# ADAPTIVE NETWORKS FOR FAULT DIAGNOSIS AND PROCESS CONTROL

**B.V.Babu*** **& M. Shailesh**
Department of Chemical Engineering
Birla Institute of Technology and Science
Pilani-333031 (Rajasthan) India

**Abstract**

The use of adaptive (neural) networks for fault diagnosis and process control is explored. Adaptive networks can be used as fault recognition systems, adaptive non-linear process models, and as controllers. Connection strengths representing correlation between inputs (alarms and sensor measurements) and outputs (faults) are made to learn by the network using the Back Propagation Algorithm (BPA). Results are presented for diagnosing faults in a Heat Exchanger – CSTR system. The network employed in the present study has eight input nodes corresponding to the state variables and eight output nodes corresponding to the eight malfunctions (faults). A hidden layer of an optimum number of eleven nodes is chosen based first on heuristics and later on various trial and error combinations. A learning rate of 0.7 is used. The training used the lower and upper limits as keys for training as well as the limits for normalization. The final fault diagnosis (after network training) involved rounding of the values from the network. The network is able to identify all the faults correctly.

*Key Words: Adaptive Networks, Fault Diagnosis, Artificial Neural Networks, Back Propagation Algorithm, Heat Exchanger, CSTR*

## 1. Introduction

Changes in the physical conditions of process units, control systems or exogenous conditions may lead to what are generally referred to as faults. Faults in the broadest sense include symptoms resulting from physical changes, such as deviations of temperature or pressure from their normal operating range, as well as physical changes themselves such as scaling, foaming, leaks and wear. Even changes in unmeasured process parameters such as heat or mass transfer coefficients can be deemed to be faults. A wide variety of techniques have been proposed to detect and diagnose faults using redundant instrumentation, Knowledge Based Expert Systems (KBES), process modeling statistical tools, digraphs and combinations of these (Watanabe et al., 1994). The difficulty with these techniques is that they involve process modeling for fault diagnosis itself can be quite a difficult job because errors in the model can be interpreted as faults thus yielding false alarms, or can prevent faults from being detected when they occur (Ungar and Powell, 1988).

What is of special interest is incipient fault diagnosis that is detection and diagnosis at the very beginning stages of the occurrence of the fault. The present study focuses on the use of Neural Networks to do incipient fault diagnosis for unsteady state processes in which faults occur. A neural network can autonomously store knowledge by learning from historical fault information and has the characteristic of associative memory. Information about faults can be learnt by training the network on a set of data such as the values of steady state process variables for normal conditions and those for identified fault conditions. If data cannot be found in the past, daily reports of process maintenance and data logs, the necessary data can be collected from a program designed to identify faults. The neural network in identifying system inefficiency is directly related to the comprehensiveness of training data. The network acts as a black box into which we send inputs and from which we get outputs. Fault Diagnosis exploits the parallelism of the network. The system chosen consists of a Heat Exchanger and CSTR. The system is analyzed and a single fault case is tackled (Ungar and Powell, 1988).

Most of the studies reported in literature on fault diagnosis using neural networks involve chemical systems with a CSTR as one of its components (Venkatasubramanian and Chan, 1989; Watanabe et al., 1989; Ungar et al., 1990; Venkatasubramanian et al., 1990; Sorsa et al., 1991; Watanabe et al., 1994). Farell and Roat (1994) proposed a framework for enhancing fault diagnosis capabilities of Artificial Neural Networks.

* Corresponding Author: Group Leader, ET Group, ESD (Workshop), BITS, Pilani; E-mail: bvbabu@bits-pilani.ac.in; Fax: (01596) 44183; Phones: (01596) 45073 Ext. 205(Off.); (01596) 42212/44977 (Res.).

## 2. Artificial Neural Networks (ANN)

Neural computing is largely motivated by the possibility of creating an artificial computing network similar to the brain and nerve cells in our body. Recent advances in neuro science and in computers have sparked a renewed interest in neural network model for problem solving. These networks are computing systems composed of a number of highly interconnected layers of simple neuron like processing elements. Computations are collectively performed by the entire network with the knowledge represented as distributed patterns of activity all over processing elements. The collective activities result in a higher degree of parallelism, which enables the network to solve complex problems. The distributed representation leads to greater fault tolerance and to graceful degradation when problems are encountered beyond the range of the network. In addition, these networks can adjust dynamically to environmental changes, infer general rules from specific examples and recognize invariences from complex high dimensional data (Lippmann, 1987).

