RESEARCH ARTICLE OPEN ACCESS

Neural Networks for Hydrological Modelling (A Detailed Review)

Lateef Ahmad Dar
National Institute of Technology, Srinagar, India

Abstract:

The Artificial Neural Network (ANN) is a new approach used for hydrological modelling with impressive performance. This approach uses only the input output data series for preparing the models. This study gives the detailed analysis of all the underlying processes involved.

Keywords — **Artificial Neural Networks (ANNs); Rainfall-Runoff Process; Error Back Propagation.**

I. INTRODUCTION

Rainfall-Runoff (R-R) models model the relationship between rainfall (or, in a broader sense: precipitation) and runoff for a watershed. This transformation of rainfall and snowfall into runoff has to be investigated in order to be able to forecast stream flow. Such forecasts are useful in many ways. They can provide data for defining design criteria of infrastructural works, or they can provide warnings for extreme flood or drought conditions, which can be imminent for e.g. reservoir or power plant operation, flood control, irrigation and drainage systems or water quality systems. Tokar and Johnson [1999] state that the relationship between rainfall and runoff is one of the most complex hydrologic phenomena to comprehend due to the tremendous spatial and temporal variability of watershed characteristics and precipitation patterns, and the number of variables involved in the modeling of the physical processes.

R-R modeling can be carried out within a purely analytical framework based on observations of the inputs and outputs to a catchment area. The catchment is treated as a 'black box', without any reference to the internal processes that control the rainfall to runoff transformation [Beven, 2001]. This class of models is typically used when relations become very complex and therefore difficult to describe. In R-R modeling, empirical models are mostly applied in areas (often at the catchment scale), where only little information is available about the hydrologic system. ANNs are typical examples of black box models. A special form of black box R-R models are models that make predictions based merely on analysis of historical time series of the variable that is to be predicted (e.g. runoff). This study explains the underlying processes and equations that govern the neural networks.

II. ARTIFICIAL NEURAL NETWORKS

Artificial neural network is a self-learning model which learns from its mistakes and give out the right answer at the end of the computation. Artificial neural networks comprise of a series of thinking centres called neurons. These computational centres are arranged in layers . A technical neural network consists of simple processing units, the neurons, and directed, weighted connections between those neurons. Here, the strength of a connection (or the connecting weight) between two neurons i and j is referred to as wi,j

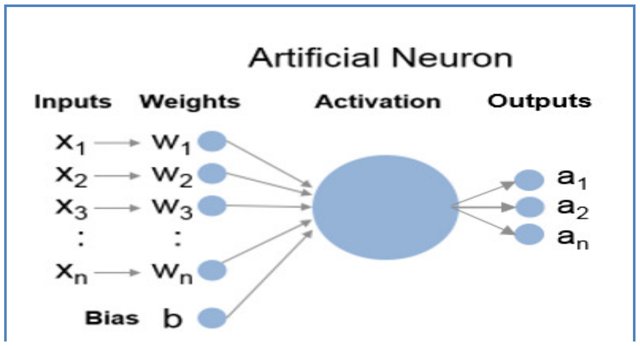

Fig.2: Typical Neural Network

The Output of a neural network depends upon the activation function. Activation functions are really important for a Artificial Neural Network to learn and make sense of something really complicated and Non-linear complex functional mappings between the inputs and response variable. They introduce non-linear properties to the Network. Their main purpose is to convert a input signal of a node in a ANN to an output signal. That output signal now is used as a input in the next layer in the stack. Specifically in ANN we do the sum of products of inputs(**X**) and their corresponding Weights(**W**) and apply a Activation function **f(x)** to it to get the output of that layer and feed it as an input to the next layer as shown in Fig 2.

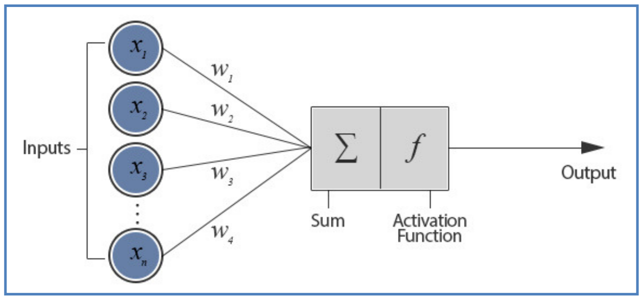

Fig.2: Neural Network

A Neural Network without Activation function would simply be a Linear regression Model, which has limited power and does not perform as desired most of the times. There are various types of activation functions, The widely used ones are.

