standard back propagation technique that is much less complex than computing the Hessian matrix. The Levenberg-Marquardt algorithm uses this approximation to the Hessian matrix in the following Newton-like update:

$$x_{k+1} = x_k - \left[J^T J + \mu J\right]^{-1} J^T e \tag{5}$$

When, the scalar μ is zero, this is just Newton's method, using the approximate Hessian matrix. When, μ is large, this becomes gradient descent with a small step size. Newton's method is faster and more accurate near an error minimum, so, the aim is to shift towards Newton's method as quickly as possible. Thus, μ is decreased after each successful step and is increased only when a tentative step would increase the performance function. In this way, the performance function will always be reduced at each interaction of the algorithm. The main drawback of the Levenberg-Marquardt algorithm is that it requires the storage of some matrices that can be quite large for certain problems. The size of the Jacobian matrix is Q×n, where Q is the number of training sets and n is the number of weights and biases in the network. It turns out that this matrix does not have to be computed and stored as a whole. For example, if we were to divide the Jacobian into two equal submatrices we could compute the approximate Hessian matrix as follows:

$$H = J^T J = \left[J_1^T J_2^T\right]\begin{bmatrix} J_1 \\ J_2 \end{bmatrix} = J_1^T J_1 + J_2^T J_2 \tag{6}$$

Therefore, the full Jacobian does not have to exist at one time. The approximate Hessian can be computed by summing a series of sub terms. Once one sub term has been computed, the corresponding sub matrix of the Jacobian can be cleared.

Conjugate gradient algorithm: The CG method is a well-known numerical technique used for solving various optimization problems. This technique differs from the standard BP algorithm in updating directions of weights. In other words, the CG algorithm searches a system of conjugate directions on the error surface and updates the weights along these directions. In practice, the process makes good uniform progress toward the solution at every time step and has been found to be effective in finding better optimization than the standard BP algorithm. The CG algorithm is well summarized by Ham and Kostanic (2001).

The used error functions: Root means squared error (RMSE):

$$E_{RMS} = \sqrt{\frac{1}{N}\sum_{m=1}^{N}(Q_{sim(m)} - Q_{obs(m)})^2} \quad (7)$$

where, ERMS is the root mean squared error; $Q_{sim(m)}$ and $Q_{obs(m)}$ are the simulated and observed stream flow values, respectively; and N is the sample size.

Root mean absolute error (RMAE) is given by:

$$E_{RMA} = \frac{E_{MA}}{\bar{Q}} \quad (8)$$

$$E_{MA} = \frac{1}{N}\sum_{m=1}^{N}\left|Q_{sim(m)} - Q_{obs(m)}\right| \quad (9)$$

where, ERMA and EMA are the root mean absolute error and mean absolute error, respectively; Correlation Coefficient (R).

Determining an ANN structure: After a particular ANN architecture and an appropriate training algorithm are selected, the next step is to determine the network structure. The learning ability and performance of an ANN depends on a suitable structure. Generally speaking, the network structure is usually decided on the input dimension, the number of hidden layers, the number of hidden neurons and the output dimension.

Selection of the number of hidden neurons: The determination of the appropriate number of hidden neurons is very important for the success of the ANNs. If the hidden layer has too many neurons this may cause each hidden neuron to memorize one of the input patterns and thereby reduce the generalization capabilities and increase the training time but without significant improvement in training results. Imrie *et al.* (2000) indicate that too few hidden neurons may have insufficient degrees of freedom to capture the underlying relationships in the data. Since, there is not a systematic or standard way to decide on the number of neurons in the hidden layer, the best way to select the hidden neurons is by trial-and-error (Table 1-4).

