DOE/ER/75700-1

DOE/ER/75700--1

DE93 009774

Neural Network Recognition of Nuclear Power Plant Transients :

First Annual Report : Rev 1, 4/15/92 - 4/15/93

Eric B. Bartlett
Principal Investigator
Iowa State University, Ames, Iowa.

February 23, 1993.

Submitted to U.S. Department of Energy
Office of Energy Research

Special Research Grant No. DE-FG-02-92ER75700 Nuclear Engineering Research

iowa state university

Neural Network Recognition of Nuclear Power Plant Transients:

First Annual Report 4/15/92 - 4/14/93

Contributing authors:

Eric Bartlett
Richard Danofsky
John Adams
Taher AlJundi
Anujit Basu
Chalapathy Dhanwada
John Kerr
Keehoon Kim
Terry Lanc

ACKNOWLEDGMENT

This work was made possible by the gracious support of the United States Department of Energy under Special Research Grant No. DE-FG02-92ER75700, entitled "Neural Network Recognition of Nuclear Power Plant Transients". Their support does not however constitute an endorsement of the views expressed in this work.

Comments and criticisms are not only encouraged but are greatly appreciated. If you have any questions or comments please contact Eric B. Bartlett by mail at:

Prof. E. B. Bartlett
Department of Mechanical Engineering
Nuclear Engineering Program
104 Nuclear Engineering Lab.
Ames, Iowa 50011-2230

by phone at (515) 294-1828, or by FAX at (515) 294-7224.

Thank you.

Eric B. Bartlett
Ames, Iowa
February 23, 1993.

Contents

NEURAL NETWORK RECOGNITION OF NUCLEAR POWER PLANT TRANSIENTS

1 Abstract

The objective of this report is to describe results obtained during the first year of funding that will lead to the development of an artificial neural network (ANN) fault - diagnostic system for the real - time classification of operational transients at nuclear power plants. The ultimate goal of this three-year project is to design, build, and test a prototype diagnostic adviser for use in the control room or technical support center at Duane Arnold Energy Center (DAEC); such a prototype could be integrated into the plant process computer or safety - parameter display system. The adviser could then warn and inform plant operators and engineers of plant component failures in a timely manner. This report describes the work accomplished in the first of three scheduled years for the project. Included herein is a summary of the first year's results as well as individual descriptions of each of the major topics undertaken by the researchers. Also included are reprints of the articles written under this funding as well as those that were published during the funded period.

2 Work Performed During the Period

2.1 Objective of Work

The objective of this research project is to develop an ANN fault-diagnostic system for the real - time detection and classification of nuclear power plant transients and component failures. The ultimate goal is to design, build, and test a prototype diagnostic adviser for use in the control room or technical support center at the Duane Arnold Energy Center (DAEC).

The significance of the proposed work is the potential improvement in nuclear power plant operational safety realized through the timely warning and diagnosis of abnormal plant operating conditions. Another technical significance of the research is the advancement of the theory of ANN techniques and their application to fault diagnosis as well as insights and advances in the areas of information prioritization and accident mitigation and management. These advances can be applied to both boiling - water reactors (BWRs) and pressurized-water reactors (PWRs) as well as any other industrial process where accident avoidance is paramount. Quicker accident diagnostics and the resultant improvement in mitigation efforts will be of great value to nuclear plant safety.

The general aim of this research is twofold: to investigate and to educate in the fields of nuclear engineering and artificial intelligence applications.

2.2 Summary of Results

The following is a brief list of accomplishments for the reporting period.

1. More than 40 transient scenarios have been collected at the DAEC training simulator. A list of these transients is given in Table 3.

2. An error analysis method has been developed and implemented on a set of ten transient scenarios obtained from San Onofre (James and Rogers 1992). This method provides an estimated level of confidence in the transient diagnosis. A complete description of this work is given in Section 4.6.

3. An input importance analysis method has been developed. This approach is actually a combination of methods using information theory (Bartlett 1992b: Section 11.2, Hyvarinen 1970, Kullback 1959, Shannon 1971, Watanabe 1969) and ANN derivatives (Bartlett and Basu 1991, Lanc 1991). The approach helps

determine the relative importance of individual plant variables for diagnosing specific sets of nuclear power plant transients (see Section 4.7.1 and 10.10).

4. Time-series information has been used for very early diagnosis of transient existence – other networks are then called to determine the specific transient. Analysis of the temporal information given by a single variable can yield much information (Bartlett 1992a). This approach may be very effective at providing very small transient diagnosis times (see Sections 4.3 & 10.1). We believe that this approach will be best suited for use in the root network which indicates if a transient is in progress but does not necessarily provide a diagnosis as to which particular transient is occurring; this job will be left to other ANNs that are specifically designed for that purpose.

5. Similar Transients are also an area of concern. Sometimes two or more transients can exhibit very similar instrument responses. Traditional expert systems, therefore, have difficulty distinguishing them. Neural networks may not (see Section 4.1).

6. We are analyzing backpropagation speed-up techniques. Increased learning rates are very important when training large and complex data sets as in nuclear power plant diagnoses. Many researchers have investigated backpropagation learning enhancement techniques. We have implemented a variety of these techniques. Section 4.5 describes our work in this area.

7. Dynamic Node Architecture Learning with Backpropagation has a very high probability of being useful. This work is a re-creation of Bartlett's Ph.D. for a boiling-water reactor using backpropagation ANNs. Section 4.2 describes our work in this area.

8. A broad - based adviser has been developed that is capable of diagnosing 23 distinct transients. This ANNs in this adviser were trained using data for 41 transient scenarios collected from the DAEC simulator. Section 4.4 describes the work leading to the development of the adviser. This represents a big step forward from the previously developed advisers (described in Sections 4.2, 4.6, 4.7.1, 4.7.2 and 4.7.4) which were basically used to demonstrate the feasibility of individual methodologies being developed. This adviser has given us great insight into the possible design structure of the final diagnostic system, and has made the researchers capable of evolving a complicated ANN solution of the kind required to bring the final diagnostic adviser to fruition.

9. Educational Goals Attained

   - Two Masters degree completed (May 1992: T. L. Lanc & August 1992: A. Basu)
   - One Masters to be completed December 1992 (K. Kim)
   - Four students passed the written Ph.D. qualifying exam. (T. AlJundi, A. Basu, C. Dhanwada, K. Kim)

10. Significant progress has been made toward the completion of the first year's goals set out in the original research proposal – see Section 3.1.

2.3 Time Devoted by Principal Investigator

The PI has devoted 25% of his time to this project. This will continue till the scheduled completion of the project.

3 Detailed Description of Research Work

3.1 Effect of Performed Work on Research Plan and Timetable

It was originally proposed to implement the approach of ANN analysis demonstrated in Bartlett 1990 and Bartlett and Uhrig 1992 to the nuclear power plant accident diagnostic problem. The tasks involved were separated into three phases that would require approximately one year each. These phases were outlined in the originally proposed research timetable as follows. (See Bartlett 1992c for revised timetable for years II and III.)

Originally Proposed Timetable

Year I

1. Specification of accident list
2. Selection of variables for plant process
3. Collection of simulator data
4. Preliminary demonstration and enhancement of network training
5. Preliminary design of diagnostic system

Year II

1. Collection of data for plant operational transients
2. Secondary demonstration and enhancement of network training
3. Secondary design of diagnostic system

Year III

1. Final design of diagnostic system

2. Final training of diagnostic system

3. Testing of final diagnostic system

Significant progress toward these goals has been accomplished during this first funding period. Below is a brief discussion of the milestones accomplished and their relationship to the originally proposed goals.

1. Specification of accident list

After careful analysis and discussions with DAEC staff, a list of 85 transient scenarios has been identified as most important during the initial stages of this research. These scenarios include two broad categories of operational transients. In the first category are the more potentially dangerous scenarios. These are most likely to be fast acting and initiated by large component failures such as a recirculation loop rupture. The second category contains the presumably more frequent but less severe scenarios. Typical scenarios in this category include, for example, small steam leakage in a steam tunnel. Table 1 shows the working list of accidents of interest. We have already collected more than 45 of these scenarios and are well on our way to defining lists for secondary component failures and accidents that exhibit similar symptoms.

2. Selection of variables for plant process

The list of variables being used for training of the ANN diagnostic adviser can be found in Table 2. After careful analysis and discussions with DAEC staff, this variable set was determined to be the most likely first set of reliable data.

Table 1: Transient Scenarios of Initial Interest

Number	Accident Designation	Accident Description
1	AD02	Automatic depressurization system channel fail to initiate
2	AD05	Serious automatic depressurization system actuation
3	CS02	Core spray injection line break, headers A & B
4	CS03	Core spray inadvertent initiation
5	CU01	Reactor water clean-up heat -exchanger tube leak
6	CU07	Reactor water clean-up leak in heat -exchanger room
7	CU09	Reactor water clean-up regenerative heat -exchanger tube leak
8	CU10	Reactor water clean-up coolant leakage outside the primary containment
9	ED01	Loss of offsite power
10	EG01	Main generator trip, primary lockout & backup lockout
11	FW01	Condenser hotwell level controller failure
12	FW02	Condensate pump trip
13	FW04	Condensate filter demineralizer resin injection
14	FW05	Feedwater hydrogen controller failure
16	FW06	Feedwater heater drain valve fails
15	FW07	Feedwater heater dump valve fails
17	FW08	Feedwater heater tube leak
18	FW09	Reactor feedwater pump trip
19	FW12	Feedwater regulator valve controller failure
20	FW13	Master and feedwater regulator valve controller oscillation
21	FW14	Feedwater flow transmitter failure
22	FW17	Main feedwater line break inside primary containment
23	FW18	Main feedwater line break outside primary containment
24	HP01	High-pressure core injection inadvertent initiation
25	HP05	High-pressure core injection steam supply line break (High-pressure core injection room)
26	HP08	High-pressure core injection steam supply line break (Torus room)
27	IA01	Loss of instrumentation air
28	IA02	Instrumentation air header leaks
29	ID01	Drywell leakage
30	ID04	High-pressure core injection room leakage
31	ID05	Reactor core isolation cooling room leakage
32	ID06	Southeast corner room leakage
33	ID07	Northwest corner room leakage
34	ID08	Torus room leakage
35	MC01	Main circulation water pump trip

Table 1: . . . continued

Number	Accident Designation	Accident Description
36	MC02	Main condenser tube blockage
37	MC03	Cooling tower fans trip
38	MC04	Main condenser air inleakage
39	MC07	Main condenser tube leakage
40	MC08	Cooling tower riser pipe break
41	MS02	Steam leak inside primary containment
42	MS03	Main steamline rupture inside primary containment
43	MS04	Main steamline rupture outside primary containment
44	MS06	Main steam isolation valve fails close
45	MS08	Steam leakage in steam tunnel
46	MS14	Loss of extraction steam to feedwater heater
47	MS19	Spurious group 1 isolation
48	MS21	Spurious group 2 isolation
49	MS23	Spurious group 3 isolation
50	MS25	Spurious group 4 isolation
51	MS27	Spurious group 5 isolation
52	MS29	Spurious group 6A isolation (reactor core isolation cooling)
53	MS30	Spurious group 6B isolation (high-pressure core injection)
54	MS32	Spurious group 7 isolation
55	NM11	Recirculation flow unit fails
56	RC01	Reactor core isolation cooling system failure
57	RD11	Control rod drive hydraulic pump trip
58	RD13	Loss of air pressure to control rod drive HCL's
59	RD14	SCRAM discharge volume level high
60	RH01	Residual heat removal pump trip
61	RH02	Residual heat removal heat exchanger tube leak
62	RH03	Residual heat removal pump discharge line break
63	RH05	Residual heat removal/suppression pool suction line blockage
64	RH06	Residual heat removal heat exchanger B/P valve fails
65	RP03	Spurious SCRAM with operator action
66	RP05	Reactor protection system SCRAM circuit failure (Anticipated transient without SCRAM)
67	RR05	Recirculation pump shaft seizure
68	RR06	Recirculation motor generator drive motor breaker trip
69	RR07	Recirculation motor generator field breaker trip
70	RR08	Recirculation pump high vibration
71	RR10	Recirculation pump speed feedback signal failure
72	RR11	Recirculation pump seal failure
73	RR12	Recirculation motor generator oil pump failure

Table 1: . . . continued

Number	Accident Designation	Accident Description
65	RP03	Spurious SCRAM with operator action
74	RR15	Recirculation loop rupture (design basis loss of coolant accident 100%)
75	RR16	Recirculation pump RPT breaker trip
76	RR17	Recirculation motor generator flow controller fails
77	RR18	Recirculation motor generator scoop tube oscillations
78	RR20	Narrow range level transmitter failure
79	RR30	Coolant leakage inside primary containment
80	RR32	Lo-lo-lo level switch failure
81	RX01	Fuel cladding failure
82	SL01	Standby liquid pump trip
83	TC01	Main turbine trip
84	TC02	Electro-hydraulic control failures
85	SW19	River water supply pump trip

3. Collection of simulator data

Table 3 includes a list of the collected scenarios. A description of each of these is included in Section 9. Notice that many of the transients have been simulated more than once. This is because either a different initial condition, a different operating power level, a different severity level, or a secondary malfunction was simulated along with the original malfunction. More simulator work is needed in order to complete the list of transients of initial interest Table 1. This work will proceed into the second year of the project.

