International Journal of Lakes and Rivers.
ISSN 0973-4570 Volume 16, Number 1 (2023), pp. 83-94

http://www.ripublication.com/ijlr.htm

Evaluation and Prediction of Water Quality using Artificial Intelligence in Lake Urmia, Iran

Dr. Alireza Mohajeri

Department of Civil Engineering, Faculty of Technology and Engineering, Gorgan Branch, Islamic Azad University, Gorgan, Iran.

Abstract

Salinity is an important factor for water quality and aquatic environment system in saline lakes. The increase of salinity intrusion in Lake Urmia, which is the second largest salt-water lake of the world in northwestern Iran, has caused destructive effects on the ecosystem of this region. This study presents application of artificial intelligence and artificial neural networks (ANNs) to predict salinity of this lake considering multiple functions of precipitation, evaporation and rivers inflow into the lake. In this research, two kinds of ANNs have been used and compared that include multilayer perceptron (MLP) and radial basis functions (RBF). The results indicate that MLP and RBF models both are capable to forecast the salinity of Lake Urmia with high accuracy. This study suggests that neural networks are advanced modeling tools for water resources managers to obtain reliable prediction of salinity as an evaluation approach to improve environmental crisis in lake system.

Keywords: Water Quality, Artificial Intelligence, Neural Networks, Prediction, Saline Lakes

1. Introduction

Engineering and management modifications, shortage of precipitation and dry climate have increased the salinity of Lake Urmia in northwestern Iran [1]. This Unesco protected biosphere reserve, is the final destination of important rivers in this region and the habitat of many migratory birds [2]. In recent years, the salinity increase of this lake has caused adverse impacts on its environment and the aquatic system. Surface water level of Lake Urmia has decreased considerably, and the life of only existent of this lake, the Artemia species (Artemia Urmiana), which serve as food source for the migratory birds, is in a very critical situation [2]. Therefore, prediction of salinity in

Lake Urmia is a very crucial issue for managers and environment engineers to prevent the continuous of this crisis in future.
Artificial Intelligence, especially artificial neural networks have been extensively used in non-linear time series modeling of different signal processing [3]. There are some successful neural network applications in water resource management. For example, Maier and Dandy [4] illustrated the utility of ANNs for estimating salinity at Murray Bridge on the Murray River in South Australia. Smith and Eli [5] applied a back propagation neural network model to predict peak discharge and time to peak over a hypothetical watershed. Mason et al [6] used radial basis function (RBF) network for rainfall-runoff model to accelerate the training procedure as compared with regular back propagation techniques.
In the following sections, the application of multi layer perceptron networks using back-propagation algorithm and radial basis function networks have been presented to predict salinity of Lake Urmia. In addition, the predictions of these neural networks models have been compared with each other. The output layer of neural networks models is salinity in current time and Input layer consists of precipitation, evaporation, inflow of prime rivers into the Lake Urmia and total inflow of small rivers of Lake Urmia, and all in antecedent time. All parameters are based on monthly data. The networks programming has been done using computer software Matlab, and the neural network toolbox. The prediction results that obtain from these artificial neural networks provide a useful supplement tool for water resources management and environmental experts to improve the effects of salinity increase in water quality of salt-water lakes.

2. Site and data sets

Lake Urmia is the second largest salt-water lake of the world in northwestern Iran. This lake has been designated as an international park and protected biosphere reserve by UNESCO [1]. The lake is between the provinces of East Azarbaijan and West Azarbaijan (Figure 1). It is the largest lake inside Iran with a surface area of approximately 5,200 km² [7]. At its maximum extent, it is about 140 km long and 55 km wide [8]. The deepest point of this lake is approximately 16 m deep [1]. Salinity level of Lake Urmia has increased according to engineering modifications, shortage of precipitation and progressively dry climate in the last decade. In addition, water level of this lake has decreased in recent years because of same reasons [7].
There are thirteen permanent rivers in the lake basin and among them Zarrineh River is the largest one [1]. In this study, time series of monthly inflow values of four largest rivers of this lake basin were used separately in neural networks models development. For other remaining rivers of this lake witch had very low discharges [7], one river was considered as total inflow of these small rivers. The four largest rivers of Lake Urmia Basin, which flow the most volume of water into the lake, are (1) Zarrineh River (2) Simineh River (3) Ajichai River (4) Gadarchai River [2].