Selection of input and output dimension: While artificial neural networks are very popular, they lack feedback connections to effectively remember and handle the previous states of information. One way that information can be introduced in artificial neural network is to input a time delay pattern that constitutes the tapped delay line information. Therefore, this network must determine the

Table 1: Comparison of different algorithms and architectures in Spring

			RMSE		MAE	
Models	Algorithms	Architecture	Training	Validation	Training	Validation
Model 1	LM	2-4-1	0.18	0.019	0.01	0.013
	CG	2.10-1	0.49	0.020	0.35	0.014
Model 2	LM	6-6-1	0.27	0.019	0.18	0.010
	CG	6-14	0.30	0.420	0.21	0.340
Model 3	LM	7-8-1	0.23	0.014	0.17	0.011
	CG	7-18-1	0.27	0.040	0.14	0.030

Table 2: Comparison of different algorithms and architectures in Summer

			RMSE		MAE	
Models	Algorithms	Architecture	Training	Validation	Training	Validation
Model 1	LM	2-4-1	0.20	0.020	0.10	0.010
	CG	2-8-1	0.33	0.042	0.20	0.020
Model 2	LM	6-2-1	0.26	0.012	0.19	0.008
	CG	6-12-1	0.56	0.070	0.40	0.040
Model 3	LM	7-16-1	0.23	0.036	0.19	0.029
	CG	7-8-1	0.20	0.040	0.14	0.029

Table 3: Comparison of different algorithms and architectures in Autumn

			RMSE		MAE	
Models	Algorithms	Architecture	Training	Validation	Training	Validation
Model 1	LM	2-2-1	0.10	0.010	0.07	0.080
	CG	2-14-1	0.58	0.018	0.40	0.009
Model 2	LM	6-2-1	0.19	0.010	0.10	0.007
	CG	6-18-1	0.26	0.310	0.15	0.200
Model 3	LM	7-4-1	0.12	0.030	0.07	0.020
	CG	7-2-1	0.15	0.030	0.09	0.020

Table 4: Comparison of different algorithms and architectures in Winter

			RMSE		MAE	
Models	Algorithms	Architecture	Training	Validation	Training	Validation
Model 1	LM	2-4-1	0.12	0.018	0.100	0.009
	CG	2-20-1	0.30	0.100	0.210	0.080
Model 2	LM	6-4-1	0.20	0.013	0.140	0.007
	CG	6-10-1	0.30	0.400	0.190	0.200
Model 3	LM	7-12-1	0.01	0.009	0.008	0.004
	CG	7-4-1	0.096	0.080	0.050	0.040

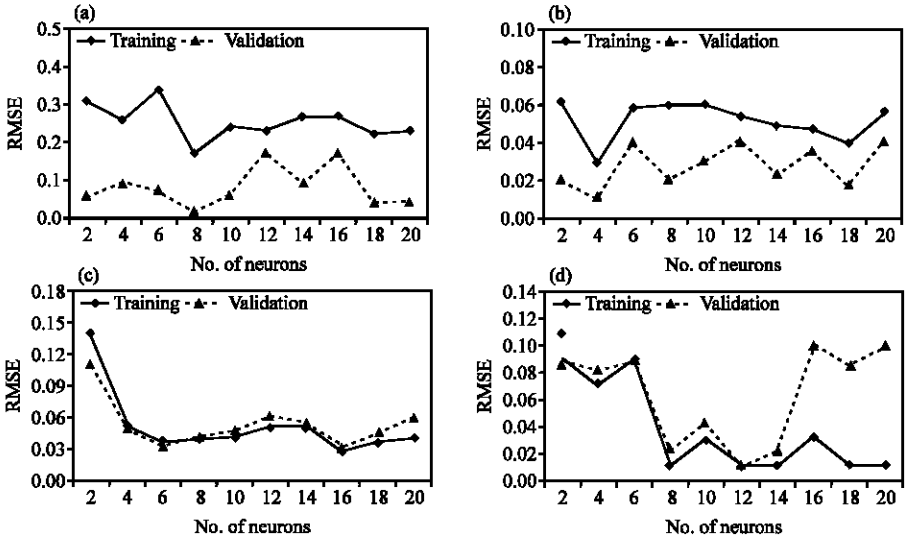

Fig. 4: Comparison of RMSE for the best result in four seasons, (a) Spring, (b) Autumn, (c) Summer and (d) Winter

