

# ARTIFICIAL NEURAL NETWORKS AND ITS APPLICATION IN COASTAL URBAN DEVELOPMENT

**DR. S. MANDAL**  
Ocean Engineering Division  
National Institute of Oceanography  
Dona Paula, Goa 403 004  
Email: smandal@darya.nio.org

## INTRODUCTION

The world population has been rapidly growing along the coastal places owing to rich resources and economical transport systems. Out of four A1 cities in India, two cities namely, Mumbai and Chennai are situated in Indian coast, hence there are practically no scopes of expanding on one side of these cities, whereas Kolkata city is about 60km from Bay of Bengal and one side is bounded by Hoogly river. There are many small cities along Indian coast. All these cities are rapidly expanding due to migration of people, increase of industrial trades and businesses, etc. There are tremendous increasing demands for proper infrastructure facilities in the cities. Planning development and modernizing the urban areas are becoming challenging problems to the city planners. Recent technological development on many aspects of urban modernization helps the city planners for future planning development and management. There are some mathematical tools and advanced remote sensing, which play important roles on urban development.

Owing to collection of good quality of remote sensing (RS) data through various satellites and widespread use of Geographic Information System (GIS), there is a demand for use of RS and GIS for proper planning development and management of modern urban infrastructure solutions in India. Along with above two tools, artificial neural networks (ANN) can play vital role for better solutions. The ANN can be applied in traffic management and growth, noise pollution, waste and effluent disposal, coastal shoreline changes, etc.

Neural networks have the ability to recognize the hidden pattern in the data and accordingly estimate the output values. Provision of model-free solutions, data error tolerance, built in dynamism and lack of any exogenous input requirement makes the neural network attractive. A neural network is an information processing system modeled on the structure of the dynamic process. It can solve the complex/nonlinear problems quickly once trained by operating on problems using an interconnected number of processing units. Its merit is the ability to deal with information whose interrelation is ambiguous or whose functional relation is not clear.

For supervised learning system, we need the actual input-output data of any particular process, which will be used to train the neural network. Once the neural network is trained, it estimates/predicts the output for unknown input data without any assumptions. In many occasions, neural network predictions are much closer to target output and better than other numerical / empirical methods.

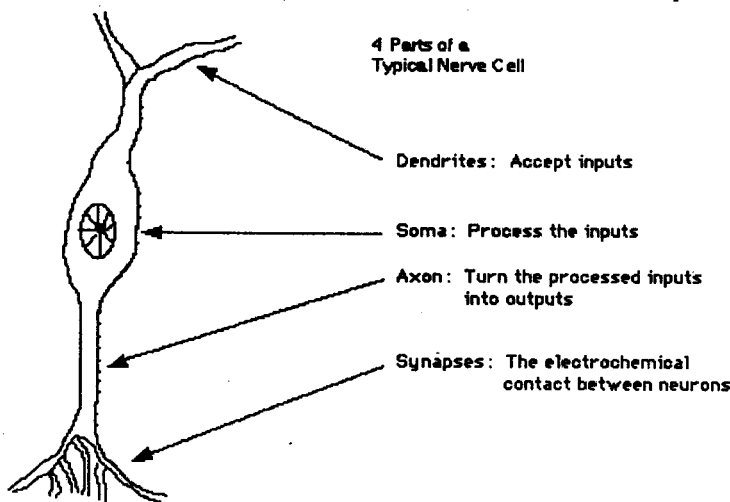
As the activities are increasing along the shore, the stability of the shoreline is becoming a great problem for the protection of valuable urban structures/land. This study can be undertaken using ANN.

This lecture will briefly highlight the basics of the ANN so that it will be easy to understand its applicability and also highlight some applications related to urban development.

## NEURAL NETWORK

The word 'Neural Network' is used to normally describe the "Artificial Neural Network"(ANN). Biological neural networks are much more complicated in their elementary structures than the mathematical models. An ANN is an information-processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this paradigm is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. This is true of ANNs as well. The easy adaptability to the solution is what distinguishes neural networks from other mathematical techniques.

