Computers chem. Engng, Vol. 14, No. 7, pp. 699–712, 1990
Printed in Great Britain.

0098-1354/90 $3.00 + 0.00

PROCESS FAULT DETECTION AND DIAGNOSIS USING NEURAL NETWORKS—I. STEADY-STATE PROCESSES

V. VENKATASUBRAMANIAN,[1]† R. VAIDYANATHAN[1] and Y. YAMAMOTO[2]

[1]Laboratory for Intelligent Process Systems, School of Chemical Engineering, Purdue University, West Lafayette, IN 47907, U.S.A.

[2]CIM Project, Corporate Planning Department, The Furukawa Electric Co. Ltd, 2–6–1 Chiyoda, Tokyo, 100, Japan

(*Received* 6 *June* 1989; *final revision received* 5 *February* 1990; *received for publication* 15 *February* 1990)

Abstract—An analysis of the learning, recall and generalization characteristics of neural networks for detecting and diagnosing process failures in steady state processes is presented. The single fault assumption has been relaxed to include multiple causal origins of the symptoms. The effect of incomplete and uncertain process symptom data such as sensor faults, and the effect of degradation of different hidden units, on the performance of the network, have been analyzed. Various neural network topologies (i.e. number of hidden units and hidden layers) have been tested and compared. The results show that accurate recall and generalization behavior is observed during the diagnosis of single faults. Performance during recall improves at first with an increase in the number of hidden units and with the amount of training, and then attains convergence. In general, performance during generalization improves with the extent of training. The networks are also able to diagnose correctly even in the presence of faulty operation of certain sensors. Networks trained on single faults are able to accurately diagnose measurement patterns resulting from multiple faults in a large majority of the cases studied. Graceful degradation of diagnostic function was observed in many of the multiple-fault cases that were not accurately diagnosed.

1. INTRODUCTION

Considerable attention has been devoted in recent years to the problem of fault detection and diagnosis in chemical plants. Failures of process equipment and instrumentation increase the operating costs and result in loss of production. Undetected or uncorrected malfunctions can induce failures in related equipments, and in extreme cases lead to catastrophic accidents. Modern chemical plants are highly complex. This coupled with the need for quick diagnosis rule out exclusive reliance on human operators for diagnosis. Hence a number of knowledge-based expert system (KBES) approaches have been proposed in the literature for automated fault diagnosis. Examples of recent work include the model-based reasoning methodology of Rich and Venkatasubramanian (1987) and Venkatasubramanian and Rich (1988) and the functional decomposition approaches of Shum *et al.* (1988) and Finch and Kramer (1988).

However, rapid deployment of these systems has been difficult to achieve due to certain inherent limitations associated with current KBES. These limitations include the tedious nature of knowledge acquisition, the inability of the sytem to learn or dynamically improve its performance and the unpredictability of the system outside its domain of expertise. A potential solution to these problems is the use of neural networks as demonstrated in the recent papers of Venkatasubramanian (1985), Hoskins and Himmelblau (1988) and Venkatasubramanian and Chan (1989).

In this study, we present a more detailed and thorough analysis of the learning, recall and generalization characteristics of neural networks for detecting and diagnosing process failures in steady state processes. The case studies investigated in this paper are also more complex than the ones reported in the earlier works. In the present study, the single fault assumption has been relaxed to include multiple causal origins of the symptoms. We also investigated the effect of incomplete and uncertain process symptom data such as sensor faults. Various neural network topologies (i.e. number of hidden units and hidden layers) have been tested and compared. In addition, the effect of degradation of different hidden units on the performance of the network has been analyzed.

2. NEURAL NETWORKS AND THE BACKPROPAGATION ALGORITHM: AN INTRODUCTION

Recent advances in neuroscience and in computer science have sparked renewed interest in neural network models for problem solving. Neural networks are computing systems composed of a number of highly interconnected layers of simple neuron-like processing elements, which process information by their dynamic response to external inputs (Kohonen, 1984; Rumelhart and McClelland, 1986). Computations are collectively performed by the entire network with knowledge represented as distributed patterns of

†Author to whom all correspondence should be addressed.

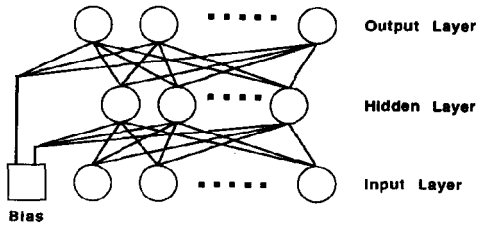


Fig. 1. Multilayer feedforward neural network.

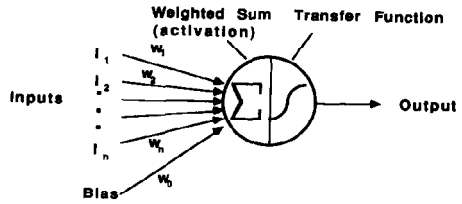


Fig. 2. Processing element (neuron).

activity over all processing elements. The collective operations result in a high degree of parallelism which enables the network to solve complex problems rapidly. The distributed representation leads to greater fault tolerance and to graceful degradation when problems are encountered beyond the range of experience of the network. In addition, neural networks can adjust dynamically to environmental changes, infer general rules from specific examples and recognize invariances from complex high-dimensional data.

Figure 1 depicts the multilayer feedforward neural network architecture used in this study. The circles represent neurons arranged in three layers: input, hidden and output. Each hidden unit is connected to each input and output unit. Each hidden and output unit is also connected to a bias whose value is always 1.0. The bias provides a threshold for the activation of the neuron and is *essential* in order to classify network input patterns into various subspaces (Hanson and Burr, 1987). Each connection has a weight associated with it. For the input layer, an input value is straightly forwarded to the next layer. The hidden and output units carry out two calculations, first a weighted sum of the inputs is taken and then the output is calculated using a non-decreasing and differentiable transfer function as shown in Fig. 2. Usually a sigmoid function $f(z)$ is used:

$$f(z) = \frac{1}{1 + e^{-z}}. \tag{1}$$

The form of this function is shown in Fig. 3.

The network of Fig. 1 learns by making changes in the weights of the connections in accordance with a learning rule. The back-propagation algorithm (Rumelhart *et al.*, 1986; Werbos, 1974) used in the present study is the most investigated supervised learning algorithm. This algorithm has been used in a lot of applications: speech recognition (Sejnowski and Rosenberg, 1987), handwriting classification (Denker *et al.*, 1989), radar signal classification (Ahalt *et al.*, 1989), game playing programs (Tesauro and Sejnowski, 1989), prediction (Lapedes and Farber, 1987; Qian and Sejnowski, 1988) and so on. The algorithm is a generalized version of the steepest descent learning algorithm developed by Widrow and Hoff (1960). The former algorithm adopts a learning rule called the *Generalized Delta Rule* (GDR) named after the original delta rule presented by Widrow and Hoff. GDR follows an iterative gradient descent algorithm designed to minimize the mean square error E:

$$E = \sum_{m=1}^{M} \sum_{i=1}^{N} (t_i^{(m)} - y_i^{(m)})^2, \tag{2}$$

where M and N denote the number of training patterns presented to the input layer and the number of units in the output layer, respectively, and $t_i^{(m)}$ represents the desired value of the ith output element given the mth pattern, while $y_i^{(m)}$ is the actual output of the same element.

Given the mth pattern, the weight updating in a supervised learning algorithm follows a general formulation:

$$w_{ji}^{(m)} = w_{ji}^{(m-1)} + \Delta w_{ji}^{(m)}, \tag{3}$$

where $w_{ji}^{(m)}$ denotes the weight of the connection between the jth element of the upper layer and the ith element of the lower layer†, in the mth learning iteration.

In GDR a weight change $\Delta w_{ji}^{(m)}$ in equation (3) is calculated as follows:

$$\Delta w_{ji}^{(m)} = \eta \cdot \delta_j^{(m)} \cdot O_i^{(m)} + \alpha \cdot \Delta w_{ji}^{(m-1)}, \tag{4}$$

where η and α denote the learning rate and the coefficient of the momentum term, respectively; $O_i^{(m)}$ is the output value of the ith element in the previous

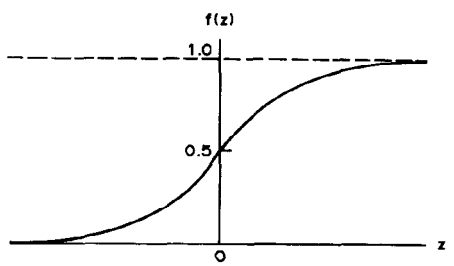


Fig. 3. Sigmoid function: $f(z) = \frac{1}{1 + e^{-z}}$.