input variables, output variables and the lag time of the basin before constructing rainfall-runoff procedures in order to increase its accuracy. In general, the selection of input variables and output variables is problem dependent. The appropriate input variables will allow the network to successfully map to the desired output and avoid loss of important information. In the present study, the input dimensions are determined by the input variables and the lag time. To determine an appropriate artificial neural network structure for forecasting the stream flow at time t + 7 in this selected basin, we develop three different models for the LM and CG algorithms, namely:

$$Q(t+7)_{valik} = f\{P(t)_{darzi}, P(t)_{kale}\}$$
$$Q(t+7)_{valik} = f\{P(t)_{darzi}, P(t)_{kale}, P(t-1)_{darzi}, P(t-1)_{kale}, T(t)_{darzi}, Q(t-1)_{valik}\}$$
$$Q(t+7)_{valik} = f\{P(t)_{darzi}, P(t)_{kale}, P(t-1)_{darzi}, P(t-1)_{kale}, Q(t)_{valik}, Q(t-1)_{valik}, Q(t-2)_{valik}\}$$

where, Q(t+7) is predicted rain fall-run off, for the time step of t+7; $Q(t)_{valik}$ is daily rainfall- runoff data of Valikbon hydrometric station $P(t)_{darzi}$ and $P(t)_{kale}$ are daily rainfall data of Darzikola and Keleh rain gauge stations for the time step of t and $T(t)_{darzi}$ is average daily air temperature data at Darzikola station for the time step of t.

Selecting the best structure and algorithm: First, it is use three different input models to determine CG and LM algorithm by using trial-and-error. Figure 4 show the selection of the hidden neurons for the best structure for Lm and CG algorithm, respectively. The hidden neurons from 1 to 20 are varied in each model and all the simulations are terminated after 10,000 iterations. Tables mentioned earlier are show the comparison of different models for LM and CG algorithm, in four seasons, respectively and the performance of different models is presented based on the criteria of Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) during spring, Summer, autumn and Winter. The results show that model 3 produces the best performance for both LM and CG algorithms. This result indirectly provides evidence that the average lag time is 7 day. Comparison of the results of Tables can be concluded that it is easy to tell the LM algorithm is superior to the CG algorithm between the used models and phases. Consequently, it is only use the LM algorithm for training the static-feed forward neural network.

RESULTS AND DISCUSSION

The performances of these two algorithms are evaluated based on the criteria of MAE, RMSE, relative

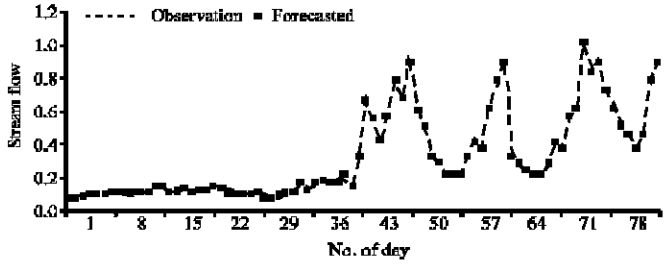

Fig. 5: Observed versus of the forecasted stream flow for a testing period of case 1