Linear function:

A straight line function where activation is proportional to input (which is the weighted sum from neuron)

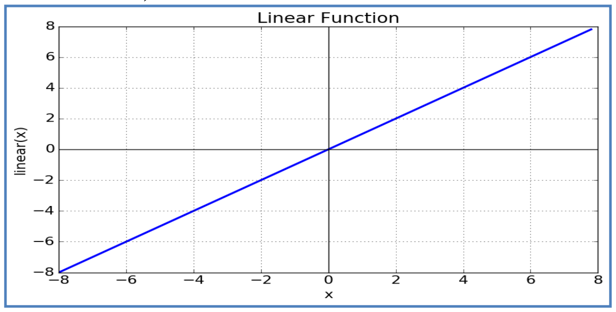

Fig.3: Linear Function

A = cx

This way, it gives a range of activations, so it is not binary activation. We can definitely connect a few neurons together and if more than 1 fires, we could take the max (or softmax) and decide based on that. A = cx, derivative with respect to x is c. That means, the gradient has no relationship with X. It is a constant gradient and the descent is going to be on constant gradient. If there is an error in prediction, the changes made by back propagation is constant and not depending on the change in input This is not that good. There is another problem too. In connected layers. Each layer is activated by a linear function. That activation in turn goes into the next level as input and the second layer calculates weighted sum on that input and it in turn, fires based on another linear activation function. No matter how many layers we have, if all are linear in nature, the final activation function of last layer is nothing but just a linear function of the input of first layer. That means these two layers (or N layers) can be replaced by a single layer. No matter how we stack, the whole network is still equivalent to a single layer with linear activation (a combination of linear functions in a linear manner is still another linear function).

Tanh Function

Another activation function that is used is the tanh function.

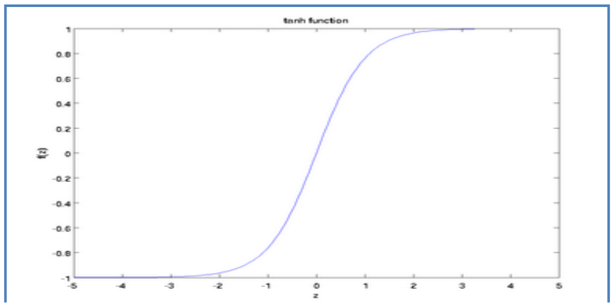

Fig.4: Tanh Function

Tanh(x) = 2sigmoid(2x)-1

It is nonlinear in nature,It is bound to range (-1, 1). The gradient is stronger for tanh than sigmoid (derivatives are steeper). Deciding between the sigmoid or tanh will depend on the requirement of gradient strength. Like sigmoid, tanh also has the vanishing gradient problem.

***Sigmoidal*:**
The sigmoid function is an activation function where it scales the values between 0 and 1 by applying a threshold.

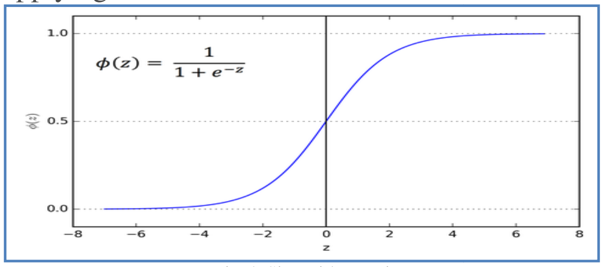

Fig.5: Sigmoid Function

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

ReLU (Rectified Linear Unit) Activation Function

A(x) = max(o,x)

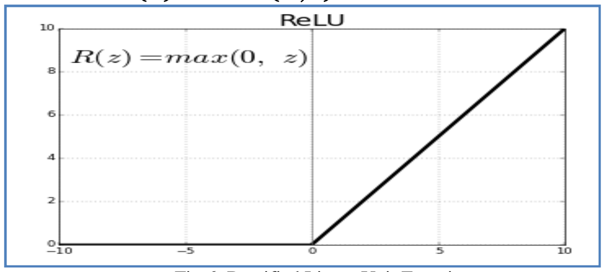

Fig.6: Rectified Linear Unit Function

The ReLu function is as shown above. It gives an output ‘x’ if x is positive and ‘0’ otherwise.

The ReLU is the most used activation function in the world right now. Since, it is used in almost all the convolutional neural networks or deep learning.

III. HOW ARTIFICIAL NEURAL NETWORK WORKS?

If O is the output vector and I is the input vector, then ANN generalised the relationship as.

$$O^a = \phi\ (I^b)$$

Where O^a is an ‘a’ dimensional input vector consisting variables, I^b is a ‘b’ dimensional output. The function ϕ is The activation function. This activation function determines the output of the network . The activation function can be a linear or a nonlinear function. However the most commonly used activation function is sigmoid function.