†The first subscript of w denotes an element belonging to the upper layer, and the second subscript denotes one belonging to the lower layer. A connection from the bias unit to the jth element is represented as w_{j0}.

layer. The momentum term prevents divergent oscillations. The error signal of the jth element in the mth learning iteration $\delta_j^{(m)}$ in equation (4) is determined as follows:

If j belongs to the output layer:

$$\delta_j^{(m)} = (t_j^{(m)} - y_j^{(m)}) \cdot f_j'(\sum_i w_{ji}^{(m)} O_i^{(m)} + w_{j0}^{(m)}) \quad (5)$$

and if j belongs to the hidden layer(s),

$$\delta_j^{(m)} = f_j'(\sum_i w_{ji}^{(m)} O_i^{(m)} + w_{j0}^{(m)}) \cdot \sum_k \delta_k^{(m)} \cdot w_{kj}^{(m)}, \quad (6)$$

where f' is the derivative of the transfer function stated before.

Therefore, GDR computes an error for each element in the output and hidden layers using equations (5) and (6) and recursively updates the weights of all the layers using equation (4), starting from the output layer and working backwards until the input layer.

3. DESCRIPTION OF THE CASE STUDIES

In order to analyze the learning, recall and generalization characteristics of neural networks for process fault diagnosis in steady state processes, two case studies were investigated. The process units involved in these examples, viz. a continuously stirred tank reactor (CSTR) and a distillation column are ubiquitous pieces of equipment in the chemical and petrochemical industries. We wished to demonstrate the usefulness of the neural network approach in the automated fault diagnosis of chemical plants using these simple but representative examples.

The first case study (cf. Fig. 4) consists of a jacketed CSTR in which an irreversible, exothermic first-order reaction A→B takes place. The reactor is provided with three control loops that control the outlet temperature, the reactor holdup and the outlet concentration. In order to validate the capability of the neural network methodology to diagnose process failures in more complex equipment trains, we studied another example where the model chemical plant of Fig. 4 was augmented by the addition of a distillation column downstream. Figure 5 illustrates the augmented chemical plant consisting of a reactor and distillation column. The reactor effluent stream containing a binary mixture of A and B is fed into the distillation column where it is separated into a distillate stream containing 98% A and a bottoms stream containing 2% A. The column is provided with PI controllers that control the overhead and bottoms product compositions by manipulating the reflux rate R, and the vapor boilup rate V_a, respectively. Table 1 lists selected malfunctions that occur in the plant shown in Fig. 4. Table 2 presents the selected malfunctions that can occur in the augmented plant of Fig. 5. Faults F_1–F_6 are global, in the sense that they affect the entire plant. Faults F_7 and F_8 are local to the distillation column.

The measurement patterns required for training and testing the neural network were obtained from simulation programs that model the behavior of the plants. Various malfunctions were simulated and the *new steady-state* measurement patterns obtained were used to train the network. This approach can be used for processes with slow dynamics and also for batch processes. As an alternative to simulation, the historical database of an actual chemical plant can be used to provide the training data.

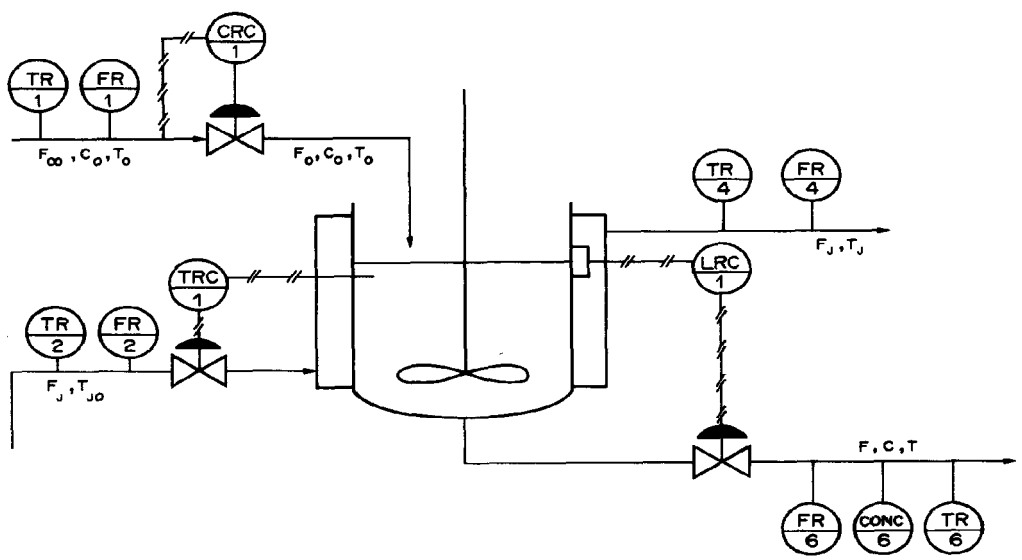


Fig. 4. Model chemical plant (reactor case study).

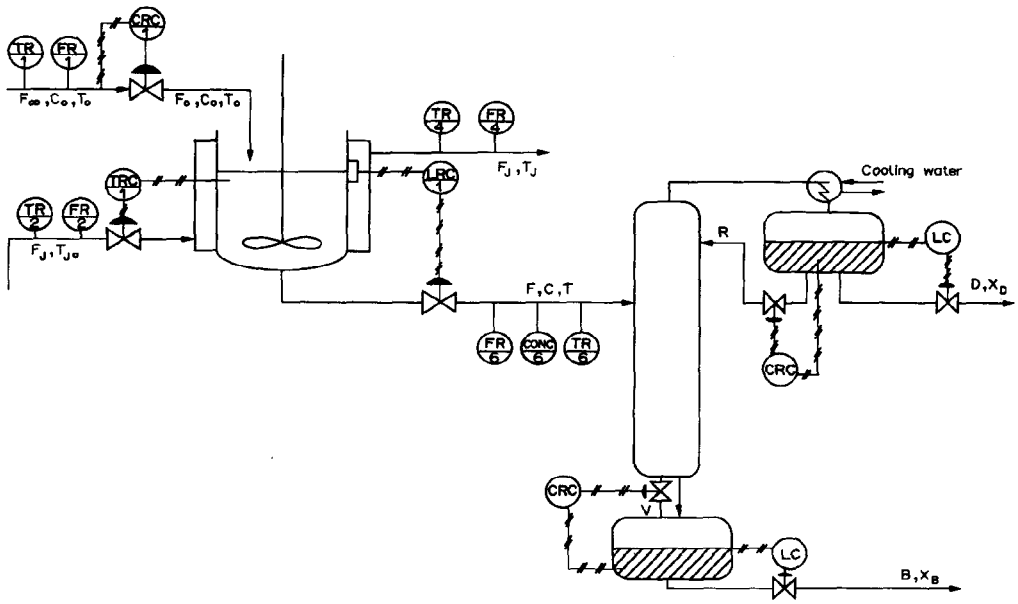


Fig. 5. Model chemical plant (reactor–distillation column case study).

The case studies under consideration are more complex than those treated by Hoskins and Himmelblau (1988) and Venkatasubramanian and Chan (1989). Each malfunction affects almost all symptoms, viz. outlet concentration of reactant C, outlet temperature T, reactor holdup V, outlet flowrate F, outlet temperature of coolant T_j and coolant flowrate F_j (cf. Fig. 6). This was not the case in the study of Venkatasubramanian and Chan (1989). In that work, the faults and symptoms were clearly delineated in a tree-like structure. Not more than three symptoms resulted from any one fault. Such a structure makes it possible to write "if... then" production rules for diagnostic expert systems. The tangled inference network in the present study makes such an implementation difficult. In the Hoskins and Himmelblau study, it was possible, in some cases, to infer the cause of abnormal symptoms from a single sensor reading; e.g. by comparing the observed temperature with the normal temperature, one can say that the responsible malfunction is high or low inlet temperature. In the present study, the malfunctions are not directly deducible from any single component of the process state vector (measurement pattern), used as the input to the neural network. In addition, we do not limit diagnosis to single faults. We also analyze the ability of the neural networks to generalize to multiple fault scenarios.