4. Preliminary demonstration and enhancement of network training

This goal has been addressed by the investigations of Basu (see Section 4.2 & 4.7.2), Dhanwada (see Sections 4.3 & 4.4), Kim (see Section 4.5, 4.6 & 4.7.4) and Lanc (see Section 4.7.1). It is important to determine the most effective method for ANN training using small numbers of accidents. In this way the larger sets of

Table 2: Initial List of Important Plant Variables Variable

	Variable Desig.	Description	Min. Value	Max. Value	Variable Unit
1	A041	Local power range monitor 16-25 flux level B	0.0	125.0	% power
2	A091	Source range monitor channel B	0.0	00.0	%
3	B000	Average power range monitor A Flux level	0.0	125.0	% power
4	B012	Reactor total core flow	0.0	60.0	Mlb/hr
5	B013	Reactor core pressure-differential	0.0	30.0	psid
6	B014	Control rod drive system flow	0.0	0.025	Mlb/hr
7	B015	Reactor feedwater loop A flow	0.0	4.0	Mlb/hr
8	B016	Reactor feedwater loop B flow	0.0	4.0	Mlb/hr
9	B017	Cleanup system flow	0.0	0.07691	Mlb/hr
10	B022	Total steam flow	0.0	8.0	Mlb/hr
11	B023	Cleanup system inlet temperature	0.0	755.0	°F
12	B024	Cleanup system outlet temperature	0.0	600.0	°F
13	B026	Recirculation loop A1 drive flow	0.0	15.1	Mlb/hr
14	B028	Recirculation loop B1 drive flow	0.0	15.1	Mlb/hr
15	B030	Reactor feedwater channel A1 temperature	280.0	430.0	°F
16	B032	Reactor feedwater channel B1 temperature	280.0	430.0	°F
17	B034	Recirculation loop A1 inlet temperature	260.0	580.0	°F
18	B036	Recirculation loop B1 inlet temperature	260.0	580.0	°F
19	B038	Recirculation A wide range temperature	50.4	789.6	°F
20	B039	Recirculation B wide range temperature	50.4	789.6	°F
21	B061	Reactor coolant total jet pumps 1-8 flow B	0.0	36.7	Mlb/hr
22	B062	Reactor coolant total jet pumps 9-16 flow A	0.0	36.7	Mlb/hr
23	B063	Reactor coolant total outlet steam flow A	0.0	2.0	Mlb/hr
24	B064	Reactor coolant total outlet steam flow B	0.0	2.0	Mlb/hr
25	B065	Reactor coolant total outlet steam flow C	0.0	2.0	Mlb/hr

Table 2: . . . continued

	Variable Desig.	Description	Min. Value	Max. Value	Variable Unit
26	B066	Reactor coolant total outlet steam flow D	0.0	2.0	Mlb/hr
27	B079	Reactor recirculation pump A motor vibration	0.0	10.0	MILS
28	B080	Reactor recirculation pump B motor vibration	0.0	10.0	MILS
29	B083	Control rod drive cooling-water differential pressure	0.0	500.0	dpsi
30	B084	Control rod drive cooling-water differential pressure	0.0	60.0	dpsi
31	B085	Torus air temperature #1	0.0	500.0	°F
32	B086	Torus air temperature #2	0.0	500.0	°F
33	B087	Torus air temperature #3	0.0	500.0	°F
34	B088	Torus air temperature #4	0.0	500.0	°F
35	B089	Drywell temperature azmuth 0 elevation 750	0.0	500.0	°F
36	B090	Drywell temperature azmuth 245 elevation 750	0.0	500.0	°F
37	B091	Drywell temperature azmuth 90 elevation 765	0.0	500.0	°F
38	B092	Drywell temperature azmuth 270 elevation 765	0.0	500.0	°F
39	B093	Drywell temperature azmuth 270 elevation 765	0.0	500.0	°F
40	B094	Drywell temperature azmuth 180 elevation 780	0.0	500.0	°F
41	B095	Drywell temperature azmuth 270 elevation 830	0.0	500.0	°F
42	B096	Drywell temperature center elevation 750	0.0	500.0	°F
43	B098	Torus water temperature	0.0	752.0	°F
44	B099	Torus water temperature	0.0	752.0	°F
45	B103	Drywell pressure	0.0	100.0	psia
46	B104	Torus pressure	0.0	100.0	psia
47	B105	Torus water level	-10.0	10.0	inch
48	B120	Torus radiation monitor A	-1.0	100.0	%
49	B121	Torus radiation monitor B	-1.0	100.0	%
50	B122	Reactor water level	158.0	218.0	inch
51	B124	Reactor water level	158.0	218.0	inch
52	B1257	Fuel zone level indication	-153.0	218.0	inch
53	B126	Reactor water level	158.0	458.0	inch
54	B127	Reactor vessel pressure	0.0	1200.0	psig

Table 2: . . . continued

	Variable Desig.	Description	Min. Value	Max. Value	Variable Unit
55	B128	Reactor vessel pressure	0.0	1200.0	psig
56	B129	Reactor vessel pressure	0.0	1500.0	psig
57	B130	Reactor vessel pressure	0.0	1500.0	psig
58	B137	Torus water level	1.5	16.0	ft
59	B138	Torus water level	1.5	16.0	ft
60	B150	Core spray A flow	-1767.8	5000.0	gpm
61	B151	Core spray B flow	-1767.8	5000.0	gpm
62	B160	Reactor core isolation cooling flow	-62.5	500.0	gpm
63	B161	High-pressure core injection flow	-437.5	3500.0	gpm
64	B162	Residual heat removal A flow	-75.0	15000.0	gpm
65	B163	Residual heat removal B flow	-75.0	150.0	gpm
66	B164	Drywell radiation monitor A	-1.0	100.0	%
67	B165	Drywell radiation monitor B	0.0	100.0	%
68	B166	Post-treat activity	0.0	100.0	%
69	B168	Pretreat activity	0.0	100.0	%
70	B171	Analyzer A O_2 concentration	-1.25	10.0	%
71	B172	Analyzer A H_2 concentration	-1.25	10.0	%
72	B173	Analyzer B O_2 concentration	-1.25	10.0	%
73	B174	Analyzer B H_2 concentration	-1.25	10.0	%
74	B180	Clean-up system flow	0.0	200.0	gpm
75	B196	Reactor water level-fuel zone A	-153.0	218.0	inch
76	B197	Reactor water level-fuel zone B	-153.0	218.0	inch
77	B247	Turbine steam bypass	0.0	500.0	°F
78	B248	Turbine steam bypass	0.0	500.0	°F
79	E000	4160 V Switch Gear bus 1A1 A-B	0.0	5.25	KV
80	F004	Condensate pump A&B discharge pressure	0.0	600.0	psig
81	F005	Low-pressure condenser circulating water inlet temperature A	0.0	200.0	°F
82	F010	High-pressure condenser circulating water outlet temperature A	0.0	200.0	°F
83	F011	Low-pressure condenser circulating water pressure differential A	0.0	10.0	dpsi
84	F015	Circulating water pump A&B discharge pressure	0.0	100.0	psig
85	F018	Cooling tower A discharge water temperature	0.0	752.0	°F
86	F019	Cooling tower B discharge water temperature	0.0	752.0	°F
87	F040	1P-1A reactor feed pump suction pressure	0.0	600.0	psig

Table 2: . . . continued

	Variable Desig.	Description	Min. Value	Max. Value	Variable Unit
88	F041	1P-1B reactor feed pump suction pressure	0.0	600.0	psig
89	F042	1P-1A reactor feed pump discharge pressure	0.0	2000.0	psig
90	F043	1P-1B reactor feed pump discharge	0.0	2000.0	psig
91	F044	Condensate total flow	0.0	8.0	Mlb/hr
92	F045	Condensate makeup flow	-10.0	100.0	Klb/H
93	F046	Condensate rejection flow	0.0	50.0	Klb/H
94	F094	Feedwater final pressure	0.0	2000.0	psig
95	G001	Generator gross watts	0.0	720.0	MWE
96	T039	Low-pressure condenser pressure	0.0	30.0	inHg
97	T040	High-pressure condenser pressure	0.0	30.0	inHg

accidents can be trained efficiently without wasting time.

5. Preliminary design of diagnostic system

A preliminary diagnostic adviser has been developed that can diagnose 23 distinct transients based on the data of 41 transient scenarios collected from the DAEC training simulator. This adviser, which is the most broad-based effort towards the development of the final diagnostic adviser, will help lead the research towards the ultimate form and structure of the final adviser. A single network was trained to detect a normal or abnormal condition at any instance of time. This "root" network activates the other diagnostic networks if it determines that the plant is in any condition other than normal. Five other networks then together produce the five-bit boolean that classifies the abnormal condition. This approach, together with the other methodologies developed as part of this research effort, has given us a preliminary design of the final adviser.

The final diagnostic system will be a complex set of independently trained interconnected networks. The approach will be to have a "root" network that diagnoses normal or abnormal conditions. Once the abnormal condition is recognized by the

Table 3: Transient Scenarios collected

Number	Accident Designation	Accident Description
1	CU10	Reactor water clean-up coolant leakage
2	CU10	Reactor water clean-up coolant leakage with automatic group 5 isolation malfunction
3	FW02	Condensate pump trip
4	FW04	Condensate Filter demineralizer resin injection
5	FW09	Reactor feedwater pump trip
6	FW12	Feedwater regulator valve controller failure
7	FW17	Main feedwater line break inside primary containment (100% severity)
8	FW17	Main feedwater line break inside primary containment (60% severity)
9	FW17	Main feedwater line break inside primary containment (30% severity)
10	FW18	Main feedwater line break outside primary containment
11	HP05	High-pressure core injection steam supply line break (High pressure core injection room 100% severity)
12	HP05	High-pressure core injection steam supply line break (High pressure core injection room 60% severity)
13	HP05	High-pressure core injection steam supply line break (High pressure core injection room 30% severity)
14	HP08	High-pressure core injection steam supply line break (Torus room)
15	MC01	Main circulation water pump trip
16	MS02	Steam leak inside primary containment
17	MS03	Main steam line rupture inside primary containment
18	MS04	Main steam line rupture outside primary containment
19	MS14	Loss of extraction steam to feedwater heater
20	MS19	Spurious group 1 isolation
21	MS32	Spurious group 7 isolation
22	RP03	Spurious SCRAM with operator action
23	RP05	Reactor protection system SCRAM circuit failure (ATWS)
24	RR10	Recirculation pump speed feedback signal failure
25	RR15	Recirculation loop rupture (design basis Loss of coolant accident 100% severity)
26	RR15	Recirculation loop rupture (design basis Loss of coolant accident 60% severity)
27	RR15	Recirculation loop rupture (design basis Loss of coolant accident 30% severity)
28	RR30	Coolant leakage inside primary containment
29	RX01	Fuel cladding failure

Table 3: . . . continued

Number	Accident Designation	Accident Description
30	TC02	Electro-hydraulic control failures
31	IC14	Spurrious SCRAMat 100% power beginning of cycle
32	IC20	Spurrious SCRAM at 100% power end of cycle
33	IC22	Spurrious SCRAM at 25% power beginning of cycle
34	IC23	Spurrious SCRAM at 75% power beginning of cycle
35	IC24	Spurrious SCRAM at 100% power middle of cycle ??

root network, a category network will determine the category of the transient. For example, the transient may be a positive reactivity insertion or a loss of coolant event. Next, the specific transient will be diagnosed based on the outputs of the root and category networks. The specific accident network will use the information from the other more basel networks, i.e., that an abnormality has occurred and that the abnormality is of the loss of coolant type to help it make its decision as to the particular type of transient, i.e., main steam line break. Next will be the error prediction networks. These networks will provide a figure of merit on the diagnosis (see Section 4.6). The approach is similar in architecture to a tree (see Figure 1) and allows for the splitting of the overall diagnostic effort into small, manageable steps. Each of the networks will be trained separately, and this will significantly reduce training computer time. Each of the networks is not required to have the same number or identical input variables which greatly reduces the complexity of data collection.