Fig. 1 depicts a simple structure of feed forward network architecture. The circles represent neurons arranged in three layers: input, hidden and output. Each hidden input is connected to each input and output unit. Each hidden and output unit is also connected to a bias. The bias provides a threshold for the activation of the neuron and is essential in order to classify network input patterns into various subspaces. Each connection has a weight associated with it. For input layer, the input value is forwarded straight to the next hidden layer. The hidden and output layer carry out two calculations: first a weighted sum of the inputs and then the output is calculated using a non-decreasing and differential transfer function. Usually a sigmoidal function is used (Lippmann, 1987).

ANNs come in a variety of different architectures, the most popular being the feed forward network trained by back-propagation (Rumelhart and McClelland, 1986). Back-propagation is an example of a mapping network that learns an approximation to a function, y equal to f(x), from sample x, y pairs. The fact that the function to be learned is nonlinear presents no problem to a back-propagation net (BPN). Representative applications of back-propagation include speech synthesis and recognition, visual pattern recognition, analysis of sonar signals, defense applications, medical diagnosis, and learning in control systems. BPNs have been applied to pattern classification problems in a number of fields, which include classification of sonar targets (Gorman and Sejnowski, 1988), speech recognition (Lippmann, 1989) and sensor interpretation (McAvoy et al., 1989). Application of BPNs to failure state recognition in chemical plants has been studied by Hoskins and Himmelblau (1988), Ferrada et al. (1989), Hoskins et al. (1988), Ungar et al. (1990), Venkatasubramanian and Chan (1989), Venkatasubramanian et al (1990), and Watanabe et al. (1989). These studies have shown that BPN classifiers have the ability to learn the classifications of a set of training examples, and can often successfully generalize this knowledge to classify new cases of known failure types. Other applications of ANN include dynamic modeling and control of chemical process systems (Bhat and McAvoy, 1989), diagnosis using back propagation neural networks (Kramer and Leonard, 1990), artificial neural network model for the equilibria prediction in ternary mixture extraction (Babu and Karthik, 1997), artificial neural networks for the estimation of heat transfer parameters in trickle bed reactors using radial basis functional networks (Babu and Sangeetha, 1998).

## 3. Network Training

Any network learns by making changes in the weights of the connections in accordance with the learning rule. There are a number of algorithms available for the above purpose but the most widely used is the back propagation algorithm (Masters, 1990), the pseudo code of which is listed below:

- Initialize the weights and offsets. Set all of them to low random values.
- Present inputs and desired outputs. This is done by presenting a continuous valued input vector and specifying the desired outputs. If the net is used as a classifier all desired outputs are set to 1. The input could be new on each turn or one could use a cyclic pattern to train.
- Calculate the actual outputs using the sigmoidal non-linearity.
- Adapt weights using a recursive algorithm starting at the output nodes and working back. Adjust the weights by the formula.

$$W_{ij}(t+1) = W_{ij}(t) + \eta\, \delta_j\, x_t'$$

Where,

$W_{ij}$ = weight from node $i$ to node $j$ at time $t$.
$\eta$ = gain term
$\delta_j$ = error term for node $j$.

If node $j$ is an output node, then

$$\delta_j = y_j(1 - y_j)(d_j - y_j)$$

where, $d_j$ = desired output of the node $j$
and $y_j$ = actual output

If node $j$ is an internal hidden node then

$\delta_j = x_j' (1 - x_j') \sum \delta_k w_{jk}$)
where,

$k$ = over all nodes in layers above node $j$

If a momentum term, $\alpha$ is added the network sometimes becomes faster and weight changes are smoothed by

$W_{ij}(t+1) = w_{ik}(t) + \eta \delta_j x_j' + \alpha[w_{ij}(t)-w_{ij}(t-1)]$

Where $0<\eta<1$.

- Repeat step 2.
- Stop.

## 4. Case Study

In the present study, fault diagnosis has been carried out using BPN of ANN for a system consisting of a heat exchanger and a CSTR (see Fig. 2). The problem definition is as follows:

483 lb/hr of a 42-API kerosene leaves the bottom of a distillation column at 390°F. It will be cooled to 200°F by 1490 lb/hr of a 34-API mild content crude oil coming from storage at 100°F and heated to 170°F. A 10-psi pressure drop is permissible. The specifications of the heat exchanger are already fixed. The outlet of this heat exchanger is the feed to a CSTR. The CSTR has an exothermic reaction taking place in it. The reaction is given by A $\rightarrow$ B. The concentration of the desired component in the feed is 0.2 lb/ft. The temperature and flow rate of the feed are also fixed.