**Human Neurons:** Much is still unknown about how the brain trains itself to process information, so theories abound. In the human brain, a typical neuron collects signals from others through a host of fine structures called *dendrites*. The neuron sends out spikes of electrical activity through a long, thin strand known as an *axon*, which splits into



**Fig 1 Biological neuron**

thousands of branches. At the end of each branch, a structure called a *synapse* converts the activity from the axon into electrical effects that inhibit or excite activity from the axon into electrical effects that inhibit or excite activity in the connected neurons. When a neuron receives excitatory input that is sufficiently large compared with its inhibitory input, it sends a spike of electrical activity down its axon. Learning occurs by changing the effectiveness of the synapses so that the influence of one neuron on another changes.

**Artificial Neuron:** An artificial neuron is a device with many inputs and one output. The neuron has two modes of operation; the training mode and the using mode. In the training mode, the neuron can be trained to fire (or not), for particular input patterns. In the using mode, when a taught input pattern is detected at the input, its associated output becomes the current output. If the input pattern does not belong in the taught list of input patterns, the firing rule is used to determine whether to fire or not.

Note that various inputs to the network are represented by the mathematical symbol,  $x(n)$ . Each of these inputs are multiplied by a connection weight, these weights are represented by  $w(n)$ . In the simplest case, these products are simply summed, fed through a transfer function to generate a result, and then output. Even though all artificial neural networks are constructed

from this basic building block the fundamentals may vary in these building blocks and there are differences.

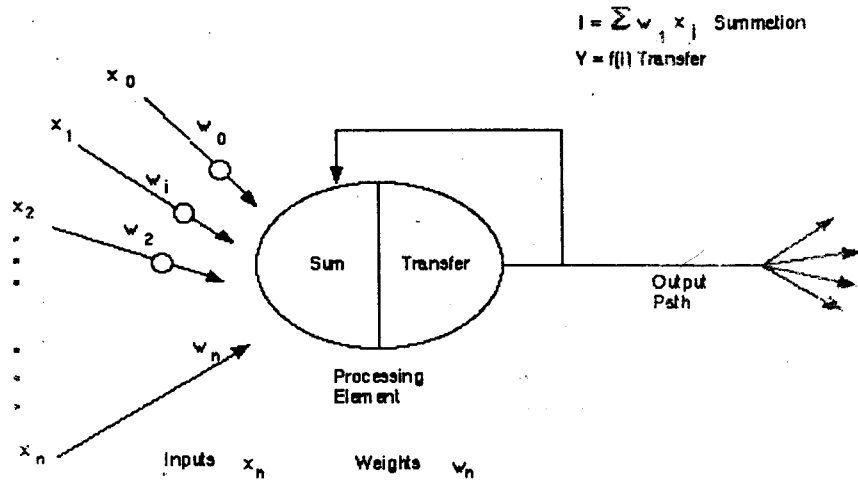


Fig 2 Artificial neuron

**Network Selection:** All artificial neural networks are based on the concept of neurons, connections and transfer functions. There is similarity between the different structures or architectures of neural networks. The majority of the variations stem from the various learning rules and how those rules modify a network's typical topology. Basically, most application of neural networks consists of the following five categories: (Beale and Jackson, 1997)

- Prediction
- Classification
- Data association
- Data Conceptualization
- Data filtering

**Feed forward neural networks:** Normally in a feed forward network there will be three layers namely the input layer, the hidden layer and the output layer. The inputs to the network are given at input layer. The number of input nodes would be equal to the number of input parameters. So the input nodes receive the data and pass them on to the hidden layer nodes. These nodes individually sum up the received values after multiplying each input value by a weight. Then they attach a bias to this sum and pass on the result through a non-linearity such as "sigmoid transfer function". This forms the input to the output layer that operates identically with the hidden layer nodes. Resulting transformed output from each output node forms the network output. The objective of the feed forward network is to minimize the global error,  $E$  given as

$$E = 1/P \sum E_p$$

and

$$E_p = 0.5 \sum (o_k - t_k)^2$$

where  $P$  = the total number of training patterns

$E_p$  = the error for  $p^{\text{th}}$  pattern.