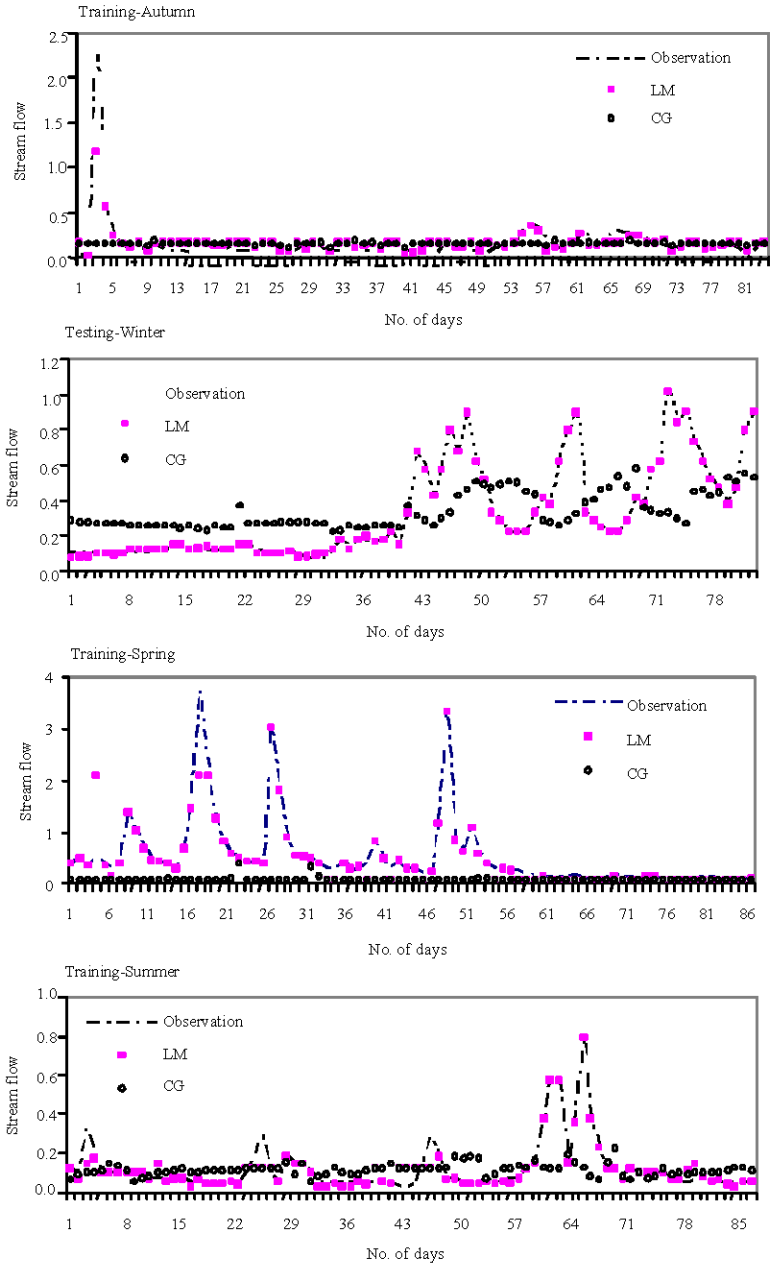

Fig. 6: Comparison of observed versus forecasted daily stream flow for a testing period of model 3 in four season for LM and CG algorithms

Mean Absolute Error (RMAE) and RMSE are defined as follows. The used tables summarize the comparative results of the CG and Lm algorithms for four seasons. It appears that both methods produce good forecasting of the stream flow in all phases. We can also find that the Lm method obtains better performance (in terms of small RMAE and RMSE) than the CG. However, the stream flow values produced by the Lm algorithm are superior to CG. For example, Fig. 5 shows a comparison of observed versus forecasted daily stream flow for a testing period of model 3 in winter. Figure 6a shows a comparison of observed versus forecasted daily stream flow for a testing period of the best model in four seasons for LM and CG algorithms.