The assumptions made are, (1) there is an unlimited quantity of hot fluid and the type of heat exchanger is fully specified, and (2) the CSTR has unlimited supply of coolant.

The steady-state equation for the CSTR is given below, which is used for the calculation of steady-state values:

$Q = UA_t(T-T_j)$, $A_t$ = heat transfer area

Energy balance equation:

$-UA_t(T-T_j) + (-\Delta H_A)V_b R_A = (\sum M_j C_j)(dT/dt)$

However, some of the inputs for this equation come from the steady-state equation for a heat exchanger, which is given by:

$Q = F_{cold}C_p(T_{coldout} - T_{coldin}) = F_{hot}C_p(T_{hotin} - T_{hotout})$

The steady-state unknown temperatures are calculated using the above equation.

## 5. Selected malfunctions for Case study

The possible malfunctions (refer Fig. 2) in the heat exchanger are changes in $T_{hotin}$, $T_{hotout}$, $T_{coldin}$, $F_{cold}$, *and* $F_{hot}$. These are global faults because a change in any of these values from the steady-state values will result in the change in the values of $C_{out}$, $F_{out}$, *and* $T_{out}$. In CSTR, the faults can be a change in $F_{in}$, $T_{in}$, $C_{in}$, $T_j$ (temperature of the coolant), and $F_j$ (flowrate of the coolant). However, the change in $F_{in}$ and $T_{in}$ are taken care of by the CSTR because these correspond to $F_{coldout}$ and $C_{coldout}$ of CSTR. Therefore, these do not come under the fault list for the system. The following table (Table-1) gives the list of possible single fault scenarios:

Table-1: List of Faults.

| S.No. | Fault (Malfunction) | Designation |
|---|---|---|
| 1. | Change in $T_{hotin}$ | F1 |
| 2. | Change in $T_{hotout}$ | F2 |
| 3. | Change in $T_{coldin}$ | F3 |
| 4. | Change in $F_{cold}$ | F4 |
| 5. | Change in $F_{hot}$ | F5 |
| 6. | Change in $C_{in}$ | F6 |
| 7. | Change in $T_j$ | F7 |
| 8. | Change in $F_j$ | F8 |

Various faults were simulated and new steady-state measurement patterns were used to train the network. It is not possible to know where a fault has occurred by measuring only the outlet concentration or temperature or flowrate. Here comes the need for fault diagnosis.

## 6. Network Architecture for Case study

The network employed in this case has eight input nodes corresponding to the state variables and eight output nodes corresponding to the eight malfunctions. A hidden layer of eleven nodes was chosen based first on thumb rules and later on various combinations. A learning rate of 0.7 was used. The training scheme is as follows:

The faults in the heat exchanger-CSTR system correspond to fluctuations in one of the eight input variables. Deviations beyond 5% of the normal values are assumed to result in malfunctions. The BPN algorithm was used to train the network. A normal matrix containing all the normal values was initialized first. Based on each value a lower limit and upper limit were used. The training used these lower limit and upper limit values as keys for training as well as the limits for normalization. The network was trained until the error was less than 0.1. The final fault diagnosis (after network training) involved rounding-off of the values from the network.

## 7. Results and Discussion

The neural network was trained using BPN algorithm. The network was able to identify all the faults correctly. The learning rate was fixed at 0.7. The mapping of the weights of the neurons from input layer to hidden layer and hidden layer to output layer are shown in Table-1 and Table-2 respectively. The network used was a single layered perceptron (one hidden layer) with eight input nodes and eight output nodes. The number of hidden nodes was first fixed using a thumb rule and thereafter the number of nodes in the hidden layer was experimented with. It was found that the network took least time to train when trained with an 11 node hidden layer. Fig. 4 shows this clearly. The number of iterations that the network took for training is a reflection of the time it took for training.

The learning rate is also experimented with and an optimum value of 0.7 was arrived at. It was found that the network trained best with a combination of learning rate of 0.7 and 11 nodes of hidden layer as can be seen from Figs. 3 & 5).