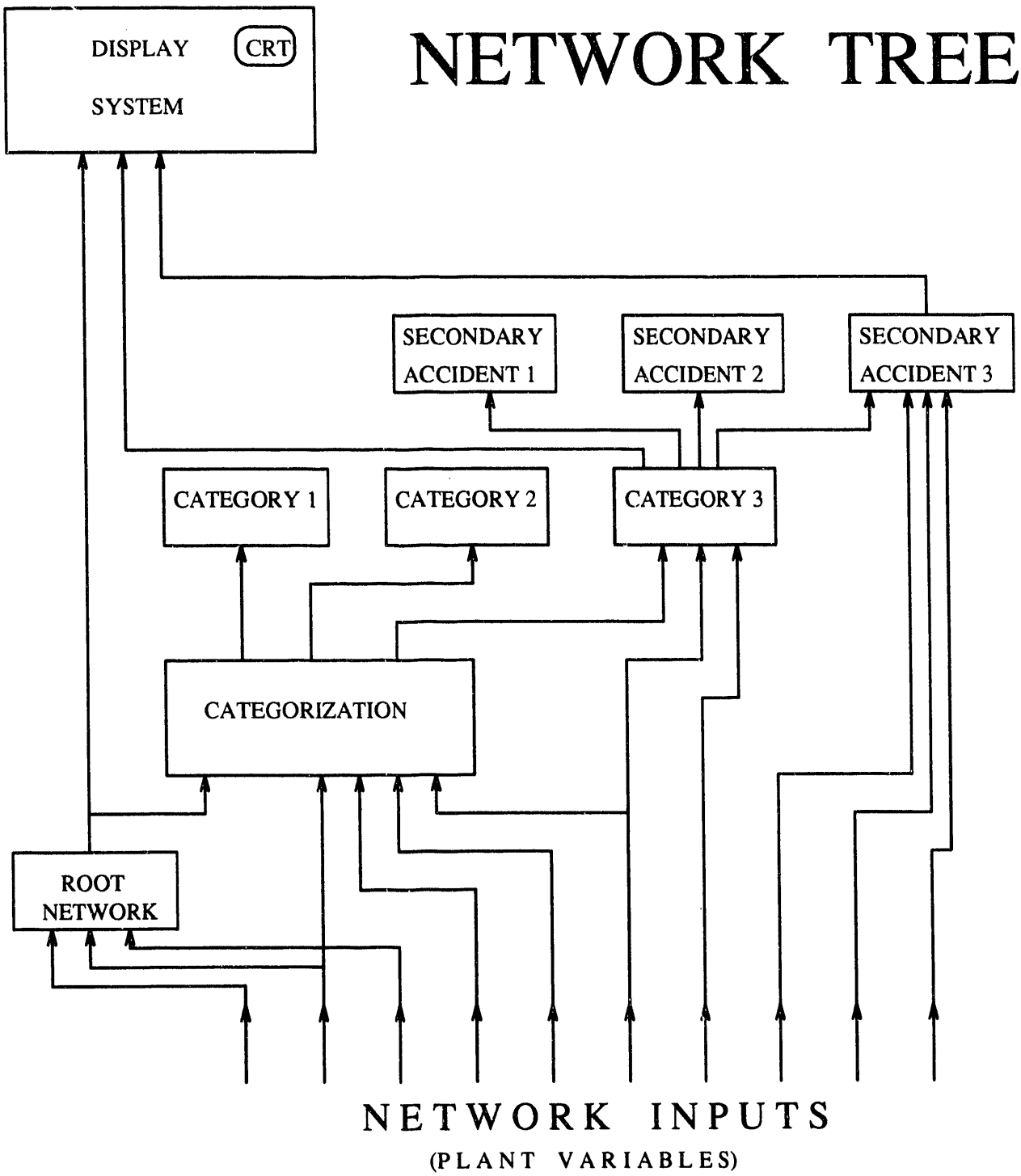

Figure 1: The tentative tree structure of the final diagnostic adviser.

4 Summary of Individual Projects

This section contains summaries of the work undertaken by each of the graduate students working on this project.

4.1 Diagnosing Similar Transients Using Artificial Neural Networks

by Taher Aljundi

4.1.1 Introduction

When an operational transient takes place at a nuclear power plant, many measuring instruments will detect the changes in the plant's conditions. Although there is a wide variety of these transients, some, because of their nature, will cause part of the measuring instruments to respond in a similar way. Consequently, groups of these instruments will show similar responses for a variety of transients. A typical example is a small coolant leak compared to a big steam leak in the torus room at a boiling water reactor power plant. In this case, responses of temperature and pressure gauges in the vicinity of the leak will be similar for both scenarios. However, since the steam is radioactive while the coolant is not, radiation monitors will tell which transient is taking place.

Previous work done here at Iowa State University (ISU) showed that ANNs have the ability to diagnose several different transients at a nuclear power plant in a very short time (Bartlett 1992, Lanc 1991, Bartlett, Basu, and Kim 1992). The next step in this area will be to further explore the capabilities of these networks in recognizing transients that are symptomatically similar.

4.1.2 Method

We have applied information theory to select the variables that have the highest diagnostic values. This method has reduced the confusion in the network by eliminating redundant variables and variables with low information content.

Training a neural network to diagnose similar transients is a very difficult task, especially when the number of these transients increases. Applying information theory

to select the important variables and other techniques to find the optimal network architecture will ease the training problem and improve the generalization ability of the network. Determining the appropriate network architecture for a given problem before training is a very important task. Too many nodes in a network will significantly increase the training time and will make it easy for the network to fit the noise of the training data, thus failing to generalize (Weigend, Rumelhart, and Huberman 1991). Different techniques for determining the optimal network architecture have been investigated and a dynamic node scheme for the backpropagation neural network has been developed here at ISU (Bartlett and Basu 1991, Bartlett and Uhrig 1991b). These techniques along with the selection of the plant's variables according to their information contents will be applied to diagnose the similar transients.

The number of plant process variables being collected from the Duane Arnold Energy Center (DAEC) simulator is 97. This number can be increased to 300 (or more) variables if the information content of the variables is not enough to diagnose all of the similar transients of interest. However, discussions with the simulator supervisor at DAEC led us to believe that 200 variables should be enough to diagnose all important similar transients in the plant.

4.1.3 Objective

The diagnostics adviser will consist of a root network connected to an array of branch networks. One of these branches will be responsible for diagnosing transients that are symptomatically similar. Special attention will be paid to these transients because usually they are more difficult to identify and diagnose. Also a good benchmark for the diagnostics adviser will be to test its performance in identifying these transients.

4.1.4 Preliminary Results

The results below demonstrate the advantage of continuous monitoring, which is the ability to detect malfunctions in a relatively short time.

The raw data that generated these results were collected without any addition of new instrumentations to the plant. Readings were taken from currently used measuring instruments in the plant. In other words, this system can be implemented in the plant at a lower cost than other alternatives such as some expert systems that require adding new instruments or periodic hand-held monitors (Frarey, Wilson, Peterson and Bartlett 1984).

Figure 2 shows the network respond toward classifying two similar transients. These transients were chosen by comparing the sequence of events that take place as a consequence of the main transient. Description of the transients can be found in Section 9 (Description of Collected Malfunction Data).

Information theory was applied to choose the most valuable 25 variables among the 97 total number of variables that were trended. The oscillations at the beginning of each accident means that the network has detected instability in the system, but it is not sure which transient is in progress. When the RMS approaches zero, it means that the network has decided which particular transient is in progress.

The network was able to recognize the first transient in a very short time (less than 4 seconds). For the second transient, however, the network took about ten seconds to recognize it.

4.1.5 Conclusion and Future Work

The results show the feasibility of using ANNs technology as a diagnostic tool for similar transients in nuclear power plants. The next step is to further explore the capabilities of these networks in diagnosing more of these similar transients.

We are also investigating the classification of these transients in the frequency domain. The way in which the monitored variables change during a transient can be described in the frequency domain using the Discrete Fourier Transform (DFT). The frequency spectrum will be monitored on-line in one-second time slices for all the variables. The changes in the frequency spectrum of the monitored variables could lead to significant information relative to the diagnosis. Certain aspects of the changes in the plant's conditions are more obvious in the frequency domain than they are in the time domain. The DFT will magnify the importance of the variables that are disturbed the most during the transient. The most disturbed variables usually represent the measuring instruments in the vicinity of the transient, and hence are directly affected by the change in the plant's conditions. This method might be useful to determine which variables have the highest diagnostics values. The transients will be classified when certain frequency components appear or disappear from the spectrum.

Because some variables oscillate during both the transient and the normal operating conditions, monitoring their oscillations in the time domain gives misleading information to the network. This is because changes in the time domain will give such variables a high diagnostic value, while actually they are not that important from a diagnostic point of view. The frequency domain, on the other hand, monitors the changes in the frequency components of these variables, and the power spectrum of the variables will measure the strength of the oscillations. This way, the variables that oscillate with the same frequency before and after the transient's onset will be disregarded because they carry no information about the transient. Classifying the transients in the frequency domain might also be immune against noisy data since the DFT allows us to filter the spectrum and reject very high frequencies which usually correspond to noise.