IV. TYPES OF ARTIFICIAL NEURAL NETWORKS:

Artificial neural networks are computational models which work similar to the functioning of a human nervous system. There are several kinds of artificial neural networks. These type of networks are implemented based on the mathematical operations and a set of parameters required to determine the output. The widely used neural networks are:

A. *Feed-forward Neural Network*

A feedforward neural network is a biologically inspired classification algorithm. It consist of a (possibly large) number of simple neuron-like processing units, organized in layers. Every unit in a layer is connected with all the units in the previous layer. These connections are not all equal: each connection may have a different strength or weight. The weights on these connections encode the knowledge of a network. Often the units in a neural network are also called nodes.

Data enters at the inputs and passes through the network, layer by layer, until it arrives at the outputs. During normal operation, that is when it acts as a classifier, there is no feedback between layers. This is why they are called feed-forward neural networks.

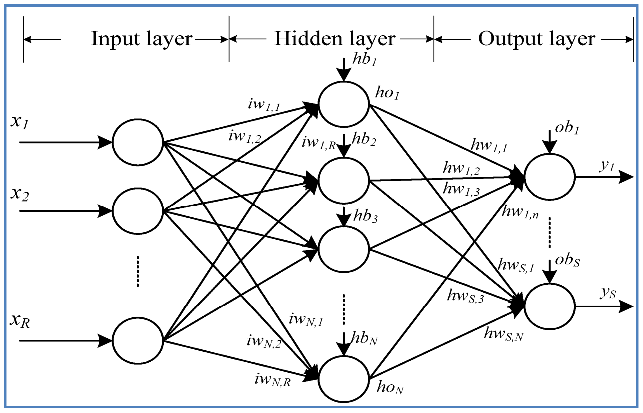
Input layer
Hidden layer
Output layer
x_1
x_2
x_R
hb_1 hb_2 hb_3 hb_N
ho_1 ho_N
$iw_{1,1}$ $iw_{1,2}$ $iw_{1,R}$ $iw_{N,1}$ $iw_{N,2}$ $iw_{N,R}$
$hw_{1,1}$ $hw_{1,2}$ $hw_{1,3}$ $hw_{1,n}$ $hw_{S,1}$ $hw_{S,3}$ $hw_{S,N}$
ob_1 ob_S
y_1 y_S

Fig.7: Feed Forward neural network

This neural network is one of the simplest form of ANN, where the data or the input travels in one direction. The data passes through the input nodes and exit on the output nodes. This neural network may or may not have the hidden layers. In simple words, it has a front propagated wave and no back propagation by using a classifying activation function usually.

B. Radial basis function Neural Network:

Radial basic functions consider the distance of a point with respect to the centre. RBF functions have two layers, first where the features are combined with the Radial Basis Function in the inner layer and then the output of these features are taken into consideration while computing the same output in the next time-step which is basically a memory. Below is a diagram which represents the distance calculating from the centre to a point in the plane similar to a radius of the circle. Here, the distance measure used in Euclidean, other distance measures can also be used. The model depends on the maximum reach or the radius of the circle in classifying the points into different categories. If the point is in or around the radius, the likelihood of the new point begin classified into that class is high. There can be a transition while changing from one region to another and this can be controlled by the beta function.

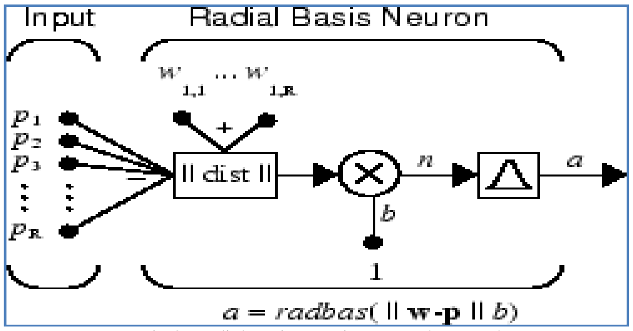

Fig.8: Radial Basis Function Neural Network

V. TRAINING OF NETWORK:

One of the most relevant features of artificial neural networks is their capability of learning from the presentation of samples (patterns), which expresses the system behavior. Hence, after the network has learned the relationship between inputs and outputs, it can generalize solutions, meaning that the network can produce an output which is close to the expected (or desired) output of any given input values. Therefore, the training process of a neural network consists of applying the required ordinated steps for tuning the synaptic weights and thresholds of its neurons, in order to generalize the solutions produced by its outputs. The set of ordinated steps used for training the network is called learning algorithm. During its execution, the network will thus be able to extract discriminant features about the system being mapped from samples acquired from the system. Usually, the complete set containing all available samples of the system behaviour is divided into two subsets, which are called training subset and test subset. The training subset, composed of 60–90 % of random samples from the complete set, will be used essentially in the learning process. On the other hand, the test subset, which is composed of 10–40 % from the complete sample set, will be used to verify if the network capabilities of generalizing solutions are within acceptable levels, thus allowing the validation of a given topology. Nonetheless, when dimensioning these subsets, statistical features of the data must also be considered. During the training process of artificial neural networks, each complete presentation of all the samples belonging to the training set, in order to adjust the synaptic weights and thresholds, will be called training epoch. We can catagorise learning processes in two distinct sorts.