The present study focused on only single fault and multiple-fault (combination of two- or three- faults simultaneously) scenario is considered. The network training becomes very difficult even for the two-fault case. The reason for this is that there can be 36 different cases. This means that there will be 44 (36+8) total cases to take care of. This would make training a very difficult task and that network itself would be very complicated. A slightly better advancement on the program currently written is a single fault case where the ranges for fault values are fixed. That way we can actually tell what kind of action can be taken while controlling this fault. This sort of demarcation helps particularly when we use fuzzy logic or neural networks itself to control the process. Fuzzy logic needs this sort of a demarcation to take an action based on the fault because it makes it easier to make a rule table.

## 8. Conclusions

- Artificial Neural Networks with Back Propagation Algorithm has been successfully applied for fault diagnosis with 8 faults in a Chemical Engineering system consisting of a heat exchanger and CSTR.
- A single layered perceptron (one hidden layer) with eight input nodes, eight output nodes, and 11 nodes in single hidden layer with a learning rate value of 0.7 gave very good results in the present study for fault detection.

## 9. References


**Babu, B.V. & Karthik, R.** (1997). Artificial Neural Network Model for the Equilibria Prediction in Ternary Mixture Extraction. *Proceedings of IIChE Golden Jubilee Congress (CHEMCON-97)*, IIT, Delhi, December 15-18.

**Babu, B.V. & Sangeetha, S.** (1998). Artificial Neural Networks for the estimation of Heat Transfer Parameters in Trickle Bed Reactors using Radial Basis Functional Networks. *Proceedings of 14th National Conference of Environmental Engineers*, Institution of Engineers (India), A.P.Centre, Hyderabad, August 7-9, pp9-15, 1998.

**Bhat, N. & McAvoy, T.** (1989). Use of Neural Nets for Dynamic Modeling and Control of Chemical Process Systems. *Proceedings of 1989 American Automation Control Conference.,* Pittsburgh, PA, pp. 1342-1348, June, 1989.

**Farell, A.E. & Roat, S.D.** (1994). Framework for enhancing Fault Diagnosis capabilities of Artificial Neural Networks. *Computers and Chemical Engineering, 18 (7),* 613-635.

**Ferrada, J.J., Gordon, M.D., & Osborne-Lee, I.W.** (1989). Application of Neural Networks for Fault Diagnosis in Plant Production. *AIChE National Meeting,* San Francisco.

**Gorman, R.P. & Sejnowski, T.J.** (1988). Analysis of Hidden Units in a Layered Network trained to classify Sonar Targets. *Neural Networks, 1,* 75-89.

**Hoskins, J.C. & Himmelblau, D.M.** (1988). Artificial Neural Network Models of knowledge representation in Chemical Engineering. *Computers and Chemical Engineering, 12,* 881-890.

**Hoskins, J.C., Kaliyur, K.M., & Himmelblau, D.M.** (1988). The application of Artificial Neural Networks to Fault Diagnosis in Chemical Processing. Paper 37d, *AIChE Spring Meeting,* Houston .

**Kramer, M.A., & Leonard.** (1990). Diagnosis using Back propagation Neural Networks – Analysis and Criticism. *Computers and Chemical Engineering, 14 (12),* 1323-1338.

**Lippmann, R.P.** (1987). An Introduction to Computing with Neural Nets. *IEEE ASSP Magazine, April 1987,* 4 - 21.

**Lippmann, R.P.** (1989). Review of Neural Networks for Speech Recognition. *Neural Computation, 1,* 1 - 38.

**Masters, T.** (1990). *Advanced Algorithms for Neural Networks.* McGraw Hill Publishing Company, New York.

**McAvoy, T.J., Wang, N.S., Naidu, S., Bhat, N., Gunter, J. & Simmons, M.** (1989). Interpreting Biosensor Data via Back-propagation. *Proceedings of International Joint Conference on Neural Networks, Washington, D.C,* I-227-233.

**Rumelhart, D.E. & McClelland.** (1986). *Parallel Distributed Processing.* M.I.T. Press, Cambridge, Massachusetts.

**Sorsa, T., Koivo, H.N. & Koivisto, H.** (1991). Neural Networks in Process Fault Diagnosis. *IEEE Transactions on Systems, Man, and Cybernatics, 21 (4),* July/August, 1991.

**Ungar, L.H. & Powell, B.A.** (1988). Fault Diagnosis Using Non-linear Adaptive Networks. *AIChE National Meeting, November,* 1988.

**Ungar, L.H., Powell, B.A. & Kamens, S.N.** (1990). Adaptive Networks for Fault Diagnosis and Process Control. *Computers and Chemical Engineering, 14 (4/5),* 561-572.