CONCLUSIONS

Various types of neural networks have been used to construct the rainfall-runoff models and led to satisfactory results. To learn the stability and effectiveness of two algorithms, trained through different length and content of data, we design three different input patterns and four cases of data sets of the Kasilian River in Iran, validate and test the networks. The LM and CG algorithms are used to search the optimizing values of connecting weights of the neural network, the results show that the LM algorithm is superior to the CG algorithm in terms of efficiency and effectiveness of the constructed network's performance. First, both static and dynamic neural networks yield a reasonably good forecast if there is adequate length and content of data included. Even though, the LM algorithm produces slightly better performance (in terms of small RMAE and RMSE values) than the CG algorithm, we notice that the LM algorithm better captures the peak flows. Second, for which non-sufficient length or content of data is involved in the training phase, the results show that the LM algorithm has significantly better performance than the CG algorithm, especially in the testing phase. The LM algorithm produces poor performance (in terms of larger RMAE and RMSE and highly underestimates the peak flow) in the testing phase. The feed forward neural networks executed fixed-weight mapping from the input space to the output space. Due to fixed weights, the output is solely determined by the present input state to the network and not the initial and past states of the neurons in the network. Consequently, the input-output mapping of the feed forward neural networks is static and the networks would be trained to store the information content in the training data and could only produce the memory structures within processes. On the contrary, the feedback neural networks do allow initial and past state involvement along with serial processing. As indicated by the results, model 5 provided the highest performance. This was due to enabling of the precipitation and rainfall, resulting in improved training and thus improved prediction. This study has shown that when improved computational efficiency measures are combined with enabling of input parameters describing physical behavior of hydro-climatologic variables, improved model predictability becomes possible by the artificial neural networks.

ACKNOWLEDGMENT

The authors would like to thanks the Centre of GIS and RS, Faculty of Natural Resources, Sari, for technical and financial support.

REFERENCES

Abebe, A.J., D.P. Solomatine and R.G.W. Venneker, 2000. Application of adaptive fuzzy rule-based models for reconstruction of missing precipitation events. Hydrol. Sci. J., 45: 425-436.

ASCE Task Committee, 2000a. Artificial neural networks in Hydrology I. J. Hydro. Eng., ASCE, 5: 115-123.

ASCE Task Committee, 2000b. Artificial neural networks in Hydrology II. J. Hydro. Eng. ASCE, 5: 124-132.

Chang, F.J., L.C. Chang and H.L. Huang, 2002. Real-time recurrent learning neural network for stream-flow forecasting. Hydrol. Processes, 16: 2577-2588.

Chang, L.C., F.J. Chang and Y.M. Chiang, 2004. A two-step-ahead recurrent neural network for stream-flow forecasting. Hydrol. Processes, 18: 81-92.

Charalambous, C., 1992. Conjugate gradient algorithm for efficient training of artificial neural networks. IEEE. Proc., 139: 301-310.

Chiang, Y.M., L.C. Chang and F.J. Chang, 2004. Comparison of static-feedforward and dynamic-fedback neural networks for rainfall-runoff modeling. J. Hydrol., 290: 297-311.

Cigizoglu, H.K., 2005. Application of the generalized regression neural networks to intermittent flow forecasting and estimation. ASCE J. Hydrol. Eng., 10: 336-341.

Cigizoglu, H.K. and O. Kisi, 2006. Methods to improve the neural network performance in suspended sediment estimation. J. Hydrol., 317: 221-238.

Clair, T.A. and J.M. Ehrman, 1998. Neural networks to assess influence of changing seasonal climates in modifying discharge, dissolved organic carbon and nitrogen export in eastern Canadian rivers. Water Resour. Res., 34: 447-455.

Coulibaly, P., Y.B. Dibike and F. Anctil, 2005. Downscaling precipitation and temperature with temporal neural networks. J. Hydrometeorol., 6: 483-496.

Coulibaly, P. and N.D. Evora, 2007. Comparison of neural network methods for infilling missing daily weather records. J. Hydrol., 341: 27-41.

Dibike, Y.B. and P. Coulibaly, 2006. Temporal neural networks for downscaling climate variability and extremes. Neural Networks, 19: 135-144.

El-Bakyr, M.Y., 2003. Feed forward neural networks modeling for K-P interactions. Chaos, Solit. Fractals, 18: 995-1000 (Elsevier).

French, M.N., W.F. Krajewski and R.R. Cuykendal, 1992. Rainfall forecasting in space and time using a neural network. J. Hydrol., 137: 1-37.

Garrote, L. and R.L. Bras, 1995. An integrated software environment for real-time use of a distributed hydrologic model. J. Hydrol., 167: 307-326.