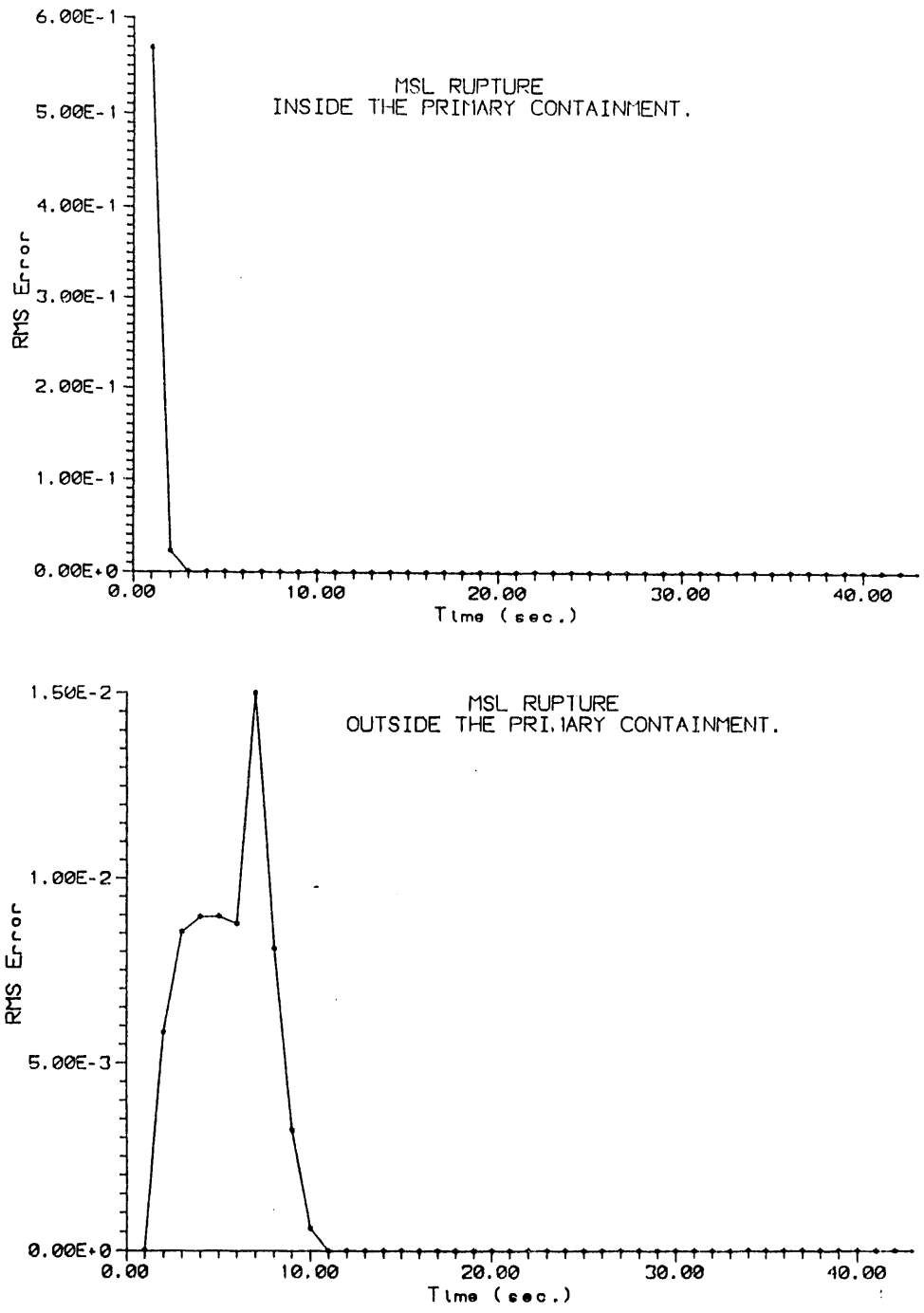

Figure 2: The RMS error between the output of the network and the desired output for a main steam line break inside the primary containment (above) and outside the primary containment (below).

4.2 A Nuclear Power Plant Status Diagnostic Adviser Using A Dynamic Node Architecture Backpropagation Neural Network

by Anujit Basu

4.2.1 Introduction

This section describes the development of an artificial neural network (ANN) based nuclear power plant status diagnostic adviser. This adviser is used to detect and distinguish seven distinct transients and normal operating conditions. The adviser indicates if the plant is in a normal operating condition, and if not, which of these seven transients it's going through. The transients were simulated on the full scale control room simulator at the Duane Arnold Energy Center (DAEC) owned by Iowa Electric Light and Power Company at Palo, IA. The goal of this ongoing project is to install and test a prototype adviser in the technical support center at DAEC.

Every transient in a nuclear power plant is unique in the changes it causes in the values of the various plant variables. So, by looking at the patterns in the variables during a transient, it is possible to recognize the transient. ANNs are able to handle this task because they are very good pattern classifiers.

The backpropagation learning algorithm (Hecht-Nielsen 1989) was used for this work. In such neural networks, the number of nodes in the input and output layers are set by the problem at hand (Caudill, 1988). The users' only real choice in network design is the number of nodes in the hidden layer. Hidden layers are the layers in between the input and output layers and are so called as they are isolated from the outside environment. It is crucial to have the appropriate number of nodes in the hidden layer because too many or too few hidden nodes will lead to unsatisfactory training and post- training performance. Backpropagation require a prespecified network architecture (or size) before training is initiated because this architecture se-

lection is going to affect the learning speed and generalization abilities of the network. Thus the effectiveness of a selected architecture can be assessed only after training (Bartlett & Basu 1991). For large and complex problems of the kind involved in nuclear power plant diagnostics, the search for an appropriate ANN architecture would prove to be an extremely time consuming affair involving a lot of guesswork. To eliminate this guesswork, it is imperative to utilize a systematic method to arrive at a proper network architecture. The Dynamic Node Architecture (DNA) scheme for the backpropagation neural network (Bartlett & Basu 1991) was developed to do this.

4.2.2 Neural Networks

Most pattern recognition problems try to find a functional relation between the inputs and the outputs. This is called a mapping. A mapping M given by (Bartlett & Basu 1991)

$$M(inputs) = outputs \tag{1}$$

can be modeled by an artificial neural network. The inputs and the outputs are either vectors or scalars.

This work used ANNs with only one hidden layer, so the networks had a total of three layers, the input, hidden, and output layers. To be consistent, in this section subscripts i, j and k shall refer to the input, hidden, and output nodes respectively, and the subscript n shall refer to the nth training pattern. Let $imax$, $jmax$ and $kmax$ be the number of nodes in the three layers. Also let x_i, x_j and x_k be the outputs of the ith input, jth hidden, and kth output layer nodes respectively. It is to be noted that the hidden and output nodes are the active nodes. The input layer nodes are inactive in that their output is equal to their input. Thus, x_i is also the input to the ith input node. The input of the jth node in the hidden layer is the weighted sum of

the outputs of the input layer nodes. Mathematically,

$$sum_{j,n} = \sum_{i=1}^{imax} (x_{i,n} * a_{j,i}) \tag{2}$$

where $x_{i,n}$ is the output of the ith input node for the nth training pattern and $a_{j,i}$ is the weight connecting the jth hidden node and the ith input node. The transfer function used in this work was the sigmoid function. The output of the jth hidden node, $x_{j,n}$, after passing the input through the transfer function, is given by

$$x_{j,n} = \frac{1}{\{1 + exp(-sum_{j,n})\}} \tag{3}$$

where $sum_{j,n}$ is given by Eq.(2). Similarly, the input to the kth output node is

$$sum_{k,n} = \sum_{j=1}^{jmax} (x_{j,n} * b_{k,j}) \tag{4}$$

where $x_{j,n}$ is given by Eq.(3), and $b_{k,j}$ is the weight connecting the kth output layer node and the jth hidden layer node. As with the hidden layer, the output of the kth output node, $x_{k,n}$, is given by

$$x_{k,n} = \frac{1}{\{1 + exp(-sum_{k,n})\}} \tag{5}$$

By the above computations, the output of the network for the given weight set can be calculated for a given pattern. A measure of the performance of a neural network is the Root Mean Square (RMS) error in the output of the network. This error is a function of the existing weight set. It can be defined as

$$RMS\ error = E(a_{j,i}, b_{k,j}) = \{\frac{1}{N * kmax} \sum_{n=1}^{N} [\sum_{k=1}^{kmax} (z_{k,n} - x_{k,n})^2]\}^{1/2} \tag{6}$$

where $z_{k,n}$ is the expected output of the kth output node for the nth training pattern, i.e. this is the value we want $x_{k,n}$ to reach. The N in the above equation is the total number of patterns in the training set.

Backpropagation (Hecht-Nielsen 1989) is by far the most commonly used neural network paradigm. Its popularity arises from its simple architecture and easy to

understand learning process. The backpropagation scheme consists of two major passes through the network (Lippmann 1987). The first is the forward activation flow, which gives the output of the network for the given input and the current set of weights. The second sweep is the backward error propagation that calculates the error in the output of all the active nodes. This process of assigning errors to all the active nodes allows us to apply the Delta rule to the output nodes and the Generalized Delta rule to the hidden nodes to arrive at a new set of weights that is nearer to the ideal weight set that we are seeking. This process is continued until the RMS error (Eq. 6) falls below a prespecified value.

The weights specifying the trained network are then used to solve the particular problem. The network is presented with input patterns that it may or may not have encountered during the training process. This process using the trained ANN is called the recall.

4.2.3 Dynamic Node Architecture

The generalization capabilities of ANNs are strongly dependent on their architecture. Theoretically, it might be possible to make the RMS error very low over a training set with a network of some arbitrary size (Hecht-Nielsen 1989), but such an approach does not guarantee generalization of features. Generalization is "the ability to quantitatively estimate the characteristics of a phenomenon never encountered before based on its similarities with things already known" (Bartlett & Basu 1991). Generalization is very important since the ultimate objective of the whole exercise is to minimize the recall error. But the recall set is rarely known during training for real-world problems. A good generalizer will be able to weed out the general characteristics from the training set and will correctly classify unseen (not used during training) patterns without being involved or led astray with the specifics of the training data set. This effective generalization requires that the network has just the right amount of ability

to distinguish details. ANNs derive this ability from their nodes and weights. Thus it is imperative to have an ANN with the appropriate architecture to solve a problem. Too few nodes will result in a network too slow to train; in fact it might even be untrainable. A network with too many nodes, on the other hand, will memorize the training set and will have a poor recall performance (Caudill 1988).

In the problem studied here involving nuclear power plant diagnostics, the formulation of the training set was in our hands. We attempted to reduce the recall error using as few of the recall patterns as possible during training. When we found that the recall error got very high in a particular region, the pattern corresponding to the highest error in the region was added to the training set. In this way, the training set was being forced into a fairly accurate representation of the recall set. Good generalization helped keep the training set size as small as possible.

The problem of network size can be likened to finding the interpolating polynomial between various points (Bartlett & Basu 1991). A reasonably small degree of the polynomial will give a fairly smooth curve. But as the degree of the polynomial increases, the curve starts oscillating between the interpolating points and becomes chaotic. Similar is the case with an ANN with too many nodes. But in the nuclear power plant diagnostics problem, we are able to force the training set to be representative of the recall set. In the polynomial interpolation analogy, this is akin to being able to add those points that are away from the smooth interpolating curve, and try to find a new interpolation. In this case, it is that much more important to be able to get the smoothest interpolating curve so that the least number of points need to be used to find the proper interpolation.

The Dynamic Node Architecture scheme works as follows. Training is initiated with as few hidden nodes as possible, which in this work was usually one node. In all likelihood, the network will be unable to learn the mapping. As the network reaches

a plateau, add one more node. Keep repeating the process till the network is able to learn the mapping (indicated by a value of RMS error smaller than a preset value). Once this is done, eliminate a node with the least importance. If this node had a near-zero importance, then we will be removing a virtually useless node (Bartlett & Basu 1991). This smaller network might now be able to learn the classification problem upon further training. This procedure of deleting nodes is continued till the network has become so small that it cannot learn the classification any more. Then start adding nodes till the network can relearn the classification. This cyclic addition and deletion of nodes leads to a final network that has the optimum architecture for the given problem. This network should give a "more general implementation of the mapping problem" (Bartlett & Basu 1991). The procedure described here can be understood from the computer simulation results given later.

4.2.4 Importance of a Node

While deleting a node, we select one with the least importance. The importance of a node is a function of the network outputs. If changes in the output of a hidden node is detrimental in deciding the output of the network more than a similar change in the output of another hidden node, it stands to reason that the former node is more important to the "dynamic functioning of the network" (Bartlett & Basu 1991) than the later node. Therefore, the importance of the jth hidden node with respect to the kth output node can be defined as (Bartlett & Basu 1991)

$$I_{(x_j|x_k)} = E[|\delta x_{k,n}/\delta x_{j,n}|] * dx_j^{max} \tag{7}$$

where E[ ] is the expectation over the entire training set and dx_j^{max} is the maximum change in the output of the jth hidden node also over the entire training set. This importance function is called the derivative importance function. The derivative in the previous equation, which is the change in output of the kth output node due to a

change in output of the jth hidden node, can be evaluated by partial differentiation of the transfer function. This gives

$$\frac{\delta x_{k,n}}{\delta x_{j,n}} = \frac{exp(-sum_{k,n}) * b_{k,j}}{[1 + exp(-sum_{k,n})]^2} \tag{8}$$

where $sum_{k,n}$ is of the form given by Eq.(5). Equation (7) gives us the partial importance of the jth hidden node with respect to the kth output node. In case of more than one hidden layer, the importance of any hidden node with respect to any output node can be found out using the chain rule. The total importance of the jth hidden node is the sum of the partial importances of that node with respect to all the output nodes. Mathematically,

$$I_{(x_j)} = \sum_{k=1}^{kmax} I_{(x_j|x_k)} \tag{9}$$

In this same way, we can define the importance of a layer as the sum of the importances of the nodes in that layer.

4.2.5 Demonstration on Simple Examples

This section demonstrates how a backpropagation neural network learns with DNA. For this purpose, three different problems are selected. The first is the Exclusive-nor problem; the second is the 8-to-1 decoder problem; and the third is a probability density function separator problem.