**Venkatasubramanian, V. & Chan, K.** (1989). A Neural Network Methodology for Process Fault Diagnosis. *American Institute of Chemical Engineers Journal. 35 (12),* 1993-2002.

**Venkatasubramanian, V., Vaidyanathan, R. & Yamamoto, Y.** (1990). Process Fault Detection and Diagnosis using Neural Networks – 1. Steady-state Processes. *Computers and Chemical Engineering.14 (7),* 699-712.

**Watanabe, K., Matsuura, I., Abe, M., Kubota, M. & Himmelblau, D.M.** (1989). Incipient Fault Diagnosis of Chemical Processes via Artificial Neural Networks. *American Institute of Chemical Engineers Journal. 35 (11),* 1803-1812.

**Watanabe, K., Hirota, S., Hou, L. & Himmelblau, D.M.** (1994). Diagnosis of Multiple Simultaneous Fault via Hierarchical Artificial Neural Networks. *American Institute of Chemical Engineers Journal. 40 (5),* 893-848.

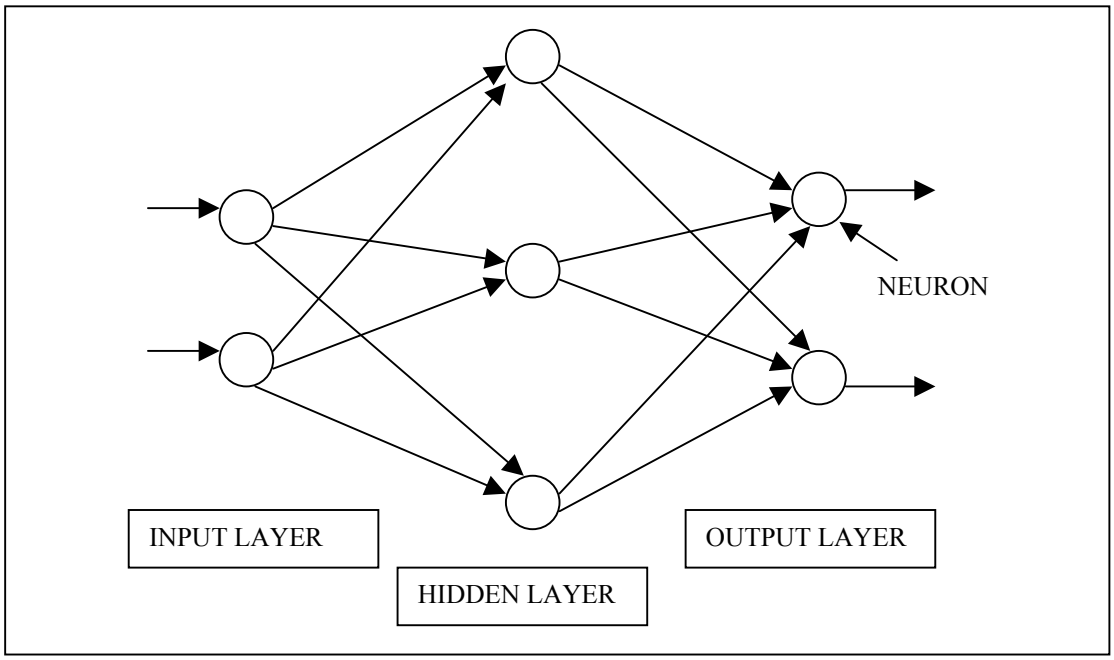


Fig. 1: Structure of a Simple Artificial Neural Network.