A. *Supervised Learning*: The supervised learning strategy consists of having available the desired outputs for a given set of input signals; in other words, each training sample is composed of the input signals and their corresponding outputs. Henceforth, it requires a table with input/output data, also called attribute/value table, which represents the process and its behaviour. It is from this information that the neural structures will formulate "hypothesis" about the system being learned. In this case, the application of supervised learning only depends on the availability of that attribute/value table, and it behaves as if a "coach" is teaching the network what is the correct response for each sample presented for its input. The synaptic weights and thresholds of the network are continually adjusted through the application of comparative actions, executed by the learning algorithm itself, that supervise the discrepancy between the produced outputs with respect to the desired outputs, using this difference on the adjustment procedure. The network is considered "trained" when this discrepancy is within an acceptable value range, taking into account the purposes of generalizing solutions. In fact, the supervised learning is a typical case of pure inductive inference, where the free variables of the network are adjusted by knowing a priori the desired outputs for the investigated system.

B. *Unsupervised Learning*:

The application of an algorithm based on unsupervised learning does not require any knowledge of the respective desired outputs. Thus, the network needs to organize itself when there are existing particularities between the elements that compose the entire sample set, identifying subsets (or clusters) presenting similarities. The learning algorithm adjusts the synaptic weights and thresholds of the network in order to reflect these clusters within the network itself. Alternatively, the network designer can specify (a priori) the maximum quantity of these possible clusters, using his/her knowledge about the problem. Normaly the data is divided into different sets viz.

- **Training set**: A set of examples used for learning, that is to fit the parameters [i.e., weights] of the classifier.
- **Validation set**: A set of examples used to tune the parameters [i.e., architecture, not weights] of a classifier, for example to choose the number of hidden units in a neural network.
- **Test set**: A set of examples used only to assess the performance [generalization] of a fully specified classifier.

VI. NEED FOR VALIDATION

The validation data set is a set of data for the function you want to learn, which you are not directly using to train the network. You are training the network with a set of data which you call the training data set. If you are using gradient based algorithm to train the network then the error surface and the gradient at some point will completely depend on the training data set thus the training data set is being directly used to adjust the weights. To make sure you don't overfit the network you need to input the validation dataset to the network and check if the error is within some range. Because the validation set is not being using directly to adjust the weights of the network, therefore a good error for the validation and also the test set indicates that the network predicts well for the train set examples, also it is expected to perform well when new example are presented to the network which was not used in the training process

VII. ADVANTAGES OF USING ANN:

- Artificial neural networks have the ability to generalize their inputs. This ability is valuable for robotics and pattern recognition systems.
- Nonlinear systems have the capability of finding shortcuts to reach computationally expensive solutions. These systems can also infer connections between data points, rather than waiting for records in a data source to be explicitly linked. This nonlinear short-cut mechanism is fed into artificial neural networking, which makes it valuable in big-data analysis.
- Artificial neural networks have the potential for high fault tolerance..

- They do not require a prior knowledge of the physical process under consideration as a precondition.

.

VIII. LIMITATIONS OF ANN:

- It is not possible to explain how the results were calculated in any meaningful way.
- There are many parameters to be set in a neural network and optimizing the network can be challenging, especially to avoid overtraining.

IX. CONCLUSIONS

The Neural network models are developed using the input-output data sets which are acquired from the previous history of rainfall and the corresponding runoff for the catchment i.e. rainfall and runoff data only. While using these models one does not understand the underlying logic which determine the output of the model. This study gave the explained review of all the processes that govern the output of the models.

REFERENCES

1. *Bar-Yam, Y. (1997). Dynamics of Complex Systems. Addison-Wesley..*
2. *Kauffman, S. (1993) Origins of Order, Oxford University Press.*
3. *McCulloch, W. and W. Pitts (1943). A Logical Calculus of the Ideas Immanent in Nervous Activity.*
4. *Bulletin of Mathematical Biophysics, Vol. 5, pp. 115-133.*
5. *Rojas, R. (1996). Neural Networks: A Systematic Introduction. Springer, Berlin.*
6. *Rumelhart, D. and J. McClelland (1986). Parallel Distributed Processing. MIT Press, Cambridge, Mass.*
7. *Young, D. Formal Computational Skills Course Notes. Http://www.cogs.susx.ac.uk/users/davidy/fcs*