The Exclusive-Nor Problem

The most basic benchmark learning problem for a neural network is the exclusive-nor problem. The training data is shown in Table 4 This is a simple two-input one-output problem. The DNA training algorithm was started with one hidden node. Thus the starting architecture was 2 x 1 x 1. Table 5 shows the training history for this problem. The starting architecture is, as expected, unable to learn the problem. A second node is added. This architecture also reaches a plateau, and a third node

Table 4: Exclusive-nor training data

Pattern	Input1	Input2	Output
1	0.0	0.0	1.0
2	0.0	1.0	0.0
3	1.0	0.0	0.0
4	1.0	1.0	1.0

Table 5: Dynamic node architecture training history for the exclusive-nor problem

Arch.			RMS	Arch.			RMS	Arch.			RMS
In	Hid	Out	Error	In	Hid	Out	Error	In	Hid	Out	Error
2	1	1	0.5012	2	3	1	0.0179	2	4	1	0.0283
2	1	1	0.3872	2	4	1	0.0193	2	4	1	0.0099
2	2	1	0.3907	2	4	1	0.0099	2	3	1	0.0288
2	2	1	0.1106	2	3	1	0.0224	2	3	1	0.0098
2	3	1	0.1190	2	3	1	0.0097	2	2	1	0.0684
2	3	1	0.0338	2	2	1	0.0423	2	2	1	0.0099
2	4	1	0.0413	2	2	1	0.0099	2	1	1	0.4996
2	4	1	0.0099	2	1	1	0.4998	2	1	1	0.4922
2	3	1	0.0252	2	1	1	0.4929	2	2	1	0.4218
2	3	1	0.0098	2	2	1	0.3219	2	2	1	0.1452
2	2	1	0.0624	2	2	1	0.0623	2	3	1	0.1826
2	2	1	0.0211	2	3	1	0.0668	2	3	1	0.0164
2	3	1	0.0302	2	3	1	0.0148	2	4	1	0.0239

is added. This 2 x 3 x 1 architecture is unable to learn the problem fast enough, and a fourth node is added. The 2 x 4 x 1 architecture manages to reach an RMS error below the target of 0.01. Now, the node with the least importance is deleted. This leaves three nodes in the hidden layer. This architecture is now able to reach the target RMS error, and yet another node is deleted. We find that on further training, the 2 x 2 x 1 network is also able to learn the problem classification. The further elimination of a node renders the network with only one hidden node. This is not sufficient to learn the problem, and a node is added. This process is continued and the algorithm oscillates around the optimum architecture as can be seen from Table 2. It is also evident that 2 x 2 x 1 is the appropriate architecture for the problem.

The 8-to-1 Decoder Problem

Eight distinct patterns can be made using three booleans. This problem involves firing one of eight output nodes for each of the eight patterns. So there are three inputs and eight outputs. The training data for this problem can be found in Table 6. As with the exclusive-nor problem, training is started with one hidden layer. So the starting architecture is 3 x 1 x 8. The training history for this problem can be found in Table 7. The training process is very similar to that for the exclusive- nor problem. The target RMS error was 0.01. The network kept adding nodes till the RMS error finally fell below the target with six hidden nodes. Then nodes began to be deleted till at three hidden nodes, the network was unable to learn the problem. Following this, nodes began to be added, and the oscillations noticed in the previous example are witnessed here too. As can be seen from Table 4, the optimum architecture arrived at by the DNA scheme is 3 x 4 x 8.

The Probability Density Function Separator Problem

In this problem, an artificial neural network with DNA is taught to recognize which of two probability density functions was used to sample a set of ten numbers.

Table 6: 8-to-1 decoder training data

Pattern									
1	Inputs	0.0	0.0	0.0					
	Outputs	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	Inputs	0.0	0.0	1.0					
	Outputs	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
3	Inputs	0.0	1.0	0.0					
	Outputs	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0
4	Inputs	0.0	1.0	1.0					
	Outputs	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0
5	Inputs	1.0	0.0	0.0					
	Outputs	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
6	Inputs	1.0	0.0	1.0					
	Outputs	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
7	Inputs	1.0	1.0	0.0					
	Outputs	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0
8	Inputs	1.0	1.0	1.0					
	Outputs	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0

Table 7: Dynamic node architecture training history for the 8-to-1 decoder problem

Arch.			RMS	Arch.			RMS	Arch.			RMS
In	Hid	Out	Error	In	Hid	Out	Error	In	Hid	Out	Error
3	1	8	0.5162	3	4	8	0.1267	3	5	8	0.0622
3	1	8	0.4872	3	4	8	0.0098	3	5	8	0.0099
3	2	8	0.4216	3	3	8	0.2218	3	4	8	0.0503
3	2	8	0.2935	3	3	8	0.0834	3	4	8	0.0091
3	3	8	0.3354	3	4	8	0.1679	3	3	8	0.1148
3	3	8	0.1974	3	4	8	0.0210	3	3	8	0.0734
3	4	8	0.2192	3	5	8	0.0604	3	4	8	0.1427
3	4	8	0.0895	3	5	8	0.0098	3	4	8	0.0323
3	5	8	0.1248	3	4	8	0.0944	3	5	8	0.0910
3	5	8	0.0227	3	4	8	0.0099	3	5	8	0.0247
3	6	8	0.0518	3	3	8	0.2421	3	6	8	0.0402
3	6	8	0.0099	3	3	8	0.0828	3	6	8	0.0098
3	5	8	0.0826	3	4	8	0.1004	3	5	8	0.0826
3	5	8	0.0099	3	4	8	0.0315	3	5	8	0.0096

Table 8: Probability density function separator training data

		I	n	p	u	t	s			Out
0.388	0.099	0.327	0.892	0.394	0.036	0.505	0.571	0.383	0.454	0.0
0.628	0.996	0.487	0.628	0.953	0.873	0.365	0.917	0.899	0.666	1.0
0.892	0.099	0.394	0.034	0.695	0.619	0.223	0.011	0.126	0.020	0.0
0.996	0.628	0.873	0.480	0.970	0.953	0.766	0.365	0.666	0.425	1.0
0.099	0.034	0.619	0.091	0.350	0.695	0.316	0.223	0.020	0.120	0.0
0.628	0.480	0.953	0.614	0.850	0.970	0.830	0.766	0.425	0.658	1.0
0.034	0.091	0.695	0.227	0.580	0.350	0.375	0.316	0.120	0.503	0.0
0.480	0.614	0.970	0.769	0.943	0.850	0.863	0.830	0.658	0.918	1.0

Table 9: Dynamic node architecture training history for the probability density function separator problem

Arch.			RMS	Arch.			RMS	Arch.			RMS
In	Hid	Out	Error	In	Hid	Out	Error	In	Hid	Out	Error
10	1	1	0.4999	10	3	1	0.0662	10	3	1	0.0992
10	2	1	0.3454	10	4	1	0.5019	10	4	1	0.1492
10	2	1	0.2055	10	4	1	0.4404	10	4	1	0.0623
10	3	1	0.2819	10	5	1	0.3216	10	5	1	0.0812
10	3	1	0.1448	10	5	1	0.0422	10	5	1	0.0488
10	4	1	0.1403	10	4	1	0.0918	10	4	1	0.1008
10	4	1	0.0413	10	4	1	0.0488	10	4	1	0.0402
10	3	1	0.1304	10	3	1	0.1281	10	3	1	0.0961

Figure 3 shows the two probability density functions. If function A was used to do the sampling, then the proper network output was chosen to be 0.00; if function B were used, the output was to be 1.00. Table 8 shows the training data.

Table 9 gives the dynamic node architecture training history for this problem. The problem, as defined, gives us ten input and one output layer nodes. We start with one hidden node and very soon reach a plateau and add a second node to the hidden node. This is continued till the number of nodes becomes four. The 10 x 4 x 1 network manages to learn the problem to a sufficient level of accuracy. The target RMS error for this problem was set at 0.05. As this is reached, the network computes the importance of each of the four hidden nodes and deletes the one with the least nodal importance. This smaller network, 10 x 3 x 1, is unable to reach an RMS below

FUNCTION A : $f(X) = 0.5\pi Cos(\pi X/2)$

OUTPUT : 0.00

FUNCTION B : $f(X) = 4X^3$

OUTPUT : 1.00

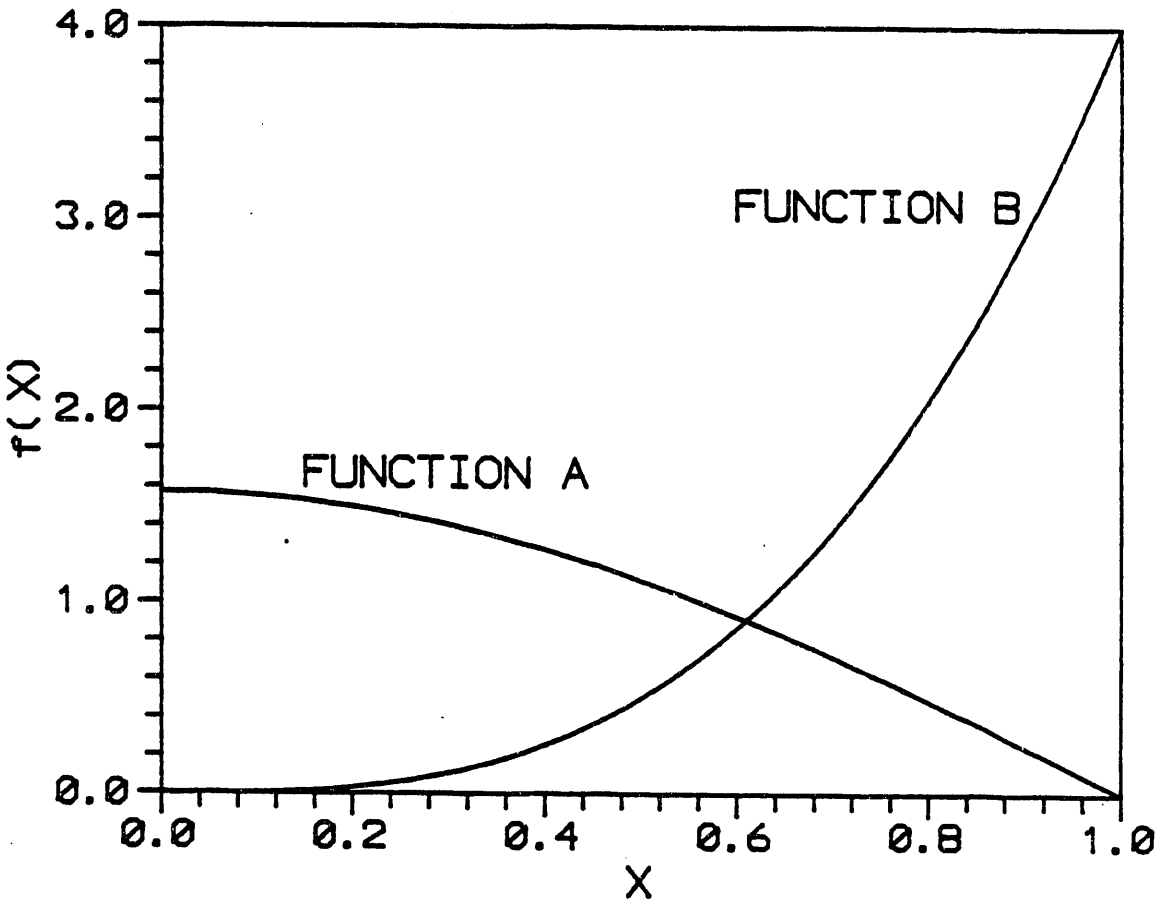

Figure 3: The two probability density functions used for the PDF separator problem.

Table 10: Comparative recall performance of neural networks derived by DNA and FNA schemes

Problem	Architecture	Scheme	Train. RMS error	Recall RMS error
Exclus-	2 x 2 x 1	DNA	9.9956E-03	0.00999
ive-nor	2 x 5 x 1	FNA	9.9278E-03	0.01831
	2 x 10 x 1	FNA	9.7582E-03	0.01983
8-to-1	3 x 4 x 8	DNA	9.0636E-03	0.00906
decoder	3 x 8 x 8	FNA	9.3911E-03	0.03642
	3 x 10 x 8	FNA	9.6355E-03	0.03849

the target upon further training, so a fourth node is added. Because this architecture does not learn the mapping fast enough a fifth node is added. When this 10 x 5 x 1 network is able to reach the target RMS error, one node is eliminated and the process continues. The scheme oscillates around the optimum architecture, which in this case is the 10 x 4 x 1 network.