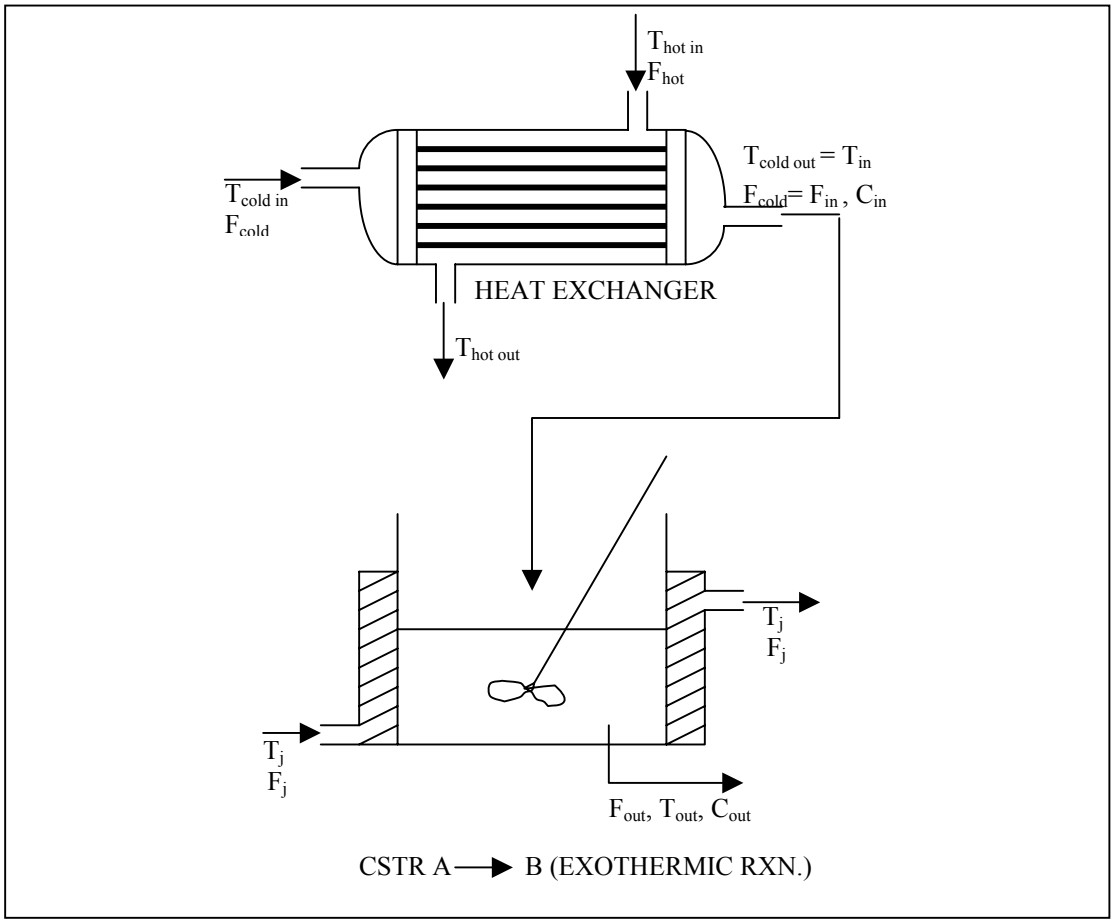


Fig. 2: A Heat Exchanger - CSTR System.

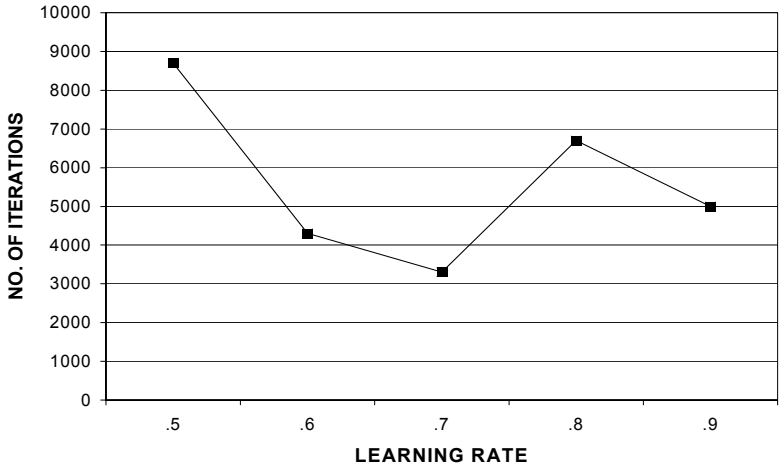


Fig. 3: No. of iterations vs. Learning rate
(No. of hidden layers=1 of 10 nodes)