Table 1. Selected malfunctions for reactor case study

Malfunction (fault)	Symbol
High inlet flowrate	F_1
Low inlet flowrate	F_2
High inlet concentration of reactant	F_3
Low inlet concentration of reactant	F_4
High inlet temperature of reactant stream	F_5
Low inlet temperature of reactant stream	F_6

4. ANALYSIS OF THE LEARNING, RECALL AND GENERALIZATION CHARACTERISTICS

4.1. Reactor case study

4.1.1. Training scheme and recall. In the case study of Fig. 4, the variables that cause malfunctions are the inlet flowrate, inlet temperature and the inlet concentration of reactant. Deviations beyond $\pm 5\%$ of the normal value of these variables are assumed to result in malfunction of the plant. Deviations within the $\pm 5\%$ threshold are assumed to not perturb the "normal" plant operation. Table 3 indicates the 12 unnormalized measurement patterns used to train the network. These data were normalized between 0 and 1 before training.

The neural network employed in this case study has six input nodes corresponding to the six state variables of the system, and six output nodes corresponding to the six possible malfunctions. Various number of hidden units were explored to find the network architecture that gives the least error during recall. Figure 7 shows a sample network with five hidden

Table 2. Selected malfunctions for reactor-distillation column case study

Malfunction (fault)	Symbol
High flowrate at reactor inlet	F_1
Low flowrate at reactor inlet	F_2
High concentration of A at reactor inlet	F_3
Low concentration of A at reactor inlet	F_4
High temperature at reactor inlet	F_5
Low temperature at reactor inlet	F_6
Bottoms composition controller failure	F_7
Distillate composition controller failure	F_8

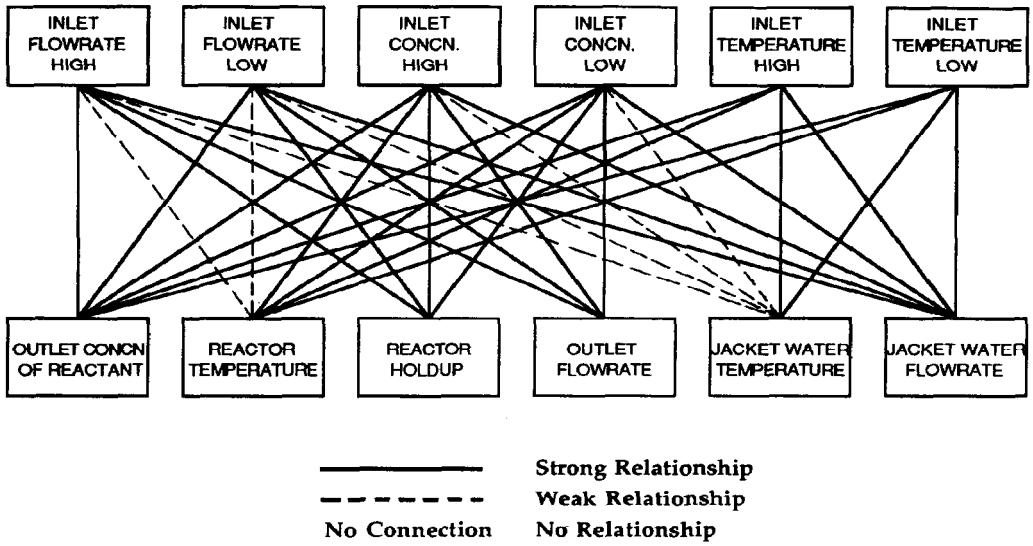


Fig. 6. Fault–symptom relationships for the reactor case study.

units. Different numbers of learning iterations were investigated. A learning rate η and a momentum term coefficient α [cf. equation (4)] of 0.9 and 0.6, respectively, were chosen for this study. In a following section, we discuss the effect of these two coefficients on the performance of the network, and the amount of training needed to achieve a certain performance.

The error measure used in this study is the r.m.s. error E defined as:

$$E = \sum_p E_p/P; \quad p = 1, 2, \ldots, P$$

and

$$E_p = \sqrt{\sum_i \frac{(d_i - o_i)^2}{N}}; \quad i = 1, , \ldots, N,$$

where E_p is the r.m.s. error for training data-set p; P is the number of training data-sets; d_i and o_i are the desired and actual outputs of node i, and N is the number of output nodes. The desired output is discretized between 0 and 1. For 15% deviations, if a fault F_i occurred, $d_i = 1$; otherwise $d_i = 0$. For 5% deviations, $d_i = 0.333$ for fault F_i and 0 otherwise. Figures 8 and 9 indicate the r.m.s. error during recall as a function of number of learning iterations, and the number of hidden units, respectively. As seen from Fig. 8, for each architecture of hidden units, the recall error decreases monotonically at first with increase in the learning iterations, and then flattens out after about 583 time steps (7000 iterations†).

It should be noted, however, that the error E defined as above is a measure of how close the actual output of the network is to the desired output. It is *not* a direct measure of the percentage of cases wrongly diagnosed. For example, the five hidden unit network trained for 417 time steps (5000 iterations) has an r.m.s. recall error of 0.0263 or 2.6%. However, this does *not* imply that the network failed to provide correct diagnosis in 2.6% of the cases tested. In reality, this network is successful in correctly identifying malfunctions in *all* the cases studied—i.e. 100%

†The number of time steps is the number of iterations divided by the number of training data-sets.

Table 3. Measurement patterns used for training (reactor case study)[a]

Number	Fault	C	T	V	F	T_j	F_j
1	$F_1(+15\%)$	0.2575	600.66	48.6	46.0	595.0	52.5
2	$F_2(-15\%)$	0.2307	599.17	47.4	34.0	594.2	46.6
3	$F_3(+15\%)$	0.2520	602.82	47.6	36.3	596.1	61.2
4	$F_4(-15\%)$	0.2315	596.74	48.4	43.7	592.9	36.9
5	$F_5(+15\%)$	0.2020	608.44	48.0	40.0	598.9	83.7
6	$F_6(-15\%)$	0.2991	589.73	48.0	40.0	588.9	8.8
7	$F_1(+5\%)$[b]	0.2494	600.24	48.2	42.0	594.8	50.8
8	$F_2(-5\%)$	0.2405	599.73	47.8	38.0	594.5	48.8
9	$F_3(+5\%)$	0.2480	600.99	47.9	38.8	595.2	53.9
10	$F_4(-5\%)$	0.2414	598.94	48.1	41.2	594.1	45.7
11	$F_5(+5\%)$	0.2296	602.96	48.0	40.0	596.2	61.8
12	$F_6(-5\%)$	0.2617	596.83	48.0	40.0	592.9	37.2
—	Normal	0.2450	600.00	48.0	40.0	594.6	49.9

[a]Units: C (mol ft^{-3}); T, T_j (^0R); V (ft^3); F, F_j (ft^3 h^{-1}).
[b]Deviations within $\pm$ 5% are not considered faults (see text).

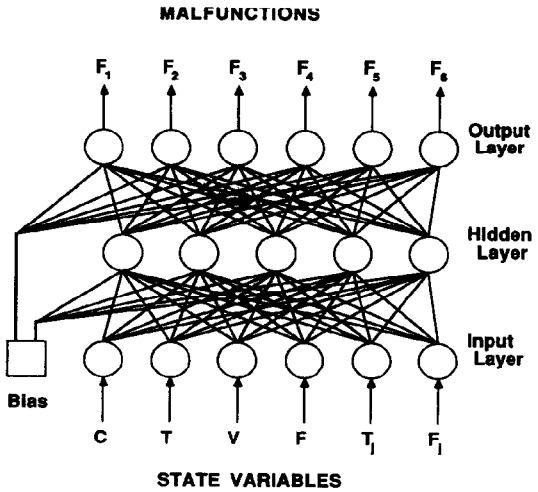


Fig. 7. Neural network with six input, five hidden and six output units.