$o_k$  = network output at  $k^{\text{th}}$  output node

$t_k$  = target output at  $k^{\text{th}}$  output node.

In the back-propagation networks the error between the target output and the network output is calculated and this will be backpropagated using various approaches. The network weights and biases are adjusted by moving a small step in the direction of negative gradient of the error function during each iteration. The iterations are repeated until a specified convergence is reached. Due to the fixed step size it converges slowly and may exhibit oscillatory behavior and hence the back-propagation networks with updated algorithms are to be used. A brief description about the working of a back propagation neural network is explained in Annexure-1.

### **SOME ANN APPLICATIONS**

Kingston et al (2000) describes an artificial neural network to model the cross-shore movement in the intensity maximum due to tides and waves. The resulting signal is then removed from the video data estimates to provide an accurate estimation of the sandbar location.

Paola and Schowengerdt (1995) have carried out a detailed comparison of backpropagation neural network and Maximum-Likelihood classifiers for urban land use classification.

Leonardi and Cirianni (2004) have applied neural networks on a study of noise pollution in urban transport. The measured noises at a particular location were analyzed for the dependence of the equivalent level of noise from the characteristics of traffic flows (quantity, composition and mean speed of the vehicular flow) and the topology of the roads, with the aim of comparing the behaviour of different neural networks, so as to supply an instrument of dynamic and flexible evaluation for the analysis of the traffic index noise, also comparing among the different network approaches the one which gives the results best fitting the observed data.

Sharma et al (2003) have studied air quality predictors based on neural networks. A method based on artificial neural network (ANN) was presented by Liu (2000) to detect the change from non-urban to urban land use. Using Levenburg-Marquart algorithm, the three-layered backpropagation network was able to identify the changes of interest with an overall accuracy of 93.10%.

May and Sivakumar (2004) have applied the artificial neural networks and multilinear regression models to predict urban storm water quality at unmonitored catchments. Violation of the assumption of data independence lead to the inclusion of insignificant variables when straightforward stepwise regression was applied. To overcome this problem, cross validation was used to determine when to stop adding variables. The artificial neural networks and regression models yielded similar prediction.

Historical census data for Baltimore MD from 1970-1990 was used to train a neural network model (Graham and Goswami, 2001). A customized graphic user interface (GUI) was developed to allow a user to create user specific data for analysis using GIS. The artificial neural network was used to make numerical projections from the extracted data. The work involved database creation and validation, scripting of an artificial intelligence model, calibration of the model from historical data, development of the GUI, making projections, and displaying the results using GIS.

Cheng et al (2003) proposed a conceptual model for defining urban growth and its complexity, in which spatial, temporal and decision-making complexity were distinguished as separate domains. They linked the conceptual model with the major current methods of modern urban modeling, such as cellular automata, neural networks, multi-agent, spatial statistics etc.

The Pearl River (Zhujiang) Delta, situated in southern China, has dramatically changed to urbanization from late 1970s onwards. Before the reforms, the Delta was one of the major lychee, mulberry and silkworm, and rice producing regions in China. Since the open door policies, the Delta has been transformed into one of the country's major economic powerhouses, with industries replacing farmland and foreign direct investment changing the function and form of urban development. Seto and Liu (2003) have compared ARTMAP neural network with maximum-likelihood for detecting this urban changes.

Yeh and Li (2002) presented a method for integrating neural networks, GIS and cellular automata that could be used in land-use planning for simulating alternative development patterns according to different planning objectives. Neural networks were used to simplify model structures and facilitate the determination of parameter values. Historic remote sensing data were used as the training data to calibrate the neural network. The original training data were assessed and modified according to planning objectives to generate alternative development patterns.