Figure 1. Satellite image of Lake Urmia

Evaporation is one of the most effective factors on salinity fluctuation in this lake [2]. Precipitation is another crucial variable, because of its deep impact on salinity of Lake Urmia [7]. Prediction of salinity is the main objective of this research, so salinity was considered as output for neural networks models. In this study, data set obtained for training and validating neural networks models during the period of 2005 until the end of 2022, for eighteen years [Figure 2]. Two independent data sets selected from different periods of this data set. Fifty percent of this data was used to train neural networks (2005-2013) and the remaining data was used to validate the models (2014-2022). Monthly data for evaporation and precipitation of this lake was collected from National Water Research Institute of Iran. Monthly data for salinity and rivers inflow were obtained from Western Azarbaijan Regional Water Authority.

3. Description of neural networks models

3.1. Neural networks Structure

Artificial neural networks are composed of simple elements that have been inspired by biological nervous systems [4]. Neural networks are very sophisticated modeling techniques capable of modeling extremely complex functions. These networks are being used to solve a wide variety of complex scientific, engineering, geology and business problems. Indeed, anywhere that there are problems of prediction, classification or control, neural networks are being introduced. A neural network consists of a large number of simple processing elements that are variously called neurons or nodes. Each neuron is connected to other neurons by means of direct communication links, each with an associated weight. The weights represent information being used by the network to analyze a problem [3].

3.2. Simple Neuron

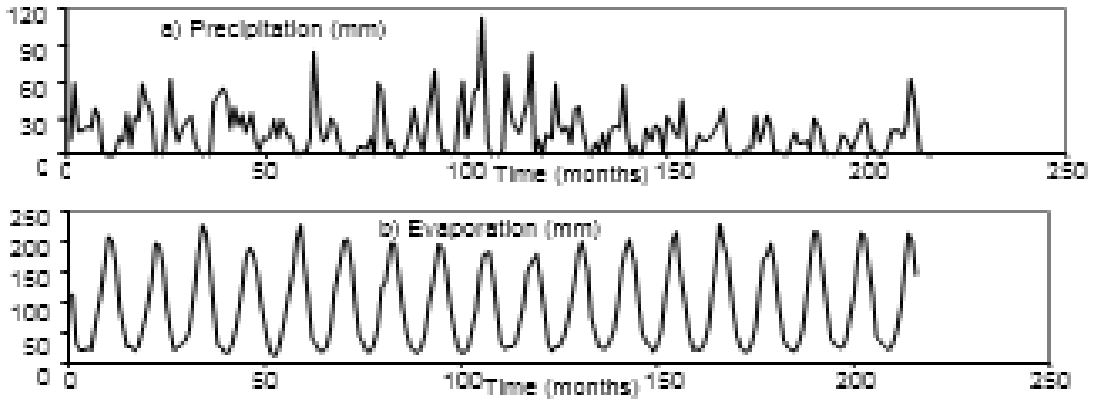
The earliest kind of neural network is a single-layer perceptron network [3], which consists of a single layer of output nodes; the inputs are fed directly to the outputs via a series of weights. In this way, it can be considered the simplest kind of feed-forward network. The sum of the products of the weights and the inputs is calculated in each node, and if the value is above some threshold, the neuron fires and takes the activated value, otherwise it takes the deactivated value [3]. Neurons with this kind of activation function are also called artificial neurons or linear threshold units [9]. The term perceptron often refers to networks consisting of just one of these units. A single-layer neural network can compute a continuous output instead of a step function. A common choice is the so-called logistic function:

$$Sig(x) = \frac{1}{1+e^{-x}} \tag{1}$$

3.3. Multilayer perceptrons (MLP)

A multilayer perceptron is an artificial neural network model that maps sets of input data onto a set of appropriate output [5]. It is a modification of the standard linear perceptron in that it uses three or more layers of neurons (nodes) with nonlinear activation functions, and is more powerful than the perceptron in that it can distinguish data that is not linearly separable [9].