"diagnostic accuracy". Table 4 indicates the output of this five hidden unit network during recall of the 12 training data-sets. In each case, the output of the node that signals a fault (e.g. Output Node 1 in the first data set), is quite close to the desired output of 1 (for $\pm$ 15% faults) or 0.333 (for $\pm$ 5% faults). At the same time, the outputs of the other output nodes are close to their desired output of zero. The r.m.s. error E merely shows quantitatively whether or not we are making any "progress" on further training. All the following plots of the error E as a function of the extent of training, should be viewed in this light.

Figure 9 shows that the number of hidden units, greater than three, has a very weak influence on the r.m.s. error. At the low learning iterations (83 and 250 time steps), there are some local minima for error, but the effect of the number of hidden units on the error decreases as the number of learning iterations increase.

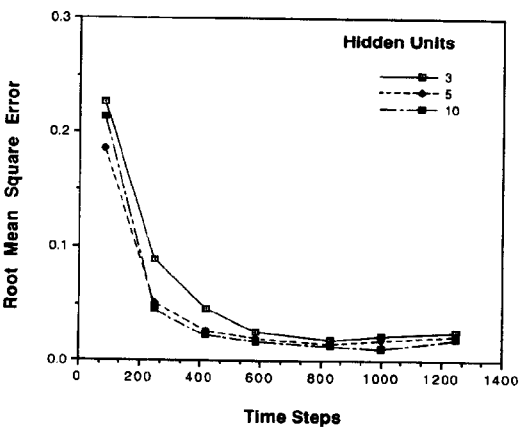


Fig. 8. Recall error as a function of time steps (reactor case study).

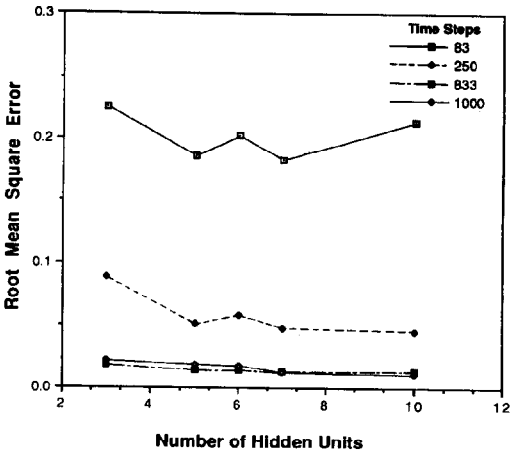


Fig. 9. Recall error as a function of hidden units (reactor case study).

4.1.2. Generalization to untrained single and multiple faults. Given the normal plant data (listed on the last line of Table 3, that was not used for training), the network with five hidden units, trained for 833 time steps, was able to perceive it as such. Networks with five hidden units, trained up to 1000 time steps were chosen for generalization of patterns resulting from untrained single faults. Measurement patterns corresponding to 10 and 25% deviations were successfully generalized by all the networks. These results are represented in Fig. 10. For 10% generalization, a minimum error state is observed with the network trained for 250 time steps. This happens because of the following reasons. The target output values of training patterns which the network has learned are either 0, 0.333 or 1; more trained networks tend to push the output during generalization toward one of these three values (although, in this case, the desired value is 0.667). This leads to a slightly higher error with an increase in training time steps. In other words, a longer training does not necessarily lead to better performance in this sort of generalization which is an interpolation between two training patterns corresponding to 5 and 15% deviations. However, despite this drop in performance, all the networks correctly identified the fault involved. Also, it should be noted that the increase in error from about 200 time steps to 1000 time steps is small as compared to the large drop in error between 100 and 200 time steps. For generalization in the 25% case, the error monotonically decreases for networks trained for longer extents. Since 25% deviations are an extrapolation of the training patterns, and since the maximum possible output of the network is 1 (corresponding to 15% deviation), the network can unambiguously generate the correct output, 1. In this case, networks trained longer exhibit better performance during generalization.

Next, multiple faults were simulated and the ability of the network to correctly identify these malfunc-

Table 4. Comparison of actual and desired outputs of a five hidden unit network trained for 417 time steps during recall (reactor case study)

Fault simulated	o/d	Output Node 1	Output Node 2	Output Node 3	Output Node 4	Output Node 5	Output Node 6	Fault diagnosed
F_1 (15%)	o	**0.9629**	0.0000	0.0116	0.0488	0.0107	0.0151	F_1
	d	**1.0000**	0	0	0	0	0	
F_2 (15%)	o	0.0000	**0.9642**	0.0467	0.0309	0.0127	0.0072	F_2
	d	0	**1.0000**	0	0	0	0	
F_3 (15%)	o	0.0019	0.0618	**0.9354**	0.0001	0.0076	0.0196	F_3
	d	0	0	**1.0000**	0	0	0	
F_4 (15%)	o	0.0513	0.0216	0.0000	**0.9507**	0.0261	0.0042	F_4
	d	0	0	0	**1.0000**	0	0	
F_5 (15%)	o	0.0090	0.0277	0.0357	0.0312	**0.9492**	0.0000	F_5
	d	0	0	0	0	**1.0000**	0	
F_6 (15%)	o	0.0222	0.0196	0.0376	0.0203	0.0000	**0.9501**	F_6
	d	0	0	0	0	0	**1.0000**	
F_1 (5%)	o	**0.3202**	0.0014	0.0156	0.0526	0.0088	0.0176	F_1
	d	**0.333**	0	0	0	0	0	
F_2 (5%)	o	0.0005	**0.3166**	0.0547	0.0216	0.0151	0.0075	F_2
	d	0	**0.333**	0	0	0	0	
F_3 (5%)	o	0.0081	0.0283	**0.3028**	0.0031	0.0117	0.0119	F_3
	d	0	0	**0.333**	0	0	0	
F_4 (5%)	o	0.0264	0.0237	0.0018	**0.3099**	0.0120	0.0106	F_4
	d	0	0	0	**0.333**	0	0	
F_5 (5%)	o	0.0117	0.0256	0.0388	0.0275	**0.3363**	0.0004	F_5
	d	0	0	0	0	**0.333**	0	
F_6 (5%)	o	0.0201	0.0227	0.0249	0.0365	0.0002	**0.3382**	F_6
	d	0	0	0	0	0	**0.333**	

o = actual output of the network; d = desired output of the network.

tions was investigated. Four multiple fault cases were considered.

Case A—Inlet concentration *10% higher* and inlet temperature *10% higher* than normal.

In this example, the increased inlet concentration by itself would cause an increase in outlet concentration. However, the increase in temperature would enhance reaction rates and therefore decrease the outlet concentration of reactant. Thus, in this candidate case, the effects of the multiple malfunctions on the reactant outlet concentration would tend to cancel out.

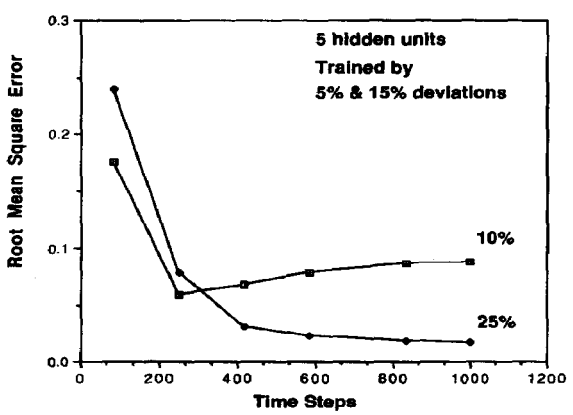


Fig. 10. Generalization to single faults (reactor case study).

On the other hand, the increased inlet concentration and inlet temperature, lead to higher outlet temperature. Thus, the effects of the malfunctions on the outlet temperature are synergistic.

Case B—Inlet concentration *10% lower* and inlet temperature *10% higher* than normal.

In this case, we can see by an analysis similar to that in Case A, that the multiple faults would tend to decrease the outlet concentration of reactant significantly, while their effects on the outlet temperature would tend to cancel out.

Case C—Inlet flowrate *10% higher* and inlet temperature *10% higher* than normal.

The increase in flowrate would cause both the outlet flowrate and the reactor holdup to increase. Also, the increase in flowrate would decrease the residence time in the reactor, lowering conversion. Hence the outlet reactant concentration would tend to increase. However, the increase in temperature would result in greater conversion, causing a decrease in the outlet concentration of the reactant. Hence the effects of these two faults on the outlet reactant concentration would tend to cancel out. Similarly, the effects on outlet temperature would also tend to cancel out.