The Kohonen Self-Organising Feature Maps neural network was applied to analyse the multi-dimensional process data, and to diagnose the inter-relationship of the process variables in a real activated-sludge of municipal wastewater treatment plant. By using component planes, some detailed local relationships between the process variables, e.g., responses of the process variables under different operating conditions, as well as the global information was discovered by Hong et al (2003).

The Hopfield Neural Network model was used to estimate the urban Orientation-Destination (OD) distribution matrix from the link volumes of the transportation network, so as to promote the solving speed and precision (Gong, 1998).

Wang et al (2003) developed an improved neural network model, which combines the adaptive radial basis function network with statistical characteristics of ozone in selected specific areas, and was used to predict the daily maximum ozone concentration level. The improved method was trained and testified by hourly time series data collected at three air pollutant-monitoring stations in Hong Kong during 1999 and 2000. The simulation results demonstrated the effectiveness and the reliability of the proposed method.

Mas (2004) presented a procedure for mapping land use/cover combining the spectral information from a recent image and data about spatial distribution of land use/cover types obtained from outdated cartography and ancillary data. Two fuzzy maps, which indicate the membership of each land use/cover class, were generated from the ancillary and spectral data, respectively, using an artificial neural networks approach. The combination of both maps was obtained using fuzzy rules. In comparison with spectral classification, this procedure allowed a statistically significant increase of accuracy of land use/cover classification (from 67% to 79%).

The dispersion characteristics of vehicular exhaust emissions on urban roadways are highly non-linear and the presence of traffic-wake adds complexities to the dispersion. ANN comprising of interconnected adaptive processing units have the capability to recognize the non-linearity present in incomplete or noisy data. One-hour average artificial neural network based carbon monoxide (CO) models was developed for two air quality control regions in Delhi city – a traffic intersection and an arterial road (Nagendra and Khare, 2004). Ten meteorological and six traffic characteristic variables were used in the model. The results demonstrated that neural network models were able to explain the effects of traffic-wake on the CO dispersion in the near field regions of a roadway.

Urban air pollution is a growing problem. Large cities in particular show, at some point of time, high concentrations of substances dangerous for human health. The difficulty in forecasting pollutant's concentration trends with a reasonable error is still an open problem. An artificial neural network (Viotti et al, 2002)) was used to forecast short and middle long-term concentration levels for some of the well-known pollutants. The results seem to be in good accord with the monitored data and allow its use as the forecasting model on a 24–48 h basis requiring only the meteorological conditions and the traffic level.

Domestic water use is generally the most important component of urban water consumption. Liu et al (2003) have applied ANN to model and forecast the water demand in urban areas. Results indicated that the WDF-ANN (water demand forecast using artificial neural network) model offers an effective way to formulate domestic water demand in Weinan City in China. The model evaluation showed that the correlation coefficients were more than 90% both for the training data and the testing data.

Shan and Xu (1996) presented a knowledge-based decision support system (KB-CEDSS) that is integrated with a multilayer artificial neural network for evaluating urban development. The multilayer ANN was trained to rapidly replicate the evaluation provided by the KB-CEDSS. By integrating knowledge-based systems, decision support systems and ANNs, the system achieved improvements in the implementation of each as well as increases the scope of the application. They discussed the structure of the hybrid system, the methods for improving learning performance of backpropagation procedure using a multilayer ANN, as well as providing an example of application.

The Land Transformation Model (LTM), which couples GIS with ANNs to forecast landuse changes was presented by Pijanowski et al (2002). They presented a version of the LTM parameterized for Michigan's Grand Traverse Bay Watershed and explores how factors such as roads, highways, residential streets, rivers, Great Lakes coastlines, recreational facilities, inland lakes, agricultural density, and quality of views could influence urbanization patterns in this coastal watershed. ANNs were used to learn the patterns of development in the region and test the predictive capacity of the model, while GIS was used to develop the spatial, predictor drivers and perform spatial analysis on the results.