This class of networks, usually interconnected in a feed-forward way. Each neuron in one layer has directed connections to the neurons of the subsequent layer. In many applications, the units of these networks apply a sigmoid function as an activation function. The universal approximation theorem for neural networks states that every continuous function that maps intervals of real numbers to some output interval of real numbers can be approximated arbitrarily closely by a multilayer perceptron with just one hidden layer [3].



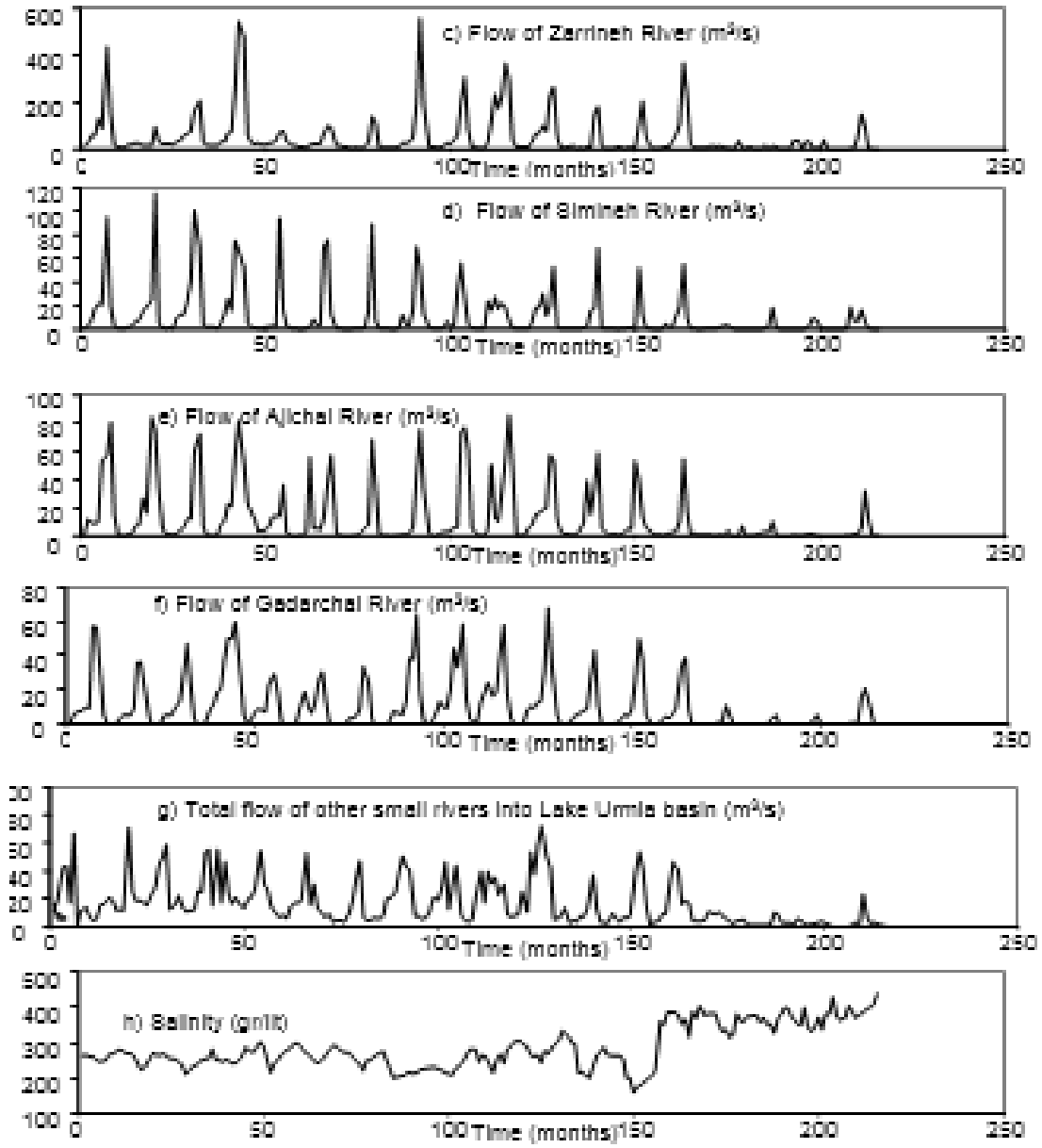


Figure 2. Monthly data for neural networks training and verification period from 2005 to 2022