Case D—Inlet flowrate *10% lower* and inlet temperature *10% higher* than normal.

The decrease in flowrate would lead both the outlet flowrate and the reactor holdup to decrease. By an analysis similar to Case C, we see that the multiple

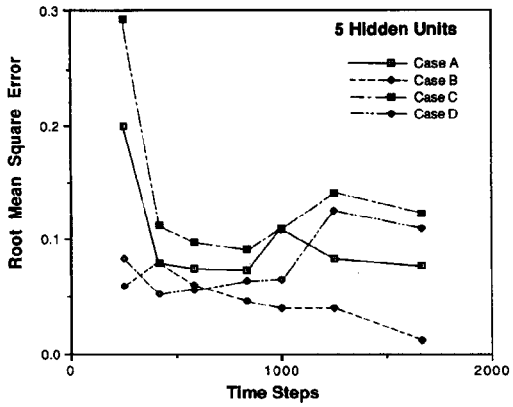


Fig. 11. Generalization to multiple faults (reactor case study).

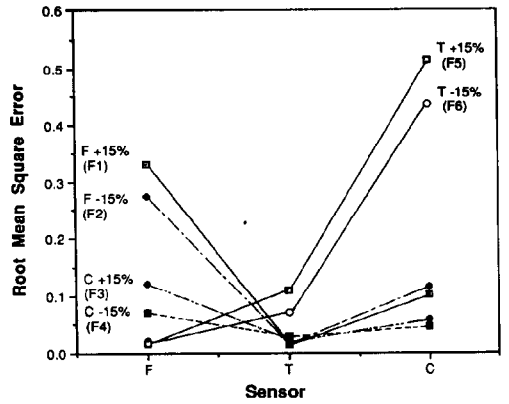


Fig. 12. Generalization with sensor faults (reactor case study).

faults would tend to lower the outlet reactant concentration synergistically, while significantly raising the outlet temperature.

These cases were tested on networks with five hidden units, trained up to 1667 time steps. In all cases, all the networks were able to accurately diagnose both causal origins of the observed symptoms. These results are presented in Fig. 11. In Fig. 11, all fault cases are correctly identified by a network with a small number of learning iterations (400 time steps). All cases, except B, do not show further improvement when tested on networks trained longer than 400 time steps. It should be noted here that no wrong diagnosis is observed in this experiment, even though output values are slightly deviated from the target values which reflect the r.m.s. errors of less than 15% in Fig. 11.

4.1.3. Generalization with sensor faults. The ability of the network with five hidden units, trained for 833 time steps, to diagnose faults in the presence of faulty sensor readings was investigated. Here, by a faulty sensor, we mean that the particular sensor always reads the normal value even though the actual value of that parameter might be abnormal. These results are presented in Fig. 12. The network was able to correctly diagnose (r.m.s. error $\leqslant 12\%$) faults F_1 and F_2 in the presence of faults in the C and T sensors, faults F_3 and F_4 in the presence of faults in the F, C and T sensors, and faults F_5 and F_6 in the presence of faults in the F and T sensors. The network failed to diagnose faults F_1 and F_2 when the outlet flowrate sensor F failed. This suggests that the F sensor is critical to the diagnosis of flowrate-related faults as may be expected. Also the network did not diagnose the correct faults F_5 and F_6, when the C sensor failed. Interestingly, the C sensor does not seem to be critical in diagnosing concentration-related faults and the T sensor is not important in detecting temperature-related faults. These results demonstrate the robustness and fault-tolerance capabilities of neural networks in diagnostic applications. Even in the presence of faulty operation of certain sensors, the network is able to offer correct diagnosis in most of the cases tested.

4.1.4. Effect of the learning rate and momentum term. The effects of the two coefficients in equation (4), viz. the learning rate η and the momentum term coefficient α on the performance of a network were also examined. Both coefficients, as a rule, are determined based on the tradeoff between a performance and a computational cost; a larger learning rate or momentum term coefficient leads to a faster convergence but a poorer performance and vice versa. Figure 13 shows the number of time steps of learning required to achieve *satisfactory* performance (5% r.m.s. error) and the recall error at convergence as a function of the learning rate. According to this figure, the learning rate of 0.5 seems to be the optimal choice (0.6% error) as far as the recall error at convergence is concerned. A learning rate of 0.9 was chosen for this study because of the lower computational cost and because the 2–3% error at convergence was accept-

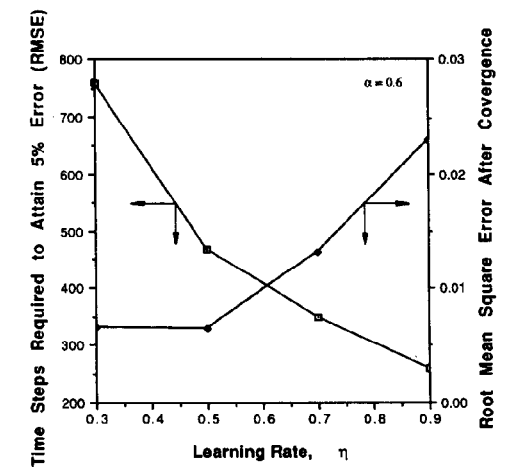


Fig. 13. Number of time steps required for satisfactory performance (5% r.m.s. recall error) and the recall error at convergence as a function of the learning rate—reactor case study.

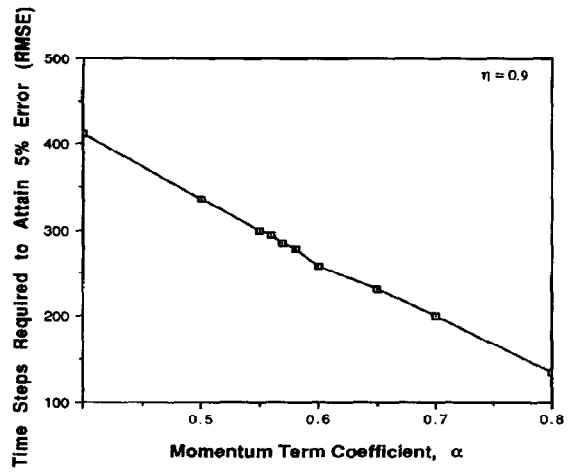


Fig. 14. Effect of the coefficient of the momentum term on the number of time steps required for satisfactory performance (5% r.m.s. recall error)—reactor case study.

able. The effect of the coefficient of the momentum term is shown in Fig. 14. Increasing the momentum term had a similar effect as increasing the learning rate with respect to the speed of learning. This trend is opposite to that observed by Huang and Lippmann (1987). This implies that the direction of the previous weight vector [the second term on the RHS of equation (4)] is similar to the direction of the current weight vector and also that it virtually learns in the steepest descent manner. Further consideration will be necessary to discuss the domain-dependency of optimality of these two coefficients.

4.1.5. Analysis of weights and role of hidden units. Some interesting observations regarding the weights and hidden units are worth mentioning. Figure 15 shows the changes in weights of connections to hidden unit 1 as the five hidden unit network was being trained. In Fig. 15, most weights change dramatically and nonmonotonically before convergence is reached. Similar behavior was observed with the other hidden units. Table 5 illustrates the importance of each hidden unit with respect to the performance of the network. The top line represents the error of the network with five hidden units after convergence. Each row below that corresponds to the error of the same network which is missing the hidden unit specified. The table shows how the performance of the network during recall is degraded by one missing hidden unit; in other words, how each hidden unit contributes to the activation of correct outputs. Although any missing hidden unit, more or less, causes degradation of the performance, the degree of degradation varies from one unit to the other. For instance, the loss of the fifth hidden unit results in more than 30% error while that of the fourth unit causes only a 13% error.