Recall Performance

Further computer simulations were carried out for the exclusive-nor and 8-to-1 decoder problems to compare the effectiveness of the networks derived by the DNA scheme with those derived using a fixed node architecture (FNA) scheme.

The DNA scheme gave a 2 x 2 x 1 network as the optimum for the exclusive-nor problem. Two networks with 2 x 5 x 1 and 2 x 10 x 1 architectures were also trained on the exclusive-nor problem. These three networks were used to recall on data corrupted by noise. The noise added was uniform 1 All networks were trained to the same level of accuracy. The results prove the earlier assertions about the benefits of architecture optimization.

4.2.3 Power Plant Diagnostics

This work uses the DNA scheme in a backpropagation neural network to create an ANN based adviser that is able to recognize and classify seven different accidents and

normal operating conditions. The data for this work was simulated at the DAEC simulator. The seven accidents are

- recirculation loop 'A' rupture inside the primary containment (RR15)
- main feedwater line 'A' break inside primary containment (FW17)
- loss of extraction steam to feedwater heaters (MS14)
- High Pressure Coolant Injection (HPCI) steam supply line break in HPCI room (HP05)
- condensate pump 'A' trip (FW02)
- main steam line 'A' rupture inside primary containment (MS03) and
- main circulation water pump 'A' trip (MC01).

Descriptions of all these accidents can be found in the Appendix. These accidents were simulated for various intervals of time till the plant reached a more or less stable condition. In each case there are a few seconds of normal operating condition before the malfunction is inserted.

For these seven accidents, we had collected the values of eighty-one variables that were used to train the adviser. A listing of these variables can be found in Table 11. The data was normalized between 0.1 and 0.9. So a normalized value of 0.9 would correspond to a 100% meter reading in the plant for that variable. Also, for each accident and the normal condition we had a unique combination of three binaries. But consistent with the extremities of the normalized values, these were 0.1 and 0.9 rather than the usual 0 and 1. The seven accidents, their expected outputs and length of simulation can be found in Table 9.

Table 11: The 81 variables trended for the seven transients.

Number	Variable	Description
1	A041	Local power range monitor 6-25 flux level B
2	A091	Source range monitor channel B
3	B000	Average power range monitor A Flux level
4	B012	Reactor total core flow
5	B013	Reactor core pressure-differential
6	B014	Control rod drive system flow
7	B015	Reactor feedwater loop A flow
8	B016	Reactor feedwater loop B flow
9	B017	Cleanup system flow
10	B022	Total steam flow
11	B023	Cleanup system inlet temperature
12	B024	Cleanup system outlet temperature
13	B026	Recirculation loop A1 drive flow
14	B028	Recirculation loop B1 drive flow
15	B030	Reactor feedwater channel A1 temperature
16	B032	Reactor feedwater channel B1 temperature
17	B034	Recirculation loop A1 inlet temperature
18	B036	Recirculation loop B1 inlet temperature
19	B038	Recirculation A wide range temperature
20	B039	Recirculation B wide range temperature
21	B061	Reactor coolant total jet pumps 1-8 flow B
22	B062	Reactor coolant total jet pumps 9-16 flow A
23	B063	Reactor coolant total outlet steam flow A
24	B064	Reactor coolant total outlet steam flow B
25	B065	Reactor coolant total outlet steam flow C
26	B066	Reactor coolant total outlet steam flow D
27	B079	Reactor recirculation pump A motor vibration
28	B080	Reactor recirculation pump B motor vibration
29	B083	Control rod drive cooling-water differential pressure
30	B084	Control rod drive cooling-water differential pressure
31	B085	Torus air temperature #1
32	B086	Torus air temperature #2
33	B087	Torus air temperature #3
34	B088	Torus air temperature #4
35	B089	Drywell temperature azmuth 0 elevation 750
36	B090	Drywell temperature azmuth 245 elevation 750
37	B091	Drywell temperature azmuth 90 elevation 765
38	B092	Drywell temperature azmuth 270 elevation 765
39	B093	Drywell temperature azmuth 270 elevation 765
40	B094	Drywell temperature azmuth 180 elevation 780
41	B095	Drywell temperature azmuth 270 elevation 830

Table 11: . . . continued

42	B096	Drywell temperature center elevation 750
43	B098	Torus water temperature
44	B099	Torus water temperature
45	B103	Drywell pressure
46	B104	Torus pressure
47	B105	Torus water level
48	B120	Torus radiation monitor A
49	B121	Torus radiation monitor B
50	B122	Reactor water level
51	B125	Fuel zone level indication
51	B126	Reactor water level
53	B137	Torus water level
54	B138	Torus water level
55	B150	Core spray A flow
56	B151	Core spray B flow
57	B160	Reactor core isolation cooling
58	B161	High-pressure core injection
59	B162	Residual heat removal A flow
60	B163	Residual heat removal B flow
61	B164	Drywell radiation monitor A
62	B165	Drywell radiation monitor B
63	B166	Post-treat activity
64	B168	Pretreat activity
65	B171	Analyzer A O_2 concentration
66	B172	Analyzer A H_2 concentration
67	B173	Analyzer B O_2 concentration
78	B174	Analyzer B H_2 concentration
69	B180	Clean-up system flow
70	B196	Reactor water level-fuel zone A
71	B197	Reactor water level-fuel zone B
72	B247	Turbine steam bypass
73	B248	Turbine steam bypass
74	E000	4160 V Switch Gear bus 1A1 A-B
75	F004	Condensate pump A&B discharge pressure
76	F040	1P-1A reactor feed pump suction pressure
77	F041	1P-1B reactor feed pump suction pressure
78	F042	1P-1A reactor feed pump discharge pressure
79	F043	1P-1B reactor feed pump discharge
80	F094	Feedwater final pressure
81	G001	Generator gross watts

4.2.7 Training the Adviser

The adviser is trained to recognize the plant status by looking at the values of the plant data at any given moment of time. The value of the 81 variables at any time is thought to contain enough information as to make this possible (Bartlett & Uhrig 1992b). The training data is selected in an iterative manner (Bartlett 1990, Bartlett & Uhrig 1991b, Bartlett & Uhrig 1992b). In the first trial, the patterns at the beginning and the end of each of the seven simulations are used as the training set. The training was begun with 1 hidden node, so the starting architecture was 81 x 1 x 3. The first trial was with 14 training patterns. The trained network was then used to recall on the whole length of the simulations, and the RMS errors of the outputs for each of the patterns was plotted out. Obviously, the network does not do a very good job of classifying all the patterns. The patterns with the worst recall errors are added to the training data set, and the network from the previous trial is used to further train the network. This process is repeated till the network can successfully detect all the accidents within a reasonable amount of time. Care is taken so that patterns too close to the initiating event are not included in the training set. This is because these patterns exist at a time when the plant is highly unstable, and inclusion of the same might confuse the ANN. Information about the various trials is in Table 10.

As seen from Table 13, the final architecture for the adviser was 81 x 36 x 3. The output RMS error of this adviser for the seven simulations can be seen in Figs. 4 through 10. This work did not investigate scenarios where the data was corrupted by noise. As can be seen from the figures, most of the accidents are detected fairly quickly by the adviser. One notable exception is the trip of the main circulation water pump. This takes about 80 seconds to be detected. But the plant remains in an unstable state for a long time after the onset of the initiating condition in this case.

Table 12: Network output, simulation time and time to detect each scenario.

Transient scenario	Desired output node activation			Transient simulation time (in sec)	Scram time (in sec)	Transient diagnosis time (in sec)
	1	2	3			
Recirculation loop 'A' rupture inside primary containment (**RR15**)	0.1	0.1	0.9	304	1	17
Main feedwater line 'A' break inside of primary containment (**FW17**)	0.1	0.9	0.1	316	4	22
Loss of extraction steam to feedwater heater (**MS14**)	0.1	0.9	0.9	346	214	32
HPCI steam supply line break inside HPCI room (**HP05**)	0.9	0.1	0.1	78	–	15
Condensate pump 'A' trip (**FW02**)	0.9	0.1	0.9	764	–	38
Main steam line 'A' rupture inside primary containment (**MS03**)	0.9	0.9	0.1	165	8	14
Main circulation water pump 'A' trip (**MC01**)	0.9	0.9	0.9	586	–	75
Normal condition	0.1	0.1	0.1	–	–	–

Table 13: Training information of the adviser.

Trial	Num. of training patterns	Num. of hidden nodes Start	End
1.	14	1	7
2.	56	7	13
3.	90	13	17
4.	127	17	22
5.	169	22	24
6.	193	24	26
7.	230	26	30
8.	246	30	30
9.	262	30	35
10.	264	35	36

4.2.8 Conclusions

A systematic method to arrive at the appropriate network architecture for any given problem has been presented. This method is used to arrive at an ANN based adviser to detect and distinguish the seven accidents. The adviser is expected to be quite robust and quick to respond in an emergency. Further work might include the broadening of the scenarios investigated with a proportional increase in the number of plant variables trended. Also, the application of the importance function to the input nodes to determine the importance of the various inputs shows promise (Lanc 1991), further work needs to be done in that respect.

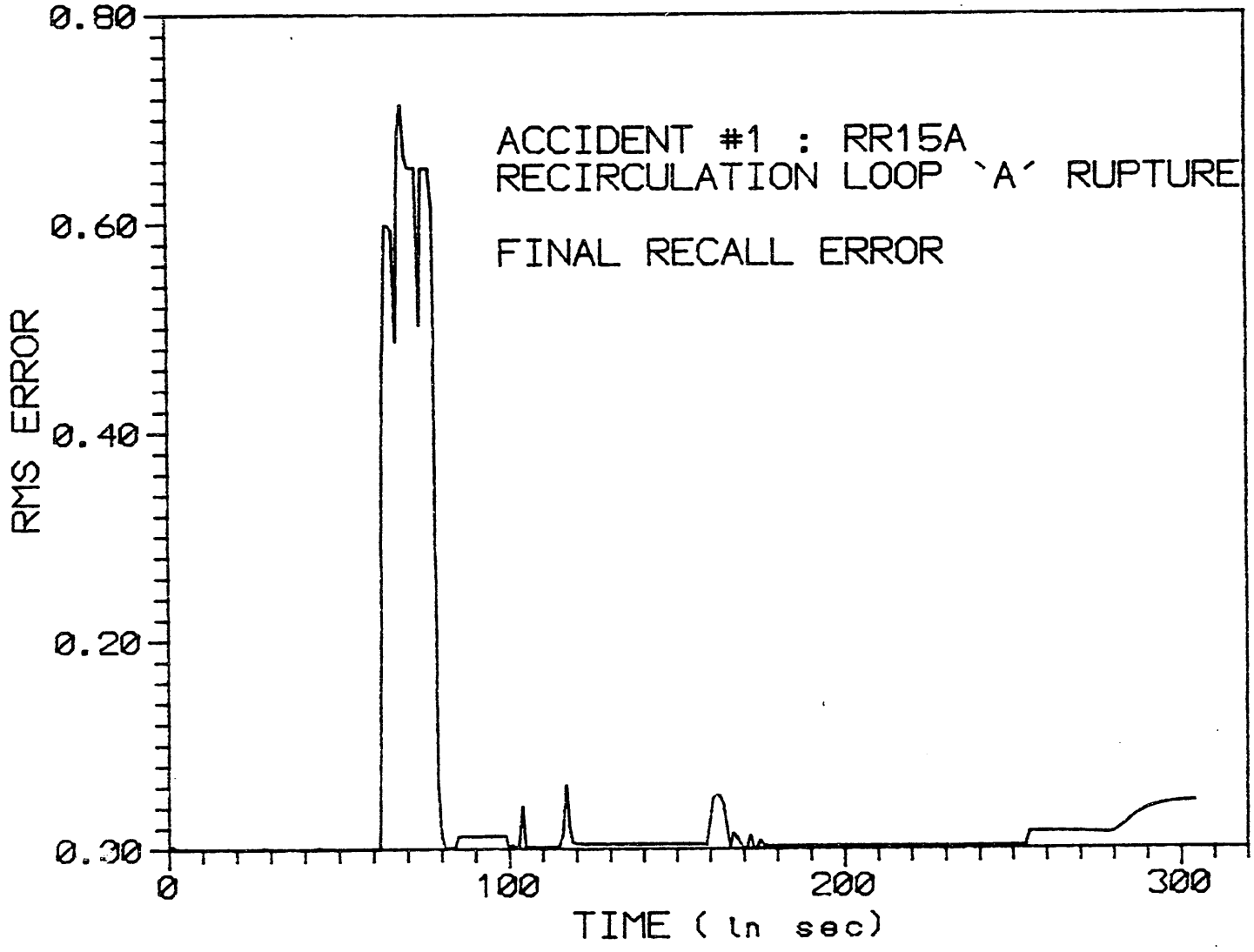

Figure 4: Final recall error for transient RR15A.