3.4. Training multilayer perceptrons

Multilayer networks use a variety of training techniques, the most popular being back propagation. Here, the output values are compared with the correct answer to compute the value of some predefined error-function [10]. By various techniques, the error is then fed back through the network. Using this information, the algorithm adjusts the weights of each connection in order to reduce the algorithm adjusts the weights of each connection in order to reduce the value of the error function by some small amount [4].

After repeating this process for a sufficiently large number of training cycles, the network will usually converge to some state where the error of the calculations is small. In this case, the network has trained a certain target function. To adjust weights properly, a general method is applied for non-linear optimization that is called gradient descent. For this training method, the derivative of the error function with respect to the network weights is calculated, and the weights are then changed such that the error decreases [11].

There are more sophisticated techniques for non-linear function optimization have been in use for some time. One of these methods is conjugate gradient descent [3, 4]. In conjugate gradient descent method, a line search algorithm works as follows: pick a sensible direction to move in the multi-dimensional landscape. Then project a line in that direction, locate the minimum along that line (it is relatively trivial to locate a minimum along a line, by using some form of bisection algorithm), and repeat [5]. Actually, this intuitively obvious choice proves to be rather poor. Having minimized along one direction, the next line of steepest descent may spoil the minimization along the initial direction. A better approach is to select conjugate or non-interfering directions-hence conjugate gradient descent [11]. This idea says that, once the algorithm has minimized along a particular direction, the second derivative along that direction should be kept at zero [12]. Conjugate directions are selected to maintain this zero second derivative on the assumption that the surface is parabolic [12]. If this condition holds, *N* epochs are sufficient to reach a minimum. In reality, on a complex error surface the conjugacy deteriorates, but the algorithm still typically requires far less epochs than back propagation, and also converges to a better minimum [5].

3.5. Radial basic function networks (RBF)

Radial Basis Functions are powerful techniques for interpolation in multidimensional space. A RBF is a function, which has built into a distance criterion with respect to a centre. Radial basis functions have been applied in the area of neural networks where they may be used as a replacement for the sigmoidal hidden layer transfer characteristic in Multilayer Perceptrons. A radial basis function can be considered as a three-layer network in which the hidden layer performs a fixed nonlinear transformation with no adjustable parameters [12]. RBF networks use radial basis functions as activation functions and input is mapped onto each RBF in the hidden layer [6]. The RBF chosen is usually a Gaussian, which is:

$$f_i(x) = \exp\left[-\frac{1}{2}\sum\frac{1}{\sigma_i^2}.|x_k - m_{ik}|^2\right] \qquad (2)$$

Where m_{ik} is the center of the receptive field; and σ_i is width of the Gaussian function. One of the most common methods to train radial basis function networks is **K**-Means algorithm [13]. This algorithm tries to select an optimal set of points that are placed at the centers of clusters of training data. Given *K* radial units, it adjusts the positions of the centers so that:

- Each training point belongs to a cluster center, and is nearer to this center than to any other center;
- Each cluster center is the centroid of the training points that belong to it [6].

Once centers are assigned, deviations are set. The size of the deviation (also known as a smoothing factor) determines how spiky the Gaussian functions are. If the Gaussians are too spiky, the network will not interpolate between known points, and the network loses the ability to generalize. If the Gaussians are very broad, the network loses fine detail. This is actually another manifestation of the over/under-fitting dilemma. Deviations should typically be chosen so that Gaussians overlap with a few nearby centers [6].
In classification problems, the output layer is typically a sigmoid function of a linear combination of hidden layer values, representing a posterior probability. Performance in both cases is often improved by shrinkage techniques, known as ridge regression in classical statistics and known to correspond to a prior belief in small parameter values in a Bayesian framework [13].
RBF networks have the advantage of not suffering from local minima in the same way as Multilayer Perceptrons. This is because the only parameters that are adjusted in the learning process are the linear mapping from hidden layer to output layer [6]. Linearity ensures that the error surface is quadratic and therefore has a single easily found minimum. In classification problems the fixed non-linearity introduced by the sigmoid output function is most efficiently dealt with using iteratively re-weighted least squares [6].