We have not been able to ascribe any clear physical connotations to the magnitudes of the weights

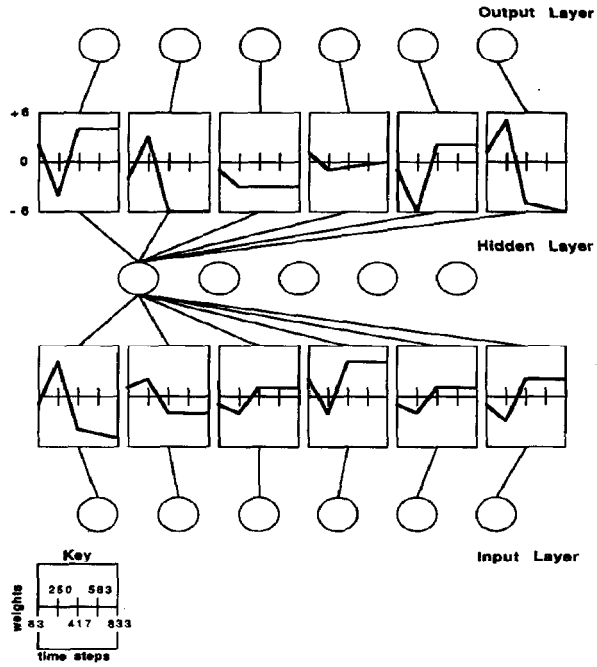


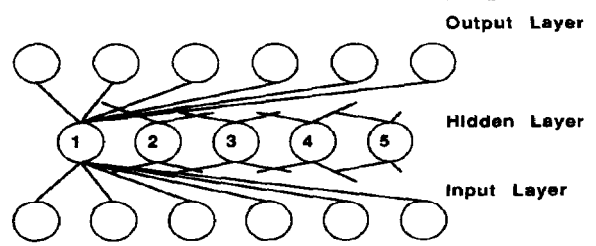
Fig. 15. Changes of the weights associated with the first hidden unit in a five hidden unit network—reactor case study.

(strengths of connections) and to the hidden units. Jones and Hoskins (1987) surmise in their simple example that strong correlations would exist between features of the input on one hand and the entity coded for by each hidden unit on the other. Similarly, such a relationship would be present between the hidden units and the features of the output. However, as an anonymous reviewer pointed out, several researchers have discounted the notion that hidden nodes and strengths of the connections possess special meaning.

4.1.6. Two hidden layers. Several kinds of two-hidden-layer networks were examined (cf. Fig. 16). In

Table 5. Degradation of network performance by a missing hidden unit (reactor case study)

Missing hidden unit	Error (r.m.s. error)
None	0.0142
1	0.1731
2	0.2031
3	0.1573
4	0.1345
5	0.3102



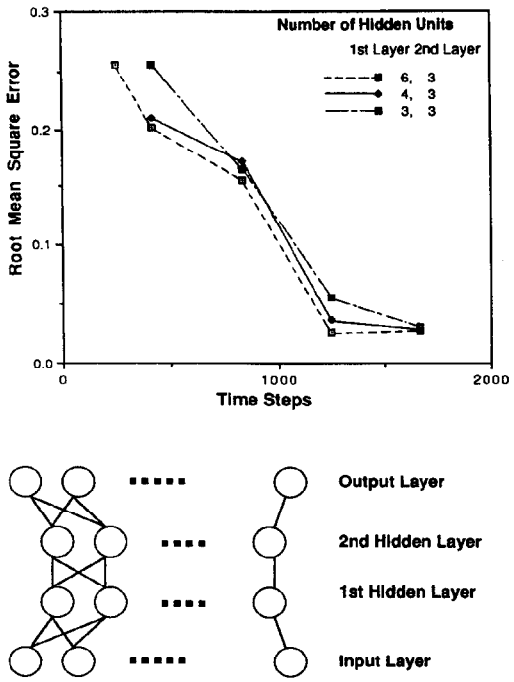


Fig. 16. Recall error as a function of time steps for two-hidden-layer networks (reactor case study).

general, these networks required about twice the learning iterations as one-hidden-layer networks. The performance, however, was similar to or even worse than that of a one-hidden-layer network. This implies that the reactor case under study is not complicated enough to be improved by a two-hidden-layer network, although it has been pointed out (Lippmann, 1987; Hanson and Burr, 1987) that for continuous inputs, as is the case in the present paper, a two-hidden-layer network is capable of discriminating any complex region.†

4.2. *Reactor–distillation column case study*

4.2.1. Training scheme and recall. In the chemical plant of Fig. 5, the variables that cause malfunctions are the inlet flowrate, inlet temperature and the inlet concentration of reactant. Here also, deviations beyond $\pm 5\%$ of the normal value of these variables are assumed to result in malfunction of the plant. Deviations within the $\pm 5\%$ threshold are assumed to not perturb the "normal" plant operation. In addition, the bottoms and overhead product composition controllers can fail. Table 6 indicates the 20 unnormalized measurement patterns used to train the network. The data were normalized before training.

The neural network employed in this case study has eight output nodes corresponding to the eight possible malfunctions. The state variables characterizing this plant are the concentration of A at reactor inlet C, the reactor effluent temperature T, the reactor holdup V, the reactor outlet flowrate (feed rate to distillation column) F, the outlet temperature of cooling water in the reactor jacket T_j, the flowrate of cooling water F_j, the reflux rate R, the vapor boilup rate V_a, the bottoms flow rate B, the distillate flow rate D, the bottoms product composition X_B, and the distillate product composition X_D. The values of X_B and X_D are expressed in terms of the mole fraction of the more volatile component A.

Here again, the learning algorithm used was back-propagation. Initially a network with 10 input nodes, corresponding to the state variables C, T, V, F, T_j, F_j, R, V_a, B and D was chosen for training. The number of hidden units was six. This network failed to recall correctly the last two malfunctions in data-sets 13–20 (cf. Table 6), even after 1000 time steps (20,000 iterations), although it was able to correctly recall the first six malfunctions after 750 time steps. This is probably due to the following reason. In data-sets 7 and 13, *seven* of the *ten* state variables are identical. (Both these cases deal with 5% excess flowrate at reactor inlet.) The state variables R, V_a and B are different in the two cases, the difference arising because of the bottoms composition controller failure in data-set 13. Since seven out of ten state variables were the same in the two cases, the network probably regarded the deviations in the rest of the variables as noise, and hence did not recognize malfunction F_7 in data-set 13. Similar arguments can be advanced for data-sets 14–20.

When controller failure occurs, the deviations (from the normal) of X_B and X_D are substantial. So, it was felt that replacement of state variables B and D in the input vector, by X_B and X_D, would improve the diagnostic function of the network. Accordingly, networks with 10 input nodes and 6, 7 or 8 hidden nodes were evaluated. These networks also failed to recognize malfunctions F_7 and F_8 even after 1000 time steps. Here again in several pairs of data (e.g. data 7 and 13; data 7 and 17; data 8 and 14; etc.), *seven* of the *ten* state variables are identical, and the network treats them as noise. The network with seven hidden units required 4000 time steps (80,000 iterations) for correct recall of all eight malfunctions.

One of the advantages of neural networks is their resistance to input noise. Here, this positive feature was clearly working against correct diagnosis. In order to address this problem, it was decided to incorporate in the input vector, two more state variables, B and D, in addition to X_B and X_D. Now, the number of variables that deviate when malfunctions F_7 or F_8 occur (with reactor functioning normally) are *five* instead of just *three* before. This would enable the network to better discriminate between data pairs like 7–13 and 8–14.

Accordingly, a network with 12 input nodes corresponding to the 12 state variables

†A single hidden layer network can delineate closed convex regions in hyperspace, even in the case of continuous inputs; but for forming arbitrary regions in the space, two hidden layers are required.