Predictions of tides and coastal waves are useful for coastal development. If the city is situated near the coast, these data are required for the design of coastal structures (jetties, breakwaters, etc.). These are also required for operations. Tidal prediction using back-propagation neural network was carried out by Mandal (2001, 2001a). Prediction of waves was carried out by Mandal et al (2004), and Subba Rao and Mandal (2004). Mandal and Pravaaharan (2003) have compared the wave prediction by existing numerical models and ANN models.

## POTENTIAL USES OF ANN

The neural networks can be efficiently used:

- Proper planning development and infrastructural solutions using RS, GIS and ANN.
- Future traffic intensity prediction using ANN
- Proper land-use-planning can be undertaken using ANN.
- Shoreline changes and its protection using ANN.
- Domestic effluent discharges into sea by numerical and ANN models.

## REFERENCES

1. Beale, R. and Jackson, T. (1997) *Neural Computing: An Introduction*, Bristol and Philadelphia: Institute of Physics Publishing.
2. Cheng, J; Ottens, H Masser, I and Turkstra, J (2003) Understanding urban growth, a conceptual model, *International journal of urban sciences*, university of Seoul, South Korea, Vol-7(1), 83-101.
3. Fahlman, SE (1988) An empirical study of learning speed in backpropagation networks. Technical report, CMU-CS-88-161, Carnegie-Mellon Univ, Computer Science Dept, Pittsburgh, PA.
4. Gong, Z (1998) Estimating the urban OD matrix – A neural network approach, *European Journal of Operational Research*, 106(1), 108-115.
5. Graham, T E and Goswami, I (2001) Baltimore's urban environment using GIS and neural networks, (<http://gis.esri.com/library/userconf/proc01/professional/papers/pap699/p699.htm>)
6. Hong, Y-ST, Michael, RR and Rao, B (2003) Analysis of a municipal wastewater treatment plant using a neural network-based pattern analysis, *Water research*, 37(7), 1608-18.
7. Kingston, KS; Ruessink, BG; Enckevort, IMJv and Davidson, MA (2000) Artificial neural network correction of remotely sensed sandbar location. *Marine Geology*, 169, 137-160.
8. Leonardi, G and Cirianni, F (2004) The application of a neural network on a study of noise pollution in urban transport: a case in Villa S Giovanni, *Air pollution XXII*, CA Brebbia (editor), (<http://library.witpress.com/pdfs/abstracts/AIR04/AIR04055AU.pdf>).
9. Liu, X. (2000) Change detection for urban growth modeling- an artificial neural network approach, 4<sup>th</sup> International Conference on Integrating GIS and Environmental Modeling (GIS/EM4), Alberta, Canada, September 2 - 8, 2000.
10. Liu, J, Savenjie, HHG and Xu, J (2003) Forecast of water demand in Weinan City in China using WDF-ANN model, *Physics and Chemistry of the Earth, Part A/B/C*, 28(4), 219-224.
11. Mas, JF (2004) Mapping land use/cover in a tropical coastal area using satellite sensor data, GIS and artificial neural networks, *Estuarine, Coastal and self Science*, 59(2), 219-230
12. Mandal, S (2001) Back-propagation neural networks in tidal level forecasting. Discussion. *J of Waterway, Port, Coastal and Ocean Engineering*; January'01, pp 54-59.
13. Mandal, S (2001a) Tides prediction using backpropagation neural networks, *Proc. International Conference in Ocean Engineering, ICOE, IIT Madras, 2001*, pp 499-504.
14. Mandal, S, Subba Rao and Raju, DH (2004) Neural network for estimation of ocean waves, *Proc. 3<sup>rd</sup> Indian Conference on Harbour and Ocean Engineering, INCHOE2004, NIO, Goa*