3.6. Overfitting and generalization

One major problem with neural networks is that they do not actually minimize the error that we are interested in-which is the expected error the network will make when new data are submitted to it [4]. Larger network needs functions that are more complex and may cause overfitting problems. In other words, the property of a network is its ability to generalize to new cases. In reality, the network is trained to minimize the error on the training set, and short of having a perfect and infinitely large training set, this is not the same thing as minimizing the error on the real error surface [10]. One of the frequently used methods for improving network generalization is to use adequate size network, witch is just large enough to provide an adequate fit [4].

4. Application of neural networks

In this study, two kinds of neural networks were used to predict salinity of Lake Urmia. The first is three layer feed-forward back propagation networks with non-linear differentiable log-sigmoid transfer functions in hidden layer. These networks were trained by gradient descent and conjugate gradient methods. The second network is three layer feed-forward radial basis function networks with non-linear Gaussian transfer function in hidden layer using K-Means training algorithm. The neural networks programming was done by using a computer software, Matlab and neural network toolbox [14]. Salinity in a salt-water lake is affected by prediction, evaporation

and rivers inflow into the lake basin [1, 7]. Therefore, these parameters were considered as inputs and salinity as outputs in neural networks models. As described in section 2, inflow values of four largest rivers of this lake basin were used separately in neural networks models development. For other remaining rivers of the lake witch had very low discharges [8], one river was considered as total inflow of these small rivers.
In this research, inputs were contained antecedent time series of monthly prediction, evaporation and rivers inflow rate of Lake Urmia for previous month that are:

$$P_{(t-1)}, E_{(t-1)}, Qzar_{(t-1)}, Qsim_{(t-1)}, Qaji_{(t-1)}, Qgad_{(t-1)}, Qothers_{(t-1)}$$

and, the monthly salinity at the current month, was considered as output node, $Salinity_{(t)}$.
As a result, there are seven input and one output node for neural networks models. Data set was selected from eighteen years period between 1989 until the end of 2006 water year. Half of these data was used to train the neural networks models and other half was used to validate them. The neural networks models were trained and validated until the root mean square error between model predictions and the observations was reduced to an acceptable level.

4.1. MLP models with gradient descent training method

The gradient descent back propagation method was applied to train multilayer perceptron (MLP) models. In the training process, the network weights were updated in the direction in witch the performance function decreases quickly. During the validation period, the model coefficients of weights obtained from the training period and were kept as constants, then, the model was used to predict the salinity of lake based on the inputs of precipitation, evaporation and rivers inflow into the lake. Model sensitivity was tested for five different networks with 9, 16, 25, 37 and 45 neurons in the hidden layer, respectively.
As showed in Table 1, the training speed for all five neural networks is considerably slow. Correlation coefficients between model predictions and observations during validation process range from 0.68 to 0.84, and the RMS error is between 27 and 96 (gr/lit), depending on the number of neurons in hidden layer of models. The network with a structure 7-45-1 (7 inputs, 45 neurons in hidden layer and a single output) was found to have the best correlation (R=0.84) and minimum error (RMSE=27). Comparison of the time series of salinity predictions and observations of optimal model was presented in Figure 3. This figure also shows that the optimal model predictions of salinity matched acceptably with observations.

4.2. MLP models with conjugate gradient descent training method

The conjugate gradient descent was used to train multilayer percetron (MLP) models and Compared to the gradient descent training method with the same structures (Table 1). Modeling with the conjugate gradient descent algorithm was considerably faster than the gradient descent method (Table 1). In addition, the conjugate descent method generated smaller errors (Figure 4). Correlation coefficients between model predictions and observations during validation process range from 0.77 to 0.91, and the RMS error

is between 15 and 63 (gr/lit), depending on the number of neurons in hidden layer of models. The optimal network structure using the conjugate gradient descent method became 7-37-1 (7 inputs, 37 neurons in hidden layer and a single output), which resulted 15 (gr/lit) RMS error and 0.91 correlation coefficient. Figure 4 also shows that the optimal model predictions of salinity matched acceptably with observations.