Table 6. Measurement patterns used for training networks for the reactor–distillation column case study

Number	Fault	C	T	V	F	T_j	F_j	R	V_a	X_B	X_D	B	D
1	$F_1(+15\%)$	0.2575	600.66	48.6	46.0	595.0	52.5	58.75	82.47	0.0200	0.9800	22.28	23.72
2	$F_2(-15\%)$	0.2307	599.17	47.4	34.0	594.2	46.6	43.70	59.31	0.0200	0.9800	18.37	15.61
3	$F_3(+15\%)$	0.2520	602.82	47.6	36.3	596.1	61.2	46.69	62.51	0.0200	0.9800	20.48	15.82
4	$F_4(-15\%)$	0.2315	596.74	48.4	43.7	592.9	36.9	55.48	79.36	0.0200	0.9800	19.81	23.88
5	$F_5(+15\%)$	0.2020	608.44	48.0	40.0	598.9	83.7	51.41	67.42	0.0200	0.9800	23.95	16.00
6	$F_6(-15\%)$	0.2991	589.73	48.0	40.0	588.9	8.8	49.96	74.06	0.0200	0.9800	15.91	24.09
7	$F_1(+5\%)$[a]	0.2494	600.24	48.2	42.0	594.8	50.8	53.78	74.73	0.0200	0.9800	21.05	20.95
8	$F_2(-5\%)$	0.2405	599.73	47.8	38.0	594.5	48.8	48.76	67.01	0.0200	0.9800	19.75	18.25
9	$F_3(+5\%)$	0.2480	600.99	47.9	38.8	595.2	53.9	49.82	68.10	0.0200	0.9800	20.52	18.29
10	$F_4(-5\%)$	0.2414	598.94	48.1	41.2	594.1	45.7	52.68	73.63	0.0200	0.9800	20.25	20.95
11	$F_5(+5\%)$	0.2296	602.96	48.0	40.0	596.2	61.8	51.42	69.72	0.0200	0.9800	21.70	18.30
12	$F_6(-5\%)$	0.2617	596.83	48.0	40.0	592.9	37.2	51.00	71.97	0.0200	0.9800	19.02	20.98
13	$F_1(+5\%)$ and F_7	0.2494	600.24	48.2	42.0	594.8	50.8	50.20	70.86	0.0329	0.9800	21.34	20.66
14	$F_2(-5\%)$ and F_7	0.2405	599.73	47.8	38.0	594.5	48.8	52.53	70.86	0.0128	0.9799	19.62	18.33
15	$F_3(+5\%)$ and F_7	0.2480	600.99	47.9	38.8	595.2	53.9	52.51	70.86	0.0146	0.9799	20.43	18.35
16	$F_4(-5\%)$ and F_7	0.2414	598.94	48.1	41.2	594.1	45.7	50.09	70.86	0.0287	0.9800	20.44	20.77
17	$F_1(+5\%)$ and F_8	0.2494	600.24	48.2	42.0	594.8	50.8	51.27	72.36	0.0200	0.9738	20.91	21.09
18	$F_2(-5\%)$ and F_8	0.2405	599.73	47.8	38.0	594.5	48.8	51.27	69.43	0.0200	0.9848	19.85	18.16
19	$F_3(+5\%)$ and F_8	0.2480	600.99	47.9	38.8	595.2	53.9	51.27	69.50	0.0200	0.9830	20.57	18.23
20	$F_4(-5\%)$ and F_8	0.2414	598.94	48.1	41.2	594.1	45.7	51.27	72.29	0.0200	0.9768	20.18	21.02

[a]Deviations within $\pm 5\%$ are not considered faults (see text).
Units: C (mol ft^{-3}); T, T_j (^{0}R); V (ft^3); F, F_j, R, V_a, B, D (ft^3 h^{-1}); X_B, X_D (dimensionless).

$(C, T, V, F, T_j, F_j, R, V_a, X_B, X_D, B, D)$ and eight outputs nodes corresponding to the eight possible malfunctions was set up. Networks with various number of hidden units were explored. The six hidden unit network failed to learn the last two malfunctions even after 4500 time steps. This probably means that the problem is too complex to be solved by a six hidden unit network. The 7-, 9- and 10- hidden unit networks were able to correctly recall all the faults after training for 3500 time steps (70,000 iterations). This large number of iterations is needed because of the size of the network and because of the problem complexity. Figure 17 illustrates the r.m.s. error during recall as a function of the number of time steps, with the number of hidden units as a parameter. Figure 17 indicates that the recall error monotonically decreases at first with an increase in the number of time steps and then tends to flatten out after about 3500 time steps (70,000 iterations). At 4000 time steps (80,000 iterations) the nine hidden unit network gives the lowest error during recall.

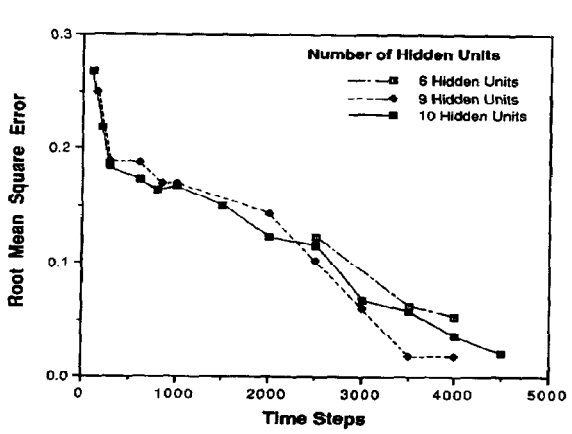


Fig. 17. Recall error as a function of time steps (reactor–distillation column case study).

4.2.2. Generalization to untrained single and multiple faults. The network with nine hidden units, trained up to 4000 time steps was chosen for generalization of single faults. Measurement patterns corresponding to 25% deviations in the reactor inlet flowrate and reactor inlet concentration were successfully generalized. Next, multiple faults were simulated and the ability of the network to correctly identify these malfunctions was investigated. Eight multiple fault cases were considered.

Case A—15% high flowrate at reactor inlet and bottoms composition controller failure.
Case B—15% low flowrate at reactor inlet and bottoms composition controller failure.
Case C—15% high concentration of A at reactor inlet and bottoms composition controller failure.
Case D—15% low concentration of A at reactor inlet and bottoms composition controller failure.
Case E—15% high flowrate at reactor inlet and distillate composition controller failure.
Case F—15% low flowrate at reactor inlet and distillate composition controller failure.
Case G—15% high concentration of A at reactor inlet and distillate composition controller failure.

Table 7. Generalization to multiple faults (12–9–8 network) (reactor–distillation column case study)

Time steps	Fault cases A (F_1, F_7)	B (F_2, F_7)	C (F_3, F_7)	D (F_4, F_7)	E (F_1, F_8)	F (F_2, F_8)	G (F_3, F_8)	H (F_4, F_8)
4000	+	F_2(r) F_7(n) F_8(w)	F_3(r) F_7(n) F_8(w)	+	F_1(n) F_4(w) F_8(r)	+	+	+
5000	+	F_2(r) F_7(n) F_8(w)	F_3(r) F_7(n) F_8(w)	+	F_1(n) F_4(w) F_8(r)	+	+	+
5500	+	F_2(r) F_7(n) F_8(w)	F_3(r) F_7(n) F_8(w)	+	F_1(n) F_8(r)	+	+	+

+ = both faults correctly identified.
r = a fault correctly identified.
n = a fault not identified.
w = a case misidentified as a fault.

Case H—15% low concentration of A at reactor inlet and distillate composition controller failure.

Generalization for multiple faults was attempted with networks with 12 input and nine hidden units. Networks trained for 4000, 5000 and 5500 time steps were explored. All the networks were able to correctly identify both malfunctions in cases A, D, F, G and H. All the three networks failed to accurately diagnose one of the two malfunctions in cases B, C and E. In these cases, the 4000- and 5000- time step networks gave a wrong diagnosis as well. The 5500-time step network made a wrong diagnosis in cases B and C only. The result for each fault case is shown in Table 7.

Networks with 10 input (corresponding to the state variables C, T, V, F, T_j, F_j, R, V_a, X_B and X_D) and seven hidden units were next investigated for multiple fault generalization. Networks trained up to 4000, 4250 and 4750 time steps were tried. The results in terms of the r.m.s. error are presented in Fig. 18. Also each fault correctly diagnosed or not diagnosed is shown in Table 8. All the networks correctly detected both malfunctions in cases A, B, C and D. In cases E, F, G and H, the networks were able to identify only one of the two malfunctions. In each of these cases, the networks failed to diagnose malfunction F_8. The 4250- time step network shows some improvement over the 4000- time step network in diagnosing fault F_8 in case F. However, this improvement is not carried over to the 4750-time step network. This is possibly because longer training guarantees improvement in performance during recall but not always in generalization.

The above results are in contrast to those obtained from the networks with 12 input and nine hidden units. In addition to the differences in the cases accurately diagnosed, the 10–7–8 networks† make no wrong diagnosis. More research is required to explain these observations, and to improve the multiple fault generalization behavior.

†A p–q–r network is a network with p input, q hidden and r output units.

5. CONCLUSIONS

An analysis of the learning, recall and generalization characteristics of neural networks for detecting and diagnosing process failures in steady state processes has been presented. We have shown how neural networks are successful in diagnosing single faults in two case studies. Accurate recall and generalization behavior is observed. Performance during recall improves at first with an increase in the number of hidden units and with the amount of training, and then attains convergence. In general, performance during generalization improves with the extent of training. An increase in the learning rate η or the momentum term coefficient α leads to faster convergence during training. However, a larger learning rate leads to a higher recall error at convergence.