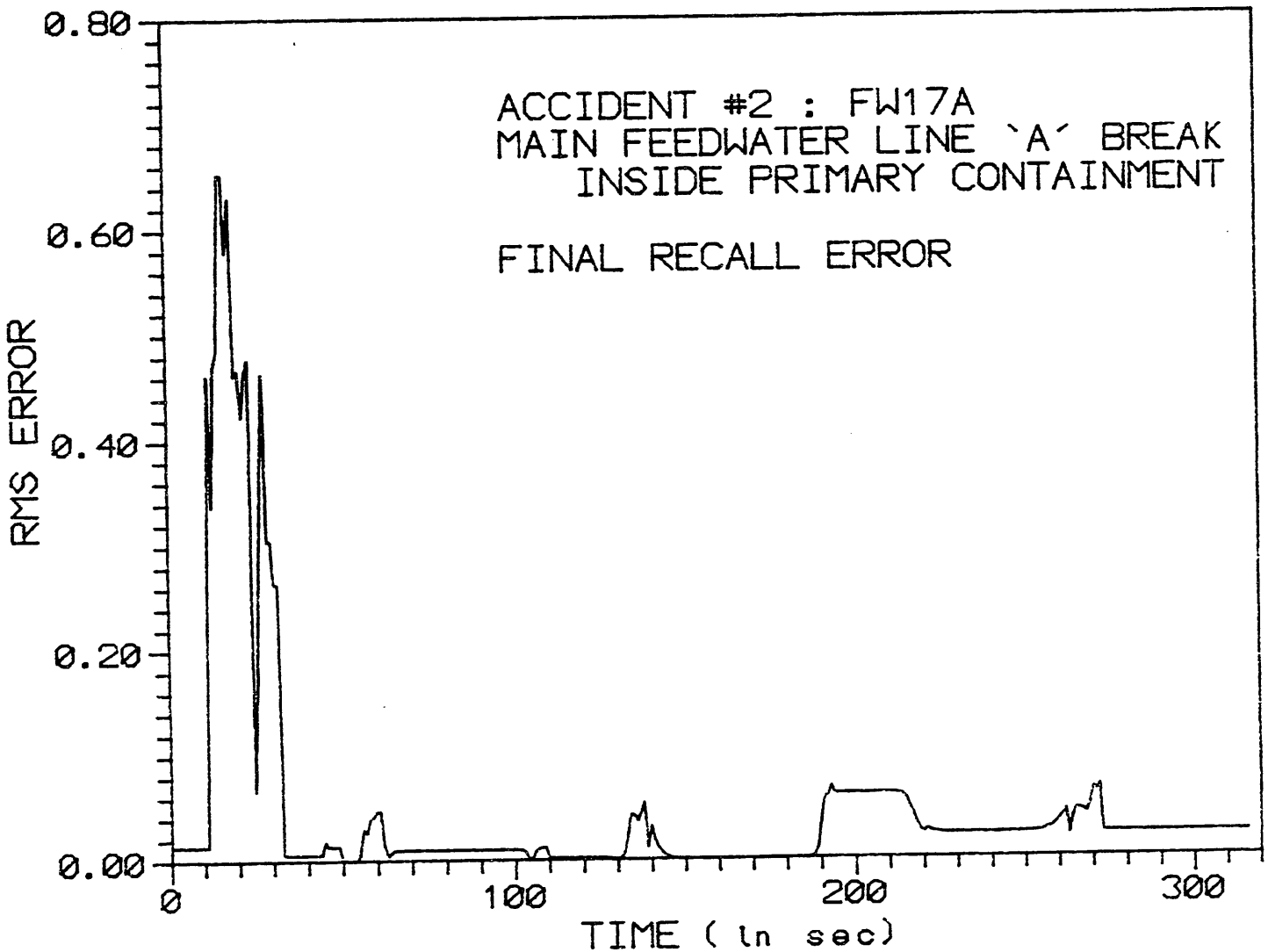

Figure 5: Final recall error for transient FW17A.

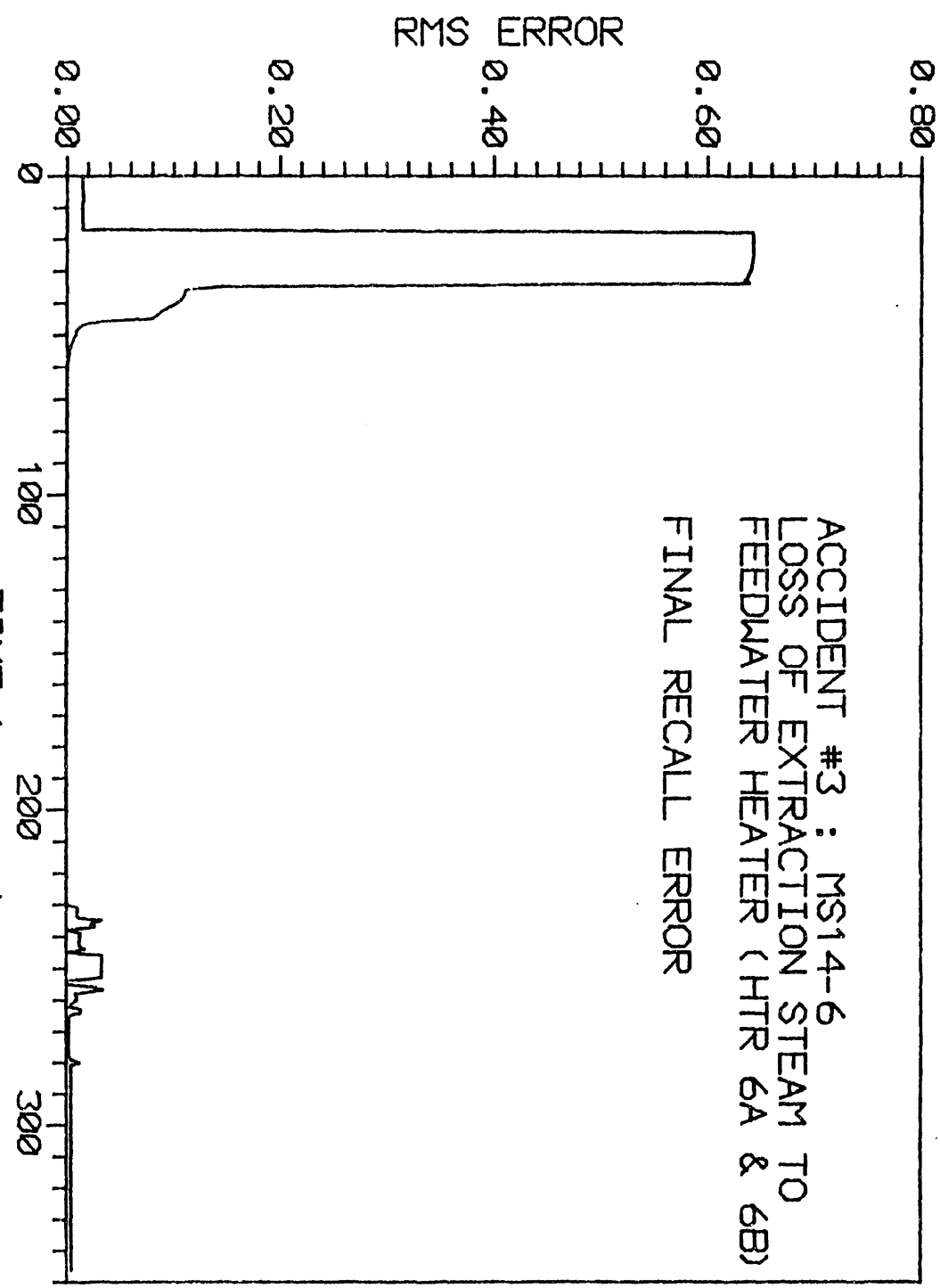

Figure 6: Final recall error for transient MS14-6.

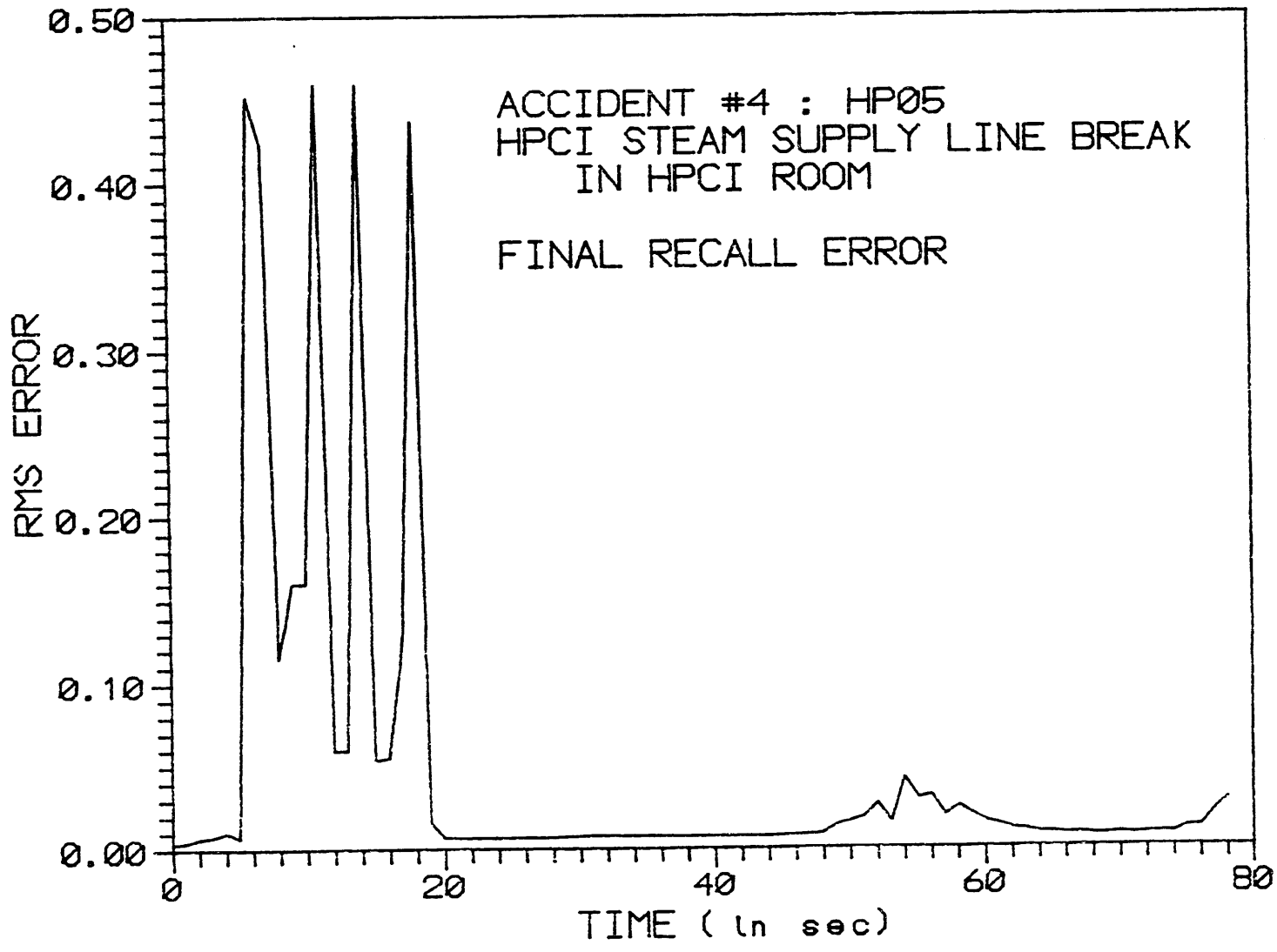

Figure 7: Final recall error for transient HP05.