15. Mandal, S and Prabhakaran, N An overview of the numerical and neural network accosts of ocean wave prediction. Proc. International conference COPEDEC, Colombo, Sri Lanka, 15-19 September 2003, 1-9.
16. Marghany, M (2002) Hopfield neural network and quasi-linear transform model for simulation longshore current pattern from radarsat, (<http://www.gisdevelopment.net/aars/acrs/2002/gis/279.pdf>)
17. May, D and Sivakumar, M (2004) Techniques for predicting total phosphorus in urban storm waters runoff at unmonitored catchments, ANZIAM J. 45 (E), pp C296-C309.
18. Nagendra, SMS and Khare, M (2004) Artificial neural network based line source models for vehicular exhaust emission predictions of an urban roadway, Transport Research Part D: Transport and Environment, 9(3), 199-208.
19. Paola, J.D. and Schowengerdt, R.A. (1995) A Detailed Comparison of Backpropagation Neural Network and Maximum-Likelihood Classifiers for Urban Land-Use Classification", IEEE Transactions on Geoscience and Remote Sensing, vol. 33, no. 4, pp. 981-996, July'95.
20. Photis, YN and Manetos, P (2003) Simulation of urban system evolution in a synergetic modelling framework. The case of Attica, Greece, Proc. ERSA, (<http://www.ersa.org/ersaconfs/ersa03/cdrom/papers/353.pdf>).
21. Pijanowski, BC, Brown, DG, Shellito, BA and Manik, GA (2002) Using neural networks and GIS to forecast land use changes: a land transformation model, J. computers, Environment and Urban Systems, 26(6), 553-575.
22. Reidmiller, M. and Braun, H. (1993) A direct adaptive method for faster backpropagation learning: The RPROP algorithm. Proc. International Conference in neural networks, San Francisco, CA.
23. Reidmiller, M. (1994) Advanced supervised learning in multi-layer perceptrons – from backpropagation to adaptive learning algorithms, Computers Standards and Interfaces, Special issue on neural networks (5).
24. Seto, KC and Liu, W (2003) Comparing ARTMAP neural network with maximum-likelihood for detecting urban change: the effect of class resolution, Photogrammetric Engineering and Remote Sensing, Volume 69, 9, pp 981-990.
25. Shan, F and Xu, LD (1996) A hybrid knowledge-based system for urban development, Expert Systems with Applications, 10(1), 157-163.
26. Sharma, S; Barai, SV and Dikshit, AK (2003) Studies of air quality predictors based on neural networks, International Journal of Environment and Pollution (IJEP), Vol-19(5).
27. Subba Rao and Mandal, S (2004) Hindcasting of storm waves using neural network. Ocean Engineering, (press),
28. Tsai, CP, Lin, C and Shen, JN (2002) Neural network for wave forecasting among multi-stations. Ocean Engineering, 29, pp 1683-1695.
29. Viotti, P, Liuti, G and Genova, PD (2002) Atmospheric urban pollution: applications of an artificial neural network to the city of Perugia, Ecological modeling, 148(1), 27-46.
30. Wang, W, Lu, W, Wang, X and Leung, AYT (2003) Prediction of maximum daily ozone level using combined neural network and statistical characteristics, Environmental International, 29(5), 555-562.
31. Yeh, A. G-O and Li, X. (2002) Urban simulation using neural networks and cellular automata for land use planning, Proc ISPRS commission4 Symp, July 9-12, 2002. (<http://www.isprs.org/commission4/proceedings/pd/papers/036.pdf>).



### ANNEXURE-1: Back-propagation neural network

The back-propagation neural network consists of several layers of nodes analogous to neurons: an input layer (i), one or more hidden layers (h) and an output layer (o). Each node in a layer receives its input from the output of the previous layer nodes or from the network input. The connections between nodes are associated to synaptic weights (W, Z) that are iteratively adjusted during the training process. A simple example of such a network is shown in Fig-A1. An additional node with a constant output is often added to the input and hidden layers. These nodes are known as *bias nodes*.