The network is also able to diagnose correctly even in the presence of faulty operation of certain sensors. These results indicate the robustness and fault tolerance capabilities of neural networks in diagnostic applications. Neural networks are not brittle. A missing hidden unit only results in a graceful degradation of performance. Two-hidden-layer networks required about twice the learning iterations as one-hidden-layer networks. The performance, however,

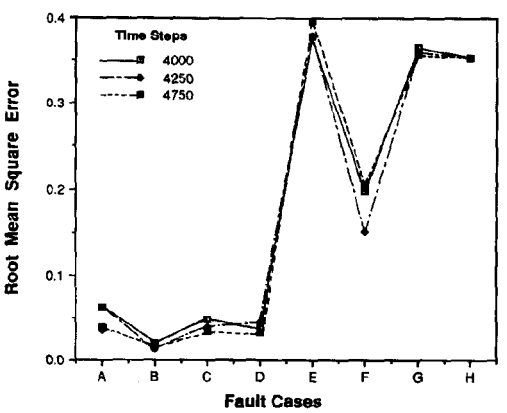


Fig. 18. Generalization to multiple faults using the 10–7–8 network (reactor–distillation column case study).

Table 8. Generalization to multiple faults (10–7–8 network) (reactor–distillation column case study)

Time steps	Fault cases A (F_1, F_7)	B (F_2, F_7)	C (F_3, F_7)	D (F_4, F_7)	E (F_1, F_8)	F (F_2, F_8)	G (F_3, F_8)	H (F_4, F_8)
4000	+	+	+	+	F_1(r) F_8(n)	F_2(r) F_8(n)	F_3(r) F_8(n)	F_4(r) F_8(n)
4250	+	+	+	+	F_1(r) F_8(n)	+	F_3(r) F_8(n)	F_4(r) F_8(n)
4750	+	+	+	+	F_1(r) F_8(n)	F_2(r) F_8(n)	F_3(r) F_8(n)	F_4(r) F_8(n)

+ = both faults correctly identified.
r = a fault correctly identified.
n = a fault not identified.

was similar to or even worse than that of a one-hidden-layer network.

In order to extend the applicability of the model, we explored the ability of the network trained on single faults, to diagnose multiple faults. In the reactor case study, the network was able to accurately diagnose all the multiple fault patterns presented to it. This ability to generalize is important because it is quite tedious to train the network on the combinatorially large number of patterns resulting from multiple faults. In the reactor-distillation column case study, generalization to multiple faults was not fully successful in some instances. However, graceful degradation of the diagnostic function was observed in many of the multiple-fault cases that were not accurately diagnosed. More work is required to improve performance in multiple-fault diagnosis. Also, other learning algorithms could be tested and their performance compared with backpropagation. If multiple faults are unlikely to happen, or if detecting single faults is far more important than detecting multiple faults from the practical point of view, other algorithms, such as Bidirectional Associative Memory (BAM) (Kosko, 1988), Hamming Network (Lippmann, 1987) or Counter-propagation Network (Hecht-Nielsen, 1987) could be alternatively used by modifying the coding schemes. They are very useful for recall from noisy data, but lack the capability of generalization to novel inputs even in single fault cases. We are currently developing an extension of the neural network approach for fault diagnosis in dynamic chemical processes.

The neural network approach requires no explicit encoding of knowledge as in conventional expert systems. Therefore neural nets appear to be well-suited for applications in which knowledge extraction is difficult or in cases where the interrelationships between process parameters are too complex. Changes in the plant configuration necessitate retraining of the neural network, which is simpler than revising the complex knowledge base of an expert system. Another advantage of neural networks is their robustness and resistance to input noise. Thus they offer reliable diagnosis. Conventional expert systems are quite brittle in comparison. However, neural networks have no explanation capability and therefore offer no insight into the problem-solving process. A judicious combination of these two contrasting approaches is desirable and will be the subject of further work.

REFERENCES

Ahalt S. C., F. D. Garber, I. Jouny and A. K. Krishnamurthy, Performance of neural network classification of noisy radar signals. *Advances in Neural Information Processing Systems*, Vol. I (D. S. Touretzky, Ed.), pp. 281–288. Morgan Kaufmann, San Mateo (1989).

Denker J. S., W. R. Gardner, H. P. Graf, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel, H. S. Baird and I. Guyon, Neural network recognizer for handwritten ZIP code digits. *Advances in Neural Information Processing Systems*, Vol. 1 (D. S. Touretzky, Ed.), pp. 323–331. Morgan Kaufmann, San Mateo (1989).

Finch F. E. and M. A. Kramer, Narrowing diagnostic focus using functional decomposition. *AIChE Jl* **34**, 25–36 (1988).

Hanson S. J. and D. J. Burr, Knowledge representation in connectionist networks. Bell Communication Research Technical Report (1987).

Hecht-Nielsen R., Counterpropagation networks. *Proc. IEEE First Int. Conf. on Neural Networks*, San Diego, Vol. II, pp. 19–32. (1987).

Hoskins J. C. and D. M. Himmelblau, Artificial neural network models of knowledge representation in chemical engineering. *Comput. chem. Engng* **12**, 881–890 (1988).

Huang W. Y. and R. P. Lippmann, Comparisons between neural net and conventional classifiers. *Proc. IEEE First Int. Conf. on Neural Networks*, San Diego, Vol IV, 485–493 (1987).

Jones W. P. and J. C. Hoskins, Back-propagation, a generalized delta learning rule. *BYTE Mag.* Oct, 155–162 (1987).

Kohonen T., *Self-Organization and Associative Memory*. Springer-Verlag, Berlin (1984).

Kosko B., Bidirectional associative memories. *IEEE Trans. Systems. Man Cybernet.* **18** (1), 49–60 (1988).

Lapedes A. and R. Farber, Nonlinear signal processing using neural networks: prediction and system modeling. Los Alamos National Laboratory Technical Report LA-UR-87-2662 (1987).

Lippmann R. P., An introduction to computing with neural nets. *IEEE ASSP Mag.* April, 4–22 (1987).

Qian N and T. J. Sejnowski, Predicting the secondary structure of globular proteins using neural network models. *J. Molec. Biol.* **202**, 865–884 (1988).

Rich S. H. and V. Venkatasubramanian, Model-based reasoning in diagnostic expert systems for chemical process plants. *Comput. chem. Engng* **11**, 111–122 (1987).

Rumelhart D. E. and J. L. McClelland, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1: Foundations*. MIT Press, Cambridge, Mass. (1986).

Rumelhart D. E., G. E. Hinton and R. J. Williams, Learning internal representations by error propagation. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations* (D. E. Rumelhart and J. L. McClelland, Eds). MIT Press, Cambridge, Mass. (1986).

Sejnowski T. J. and C. R. Rosenberg, Parallel networks that learn to pronounce English text. *Complex Systems* **1**, 145–168 (1987).

Shum S. K., J. F. Davis, W. F. Punch III and B. Chandrasekaran, An expert system approach to malfunction diagnosis in chemical plants. *Comput. chem. Engng* **12**, 27–36 (1988).

Tesauro G. and T. J. Sejnowski, A parallel network that learns to play backgammon. *Artificial Intell.* **39**, 357–390 (1989).

Venkatasubramanian V., Inexact reasoning in expert systems: a stochastic parallel network approach. *Proc. The Second Conf. on Artificial Intell. Applics.* IEEE Computer Society, Miami Beach (1985).

Venkatasubramanian V. and K. Chan, A neural network methodology for process fault diagnosis. *AIChE Jl* **35**, 1993–2005 (1989).

Venkatasubramanian V. and S. H. Rich, An object-oriented two-tier architecture for integrating compiled and deep-level knowledge for process diagnosis. *Comput. chem. Engng* **12**, 903–921 (1988).

Werbos P. J., Beyond regression: new tools for prediction and analysis in the behavioral sciences. Ph.D. Thesis in Applied Mathematics, Harvard University (1974).

Widrow B. and M. Hoff, Adaptive switching circuits. 1960 IRE WESCON Convention Record, Part 4, 96–104, August (1960).