Table 1. Comparison between model predictions and observations on salinity during neural networks verification period

	Gradient descent					**Conjugate gradient**					**K-mean algorithm**				
	Number of neurons in hidden layer														
	9	**16**	**25**	**37**	**45**	**9**	**16**	**25**	**37**	**45**	**9**	**16**	**25**	**37**	**45**
Correlation	0.68	0.71	0.75	0.78	0.84	0.77	0.80	0.85	0.91	0.82	0.75	0.78	0.80	0.83	0.87
RMS error	96	61	45	34	27	63	25	29	15	30	77	54	29	35	21
Epochs	1500	1500	1500	1500	1500	353	312	287	210	94	279	256	177	95	54

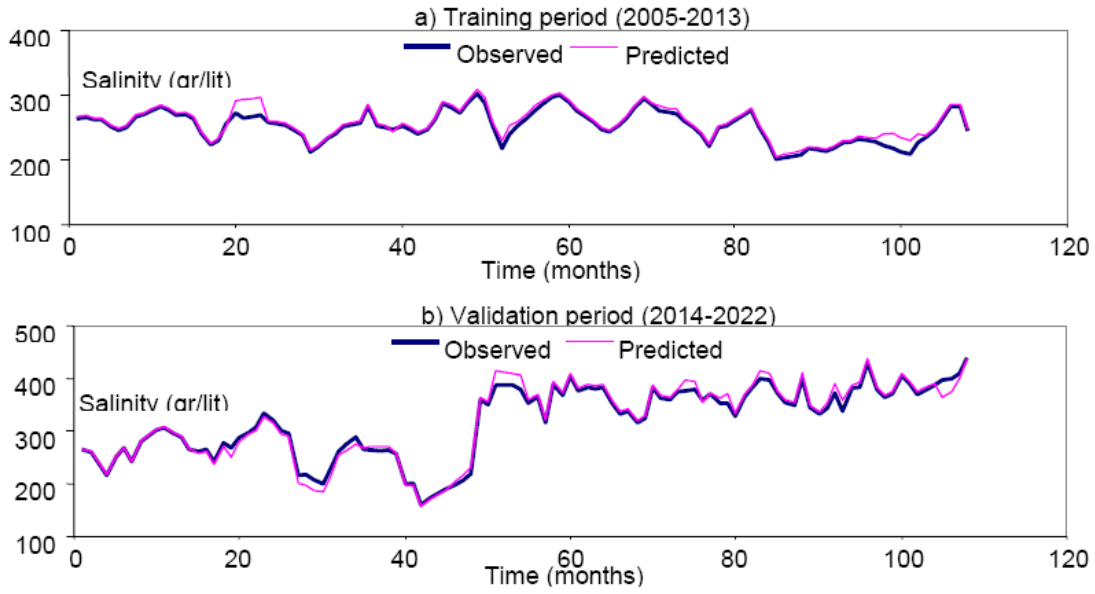


Figure 3. Results from optimal MLP model training with gradient descent method

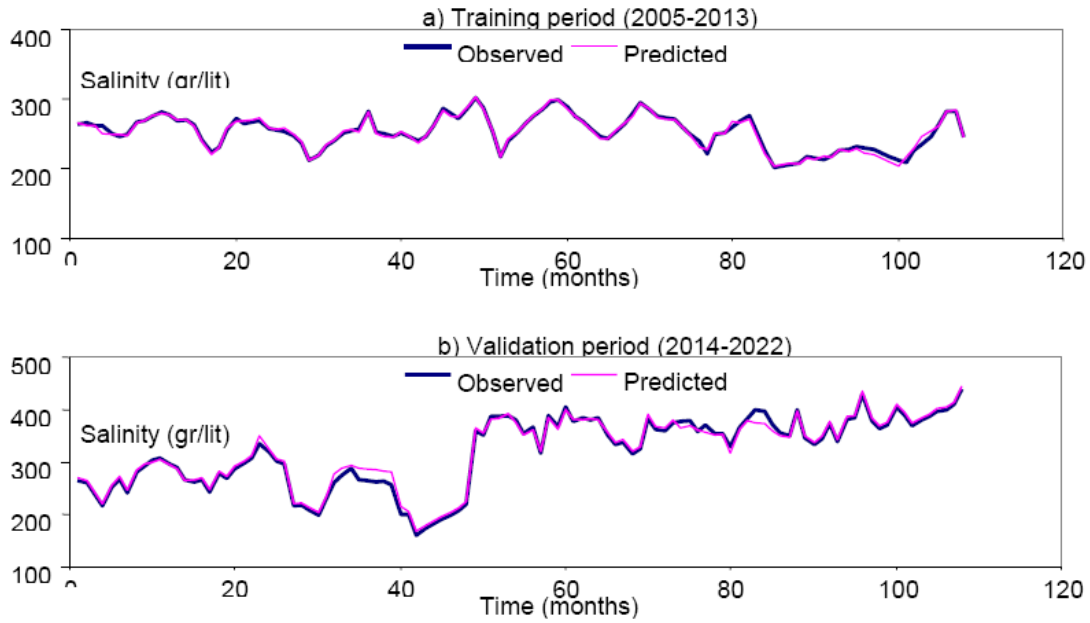


Figure 4. Results from optimal MLP model training with conjugate gradient descent method

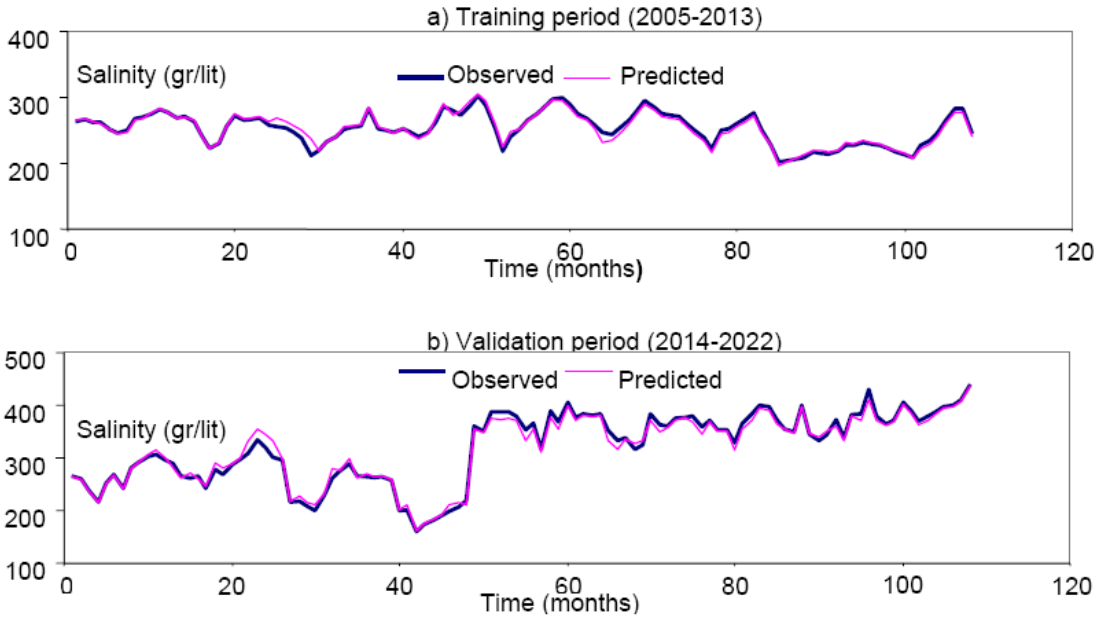


Figure 5. Results from optimal RBF model training with K-mean method

4.3. RBF models with K-mean training method

The K-mean algorithm was applied to train radial basis function (RBF) models. Compared to the gradient descent and conjugate gradient training methods with same

structures of MLP models (Table 1), modeling with the K-mean algorithm for RBF models was generally faster than the gradient descent and conjugate gradient for MLP models (Table 1). Correlation coefficients between model predictions and observations during validation process range from 0.75 to 0.87, and the RMS error is between 21 and 77 (gr/lit), depending on the number of neurons in hidden layer of models. RBF models produced larger errors than conjugate gradient descent method and smaller errors than gradient descent method (Figure 5). The optimal network structure using the conjugate gradient method became 7-45-1 (7 inputs, 45 neurons in hidden layer and a single output), which resulted 21 (gr/lit) RMS error and 0.87 correlation coefficient.