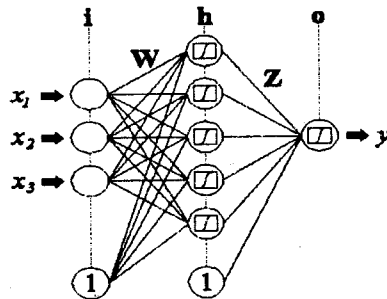


Fig-A1 Back-propagation neural network

Each hidden and output node is associated to an activation function, i.e. a differentiable function of the node total input. Several functions can be used as activation functions, but the most common choice is the sigmoid transfer function:

$$f(a) = \frac{1}{1 + e^{-a}} \quad (a1)$$

Provided that the activation function of the hidden layer nodes is non-linear, an error back-propagation neural network with an adequate number of hidden nodes is able to approximate every non-linear function.

A neural network works at its best if all its synaptic weights have been properly adjusted. The error back-propagation algorithm is a way to compute these weights and involves four steps: (1) the network is initialized by assigning random values to synaptic weights; (2) a training pattern is fed and propagated forward through the network to compute an output value for each output node; (3) computed outputs are compared with the expected outputs; (4) a backward pass through the network is performed, changing the synaptic weights on the basis of the observed output errors. Steps 2 through 4 are iterated for each pattern in a training set, then the network performance is checked (usually on the basis of a mean squared error) and a new set of training patterns is submitted to the network (i.e. a new *epoch* is started) if it needs further optimization.

In the case of neural networks with a single hidden layer, the forward propagation step is carried out as follows:

$$h_k = \sum_{j=1}^{n_i+1} i_j w_{jk} \quad (k = 1, \dots, n_h) \quad (a2)$$

where  $i_j$  are the outputs of the input layer (i.e. the network inputs and 1 for the bias node) and  $w_{jk}$  are the weights of the connections between input and hidden layers. To compute the outputs of the hidden layer, these weighted sums are passed to the activation function, but for the bias node, that is forced to have an output equal to 1:

$$h_k = f(h_k) \quad (a3)$$

$$h_{n_h+1} = 1 \quad (a4)$$

Then, the network outputs are computed in the same way:

$$o_i = \sum_{k=1}^{n_k+1} h_k z_{ki} \quad (i = 1, \dots, n_o) \quad (a5)$$

$$o_i = f(o_i) \quad (a6)$$

After the forward propagation, estimated outputs  $o_i$  are compared with expected outputs  $y_i$  and a mean quadratic error for the current pattern is computed as:

$$E = \frac{1}{n_o} \sum_{i=1}^{n_o} (y_i - o_i)^2 \quad (a7)$$

Then, in the back-propagation step, all the synaptic weights are adjusted in order to follow a gradient descent on the error surface. For the connections between hidden and output layers,  $z_{ki}$  are changed as:

$$z_{ki} = z_{ki} + \eta \delta_i^o h_k \quad (k = 1, \dots, n_k + 1; i = 1, \dots, n_o) \quad (a8)$$

where  $\eta$  is a constant (learning rate) and:

$$\delta_i^o = (y_i - o_i) \cdot f'(o_i) = (y_i - o_i) \cdot o_i(1 - o_i) \quad (a9)$$

The weights  $w_{jk}$  of the connections between hidden and input layer are also adjusted:

$$w_{jk} = w_{jk} + \eta \delta_k^h i_j \quad (k = 1, \dots, n_k + 1; j = 1, \dots, n_i + 1) \quad (a10)$$

where  $\delta_k^h$  are computed as:

$$\delta_k^h = f'(h_k) \cdot \sum_{i=1}^{n_o} \delta_i^o z_{ki} = h_k(1 - h_k) \cdot \sum_{i=1}^{n_o} \delta_i^o z_{ki} \quad (a11)$$

The network training is iterated until a given condition is met. Minimization of the quadratic error is usually involved, but other criteria can also be used. Here, the weight adjustment process does not provide a unique optimized result, since many non-deterministic factors (e.g. different starting values of the synaptic weights) can affect the network training.