Giustolisi, O. and D. Laucelli, 2005. Improving generalization of artificial neural networks in rainfall runoff modeling. Hydrol. Sci. J., 50: 439-457.

Halff, A.H., H.M. Halff and M. Azmoodeh, 1993. Predicting runoff from rainfall using neural networks. Proceeding of Engineering Hydrol. 1993 ASCE, New York, 760-765.

Ham, F.M. and I. Kostanic, 2001. Principles of Neurocomputing for Science and Engineering. 1st Edn., McGraw-Hill, ISBN: 0070259666, pp: 479.

Hsu, K., H.V. Gupta, Gao, S. Sorooshian and B. Imam, 2002. Self organizing linear output map (SOLO): An artificial neural network suitable for hydrologic modelling and analysis. J. Hydrol., 38: 1-17.

Hu, T.S., K.C. Lam and S.T. Ng, 2001. River flow time series prediction with a range-dependent neural network. Hydrol. Sci. J., 46: 729-745.

Imrie, C.E., S. Durucan and A. Korre, 2000. River flow prediction using artificial neural networks: Generalisation beyond the calibration range. J. Hydrol., 233: 138-153.

Kisi, O., 2004. River flow modeling using artificial neural networks. J. Hydrol. Eng., 9: 60-63.

Kumar, D.N., K.S. Raju and T. Sathish, 2004. River flow forecasting using recurrent neural networks. Water Resour. Mange., 18: 143-161.

Lin, G.F. and L.H. Chen, 2004. A non-linear rainfall-runoff model using radial basis function network. J. Hydrol., 289: 1-8.

Luk, K.C., J.E. Ball and A. Sharma, 2000. A study of optimal model lag and spatial inputs to artificial neural network for rainfall forecasting. J. Hydrol., 227: 56-65.

Luk, K.C., J.E. Ball and A. Sharma, 2001. An application of artificial neural networks for rainfall forecasting. Math. Computer Modell., 33: 683-693.

Maier, H.R. and G.C. Dandy, 2000. Neural networks for prediction and forecasting of water resources variables: A review of modeling issues and applications. Environ. Model. Software, 15: 101-123.

Mazvimavi, D., A.M.J. Maijerink, H.H.G. Savenije and A. Stein, 2005. Prediction of flow characteristics using multiple regression and neural networks: A case study in Zimbabwe. Phys. Chem. Earth, 30: 639-647.

Nayak, P.C., K.P. Sudheer, D.M. Rangan and K.S. Ramasastri, 2005. Short-term flood forecasting with a neurofuzzy model. Water Resour. Res., 41: 2517-2530.

Olsson, J., C.B. Uvo, K. Jinno, A. Kawamura, K. Nishiyama, N. Koreeda, T. Nakashima and O. Morita, 2004. Neural networks for rainfall forecasting by atmospheric downscaling. J. Hydrol. Eng., 9: 1-12.

Rajurkara, M.P., U.C. Kothyarib and U.C. Chaubec, 2004. Modeling of the daily rainfall-runoff relationship with artificial neural network. J. Hydrol., 285: 93-113.

Rami'rez, M.C.P., H.F.C. Velho and N.J. Ferreira, 2005. Artificial neural network technique for rainfall forecasting applied to the Saõ Paulo region. J. Hydrol., 301: 146-162.

Riad, S., J. Mania, L. Bouchaou and Y. Najjar, 2004. Predicting catchment flow in a semi-arid region via an artificial neural network technique. Hydrol. Processes J., 18: 2387-2393.

Salehi, F., S.O. Prasher, S. Amin, A. Madani, S.J. Jebelli, H.S. Ramaswamy and C.T. Drury, 2000. Prediction of annual nitrate-n losses in drain outflows with artificial neural networks. Trans. ASAE, 43: 1137-1143.

Sudheer, K.P., A.K. Gosain and K.S. Ramasastri, 2002. A data-driven algorithm for constructing artificial neural network rainfall-runoff models. Hydrol. Process., 16: 1325-1330.