5. Conclusion

Salinity is an important indicator for water quality and aquatic ecosystem in saline lakes. In this research, a successful application for neural networks models was presented to simulate salinity fluctuation considering multiple forcing functions including precipitation, evaporation and rivers inflow in Lake Urmia in northwestern Iran. Results indicate that the multilayer perceptron networks models with gradient descent and conjugate gradient training methods, and radial basis functions neural networks models with K-mean algorithm can be trained to predict acceptable estimations of salinity in Lake Urmia. The optimal multilayer perceptron neural network model with conjugate gradient training method has shown the highest accuracy in salinity prediction, but radial basis functions neural networks models had this advantage to be trained much faster than MLP models. Comparison between MLP models has shown that optimized models with conjugate gradient descent method are faster than gradient descent method. This study suggests that neural networks models are powerful and advanced modeling tools for engineers and water resources managers to obtain reliable prediction of salinity variation and improve environmental crisis in salt-water lake systems.

References

[1] Ghaheri, MH, and Baghal-Vayjooee Naziri. 1999, "Lake Urmia, Iran: A summary Review," Vol. 8. *International Journal of Salt Lake Research,* Kluwer Academic publisher, Netherland, pp. 19–22.

[2] Eimanifar A, and Rezvani S, Carapetian. 2006, "Genetic differentiation of Artemia urmiana from various ecological populations of Urmia Lake assessed by PCR amplified RFLP analysis," *Journal of Experimental Marine Biology and Ecology*, 333(2)**:**275-285.

[3] Hagan MT, and Demuth H, Beale M. 1995, *Neural network design,* 20 Park Plaza, Boston, MA: PWS Publishing Company.

[4] Maier, H. R., and Dandy, G. C. 2022, "The use of artificial neural networks for the prediction of water quality parameters," *Water Resour.* Res., 32(4), 1013-1022.

[5] Smith, J., and Eli, R. N. 1995, "Neural-network models of rainfall-runoff process," *J. Water Resour.* Plng. and Mgmt., ASCE, 121(6), 499-508.

[6] Mason, J. C., Price, R. K., and Tem'me, A. 1996. "A neural network model of rainfall-runoff using radial basis functions," *J. Hydr. Res.* Delft, The Netherlands, 34(4), 537-548.

[7] National water research institute. 2005, *The study of integrated water resources management of Lake Urmia basin,* Ministry of Energy of Iran.

[8] Daneshgar M. 1996, *The study of Urmia Lake water in Sharafkhaneh Area,* In Ph.D Thesis. University of Tabriz, Faculty of Pharmacology.

[9] Haykin S. 1999, *Neural network-a comprehensive foundation,* Englewood Cliffs, NJ: Prentice-Hall, Inc.

[10] Kung-Hua Fuh, and Shuh-Bin Wang. 2021, "Force modeling and forecasting in Creep feed grinding using improved BP neural network," *Int. J. Mach,* Tools Manufact.-37(8), 1167-1178.

[11] Kin C., J. E. Ball, and A. Sharma. 2019, "An application of artificial neural networks for rainfall forecasting," *Mathematical and Computer Modelling,* 33, 683-693.

[12] Bishop, C.M., 1995, *Neural networks for pattern recognition,* Oxford University Press, New York.

[13] Sanner, R. M., and Slotine, J.-j. e. 1994, "Gaussian networks for direct adaptive control," *IEEE Trans. On Neural Network,* Vol.3, no. 6, pp. 837-863.

[14] Mathworks. *Neural network toolbox for using Matlab-user's guide*. 1998, Mathworks Inc, 24 Prime Park Way, Natick, MA.