## ANNEXURE-2: To solve the xor problem using back-propagation neural network

The exclusive-or (xor) is defined as follows

| x | y | z |
|---|---|---|
| 1 | 0 | 1 |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 1 | 1 |

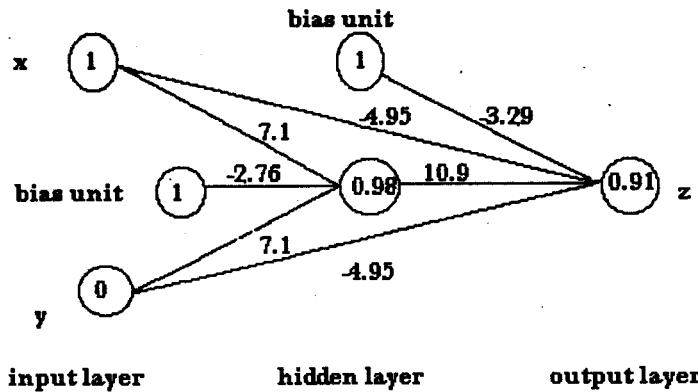


Fig-A2 A three layer network to solve xor problem with weights produced by back- propagation.

Here the circles represent *neurons or nodes or units* and the numbers within the circles represent the activation values of the nodes. The main nodes are arranged in layers. There are three layers: the *input layer* that contains the values for x and y, a *hidden layer* that contains one node and an *output layer* that gives the output value, z. The hidden layer is so-named because the network can be regarded as a black box with inputs and outputs that can be seen but the hidden units cannot be seen. There are two other units present called *bias units* whose values are always 1.0. The lines connecting the circles represent *weights* and the number beside a weight is the value of the weight. Much of the time back-propagation networks only have connections within adjacent layers however this one has two extra connections that go directly from the input units to the output unit. In some problems, like xor these extra input-output connections make training the network much faster. Networks are usually described by the number of units in each layer so the network in Figure-A2 can be described as a 2-1-1 network with extra input-output connections.

To compute the value of the output unit, z, we place values for x and y on the input layer units. Let these values be 1.0 and 0.0 as in Figure-A2. First we compute the value of the hidden layer unit, h. The first step of this computation is to look at each lower level unit and the bias unit that is connected to the hidden unit. For each of these connections, find the value of the unit and multiply by the weight and sum all the results. The calculations are:

$$\begin{aligned}
 1.0 * 7.1 &= 7.10 \\
 1.0 * -2.76 &= -2.76 \\
 0.0 * 7.1 &= 0.0 \\
 \text{sum} &= 4.34
 \end{aligned}$$

In some neural networks we might just leave the activation value of the unit to be 4.34. In this case we would say that we are using the linear activation function. In backpropagation network to obtain best fit, this value is passed to certain types of non-linear functions. The commonly used non-linear transfer function is:

$$v = 1/(1 + e^{-s})$$

where, s is the sum of the inputs to the neuron and v is the value of the neuron. Thus, with s = 4.34, v = 0.987. This particular function is called the 'standard sigmoid'. The general function used to compute the value of a neuron can be called the *activation function*, *squashing function* or *transfer function*. The calculations for the output unit, z are:

$$\begin{aligned}
 1.000 * -4.95 &= -4.95 \\
 0.000 * -4.95 &= 0.00 \\
 0.987 * 10.9 &= 10.76 \\
 1.000 * -3.29 &= -3.29
 \end{aligned}$$

$$\text{sum} = 2.52$$

$$v = 0.91$$

The output, 0.91 is not quite 1 but for this example it is close enough. When using this particular activation function for a problem where the output is supposed to be a 0 or 1 getting the output to within 0.1 of the target value is a very common standard but other people may want to get to closer than this. With this particular activation function it is actually somewhat hard to get very close to 1 or 0 because the function only approaches 1 and 0 as the input to the function approaches  $\infty$  and  $-\infty$ . The values computed for the xor function are:

| x | y | Z (actual) |
|---|---|------------|
| 1 | 0 | 0.91 (1)   |
| 0 | 0 | 0.08 (0)   |
| 0 | 1 | 1.00 (1)   |
| 1 | 1 | 0.10 (0)   |

\*\*\*\*\*