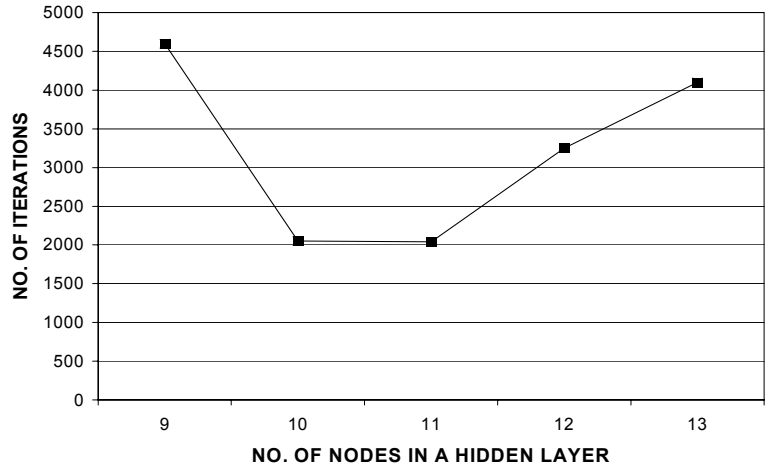


Fig. 4: No. of iterations vs. No. of Nodes in hidden layer

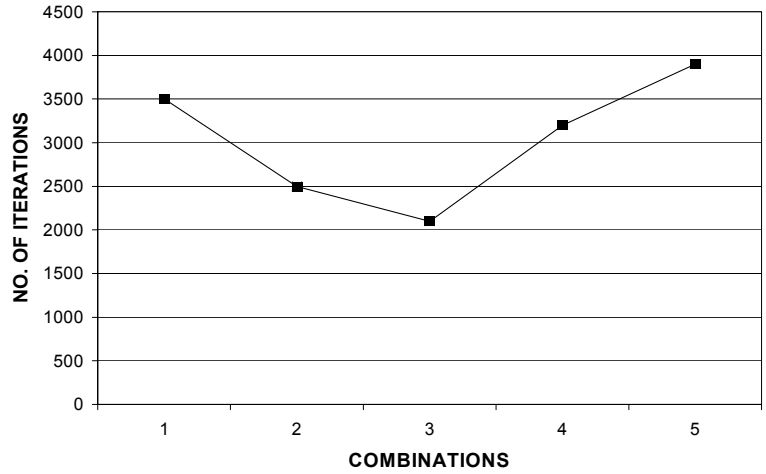


Fig. 5.: No. of iterations vs. Combinations.

Table-1. Mapping of the Weights of the neurons from input layer to hidden layer

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | .442 | .645 | .458 | .268 | .900 | .200 | .053 | .471 | .259 | .306 | .419 |
| 2 | .966 | .493 | .437 | .300 | .195 | .419 | .780 | .232 | .430 | .834 | .202 |
| 3 | .672 | .489 | .047 | .452 | .769 | .214 | .087 | .166 | .676 | .435 | .870 |
| 4 | .142 | .789 | .891 | .568 | .774 | .645 | .470 | .934 | .575 | .963 | .596 |
| 5 | .499 | .115 | .631 | .298 | .218 | .596 | .094 | .311 | .094 | .573 | .887 |
| 6 | .401 | .719 | .246 | .388 | .937 | .757 | .750 | .665 | .830 | .205 | .213 |
| 7 | .982 | .781 | .035 | .849 | .499 | .003 | .631 | .828 | .046 | .337 | .379 |
| 8 | .275 | .762 | .094 | .230 | .129 | .642 | .443 | .377 | .822 | .055 | .385 |

Table-2. Mapping of the Weights of the neurons from hidden layer to output layer

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| 1 | .2134 | .4541 | .1121 | .982 | .3351 | .4312 | .8051 | .4556 |
| 2 | .2312 | .4873 | .4462 | .3916 | .3221 | .2314 | .5024 | .6438 |
| 3 | .4706 | .0865 | .7982 | .1484 | .4699 | .837 | .1633 | .2619 |
| 4 | .0114 | .7320 | .3212 | .0493 | .4564 | .4124 | .6856 | .6817 |
| 5 | .0021 | .9820 | .7661 | .4849 | .2965 | .4299 | .1337 | .4356 |
| 6 | .0136 | .0464 | .3113 | .3147 | .2187 | .34161 | .8852 | .4671 |
| 7 | .021 | .0923 | .4996 | .2146 | .8031 | .9373 | .4856 | .0647 |
| 8 | .479 | .5621 | .0485 | .9802 | .4972 | .5591 | .3118 | .4719 |
| 9 | .823 | .7461 | .1487 | .1456 | .6913 | .4896 | .4561 | .4123 |
| 10 | .232 | .4422 | .0890 | .4280 | .5390 | .1657 | .6524 | .3918 |
| 11 | .465 | .3926 | .1812 | .3931 | .3216 | .3859 | .0400 | .6127 |