Figure 8: Final recall error for transient FW02.

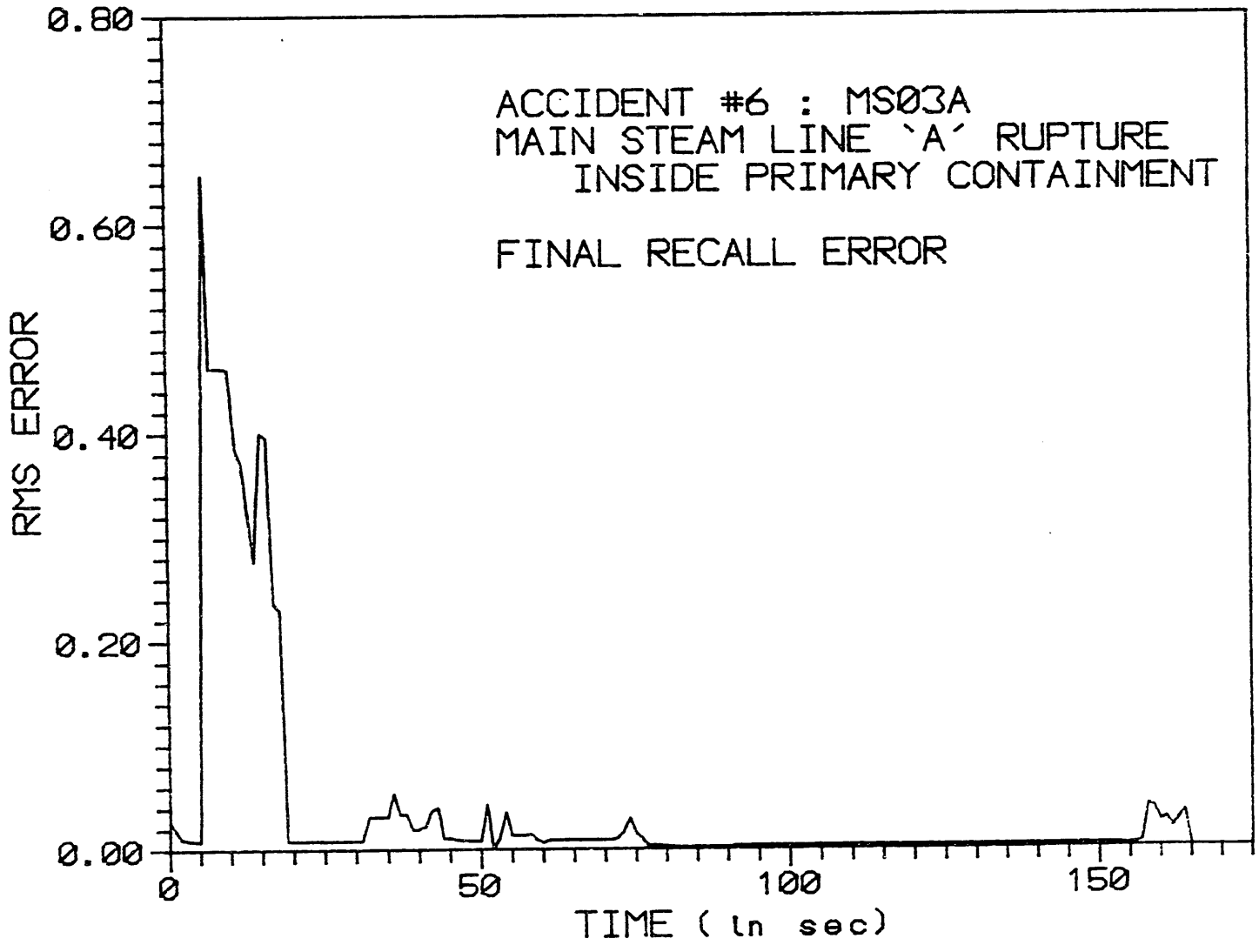

Figure 9: Final recall error for transient MS03A.

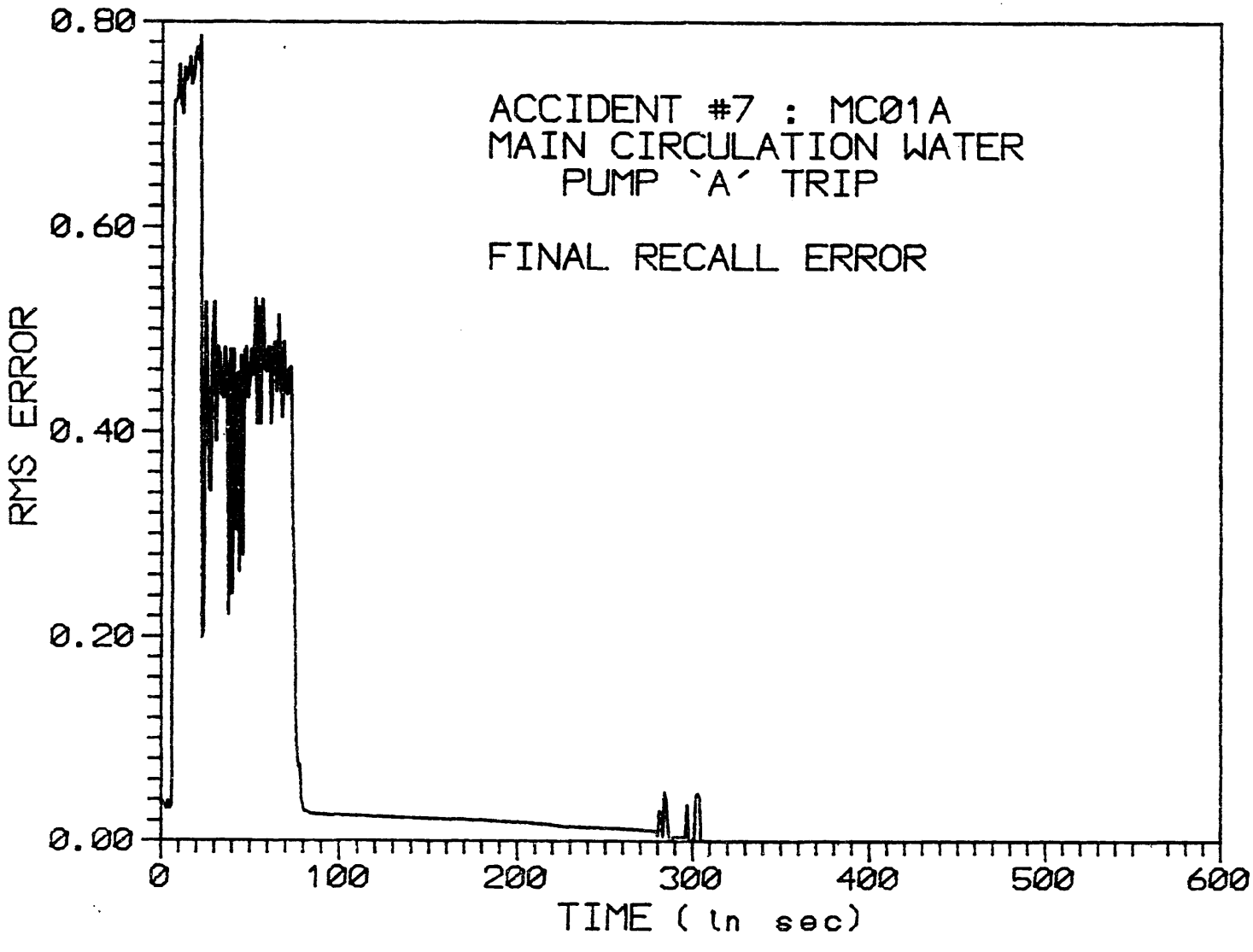

Figure 10: Final recall error for transient MC01A.

4.3 Application of Artificial Neural Networks for Nuclear Power Plant Diagnostics

by Chalapathy V. Dhanwada

The use of Artificial Neural Networks (ANNs) for nuclear power plant diagnostics is an innovative approach for improved plant safety. ANNs capable of a fast assessment of plant status were shown to be feasible in the past [(Bartlett and Uhrig, 1992; Ikonomopoulos et al. 1991). Currently, the use of ANNs in plant diagnostics is being investigated in the Nuclear Engineering Department at Iowa State University. The research work is focused on developing ANN paradigms capable of fast and accurate diagnosis of incipient abnormal conditions. An ANN based adviser suitable for use in plant control room is envisaged. Such an adviser can improve plant safety by directing the attention of the operators to abnormal conditions as soon as they occur. It also assists the operators in their decision process by furnishing its fast diagnosis of these conditions. The author's activities under the research project are summarized in this chapter.

4.3.1 Introduction

Abnormalities occurring in a nuclear power plant and their classification are briefly discussed in this section. An introduction to artificial neural networks and their application to plant diagnostics is also given.

4.3.2 Plant Abnormalities

Abnormalities occurring in a plant may be classified as incidents, transients, or accidents depending upon their severity. The Nuclear Regulatory Commission has divided the spectrum of possible abnormalities into nine classes, in increasing order of severity. A summary of this classification is shown in Table 14. An abnormality can be

triggered by a malfunction. As a result, a transient may occur and could develop into an accident depending on the type of malfunction. Usually a scram may occur, or a safety system may be actuated to stall or revert the progress of undesirable events. However, under the remote possibility that all safety systems fail, release of radioactivity into the environment may result. The purpose of using ANNs is to recognize the abnormality as soon as possible after its inception. Its diagnosis within the transient portion is desirable as that allows more time for the operators to take proper actions.

Several simulated abnormalities were considered by the author for recognition by ANNs. They represent a variety of malfunctions and a range of severity. The abnormalities are discussed in more detail in the next section. The ANNs were trained until the selected abnormalities could be classified within the transient portion. The abnormalities are referred to as "transients" in the following report.

4.3.3 Artificial Neural Networks

The structure of an artificial neural network is based on present understanding of biological neural systems. A network is composed of nodes and weights. The nodes are nonlinear computational elements and are linked to other nodes by weights. Thus, the output of a node is weighed and passed as input to other node(s). In its elementary form, a node sums all its inputs and passes the result through a nonlinear function. The weights of the network are variable. The set of rules which specify the initial weights and stipulate how they should be changed in order to improve the net's performance is called the training or learning algorithm. Several ANN training algorithms are available in the literature [Lippmann 1987, Widrow and Lehr 1990). In the present work the backpropagation (BP) technique (Hecht-Nielsen 1989) is used. The following discussion is restricted to ANN structure as applicable to multilayer networks trained using the BP technique.

Table 14: NRC Classification of plant abnormalities.

Class 1		Trivial incidents.
Class 2		Small releases outside containment.
Class 3		Radwaste system failures.
	3.1	Equipment leakage or malfunction.
	3.2	Release of waste gas storage tank contents.
	3.3	Release of liquid waste storage tank contents.
Class 4		Fission products to primary system (BWR).
	4.1	Fuel cladding defects.
	4.2	Off design transients that induce fuel failures above those expected.
Class 5		Fission products to primary and secondary systems (PWR).
	5.1	Fuel cladding defects and steam generator leaks.
	5.2	Off design transients that induce fuel failures above those expected and steam generator leak.
	5.3	Steam generator tube rupture.
Class 6		Refueling accidents.
	6.1	Fuel bundle drop.
	6.2	Heavy object drop onto fuel in core.
Class 7		Spent fuel handling accident.
	7.1	Fuel assembly drop in fuel storage pool.
	7.2	Heavy object drop onto fuel rack.
	7.3	Fuel cask drop.
Class 8		Accident initiation events considered in design basis evaluation in the safety analysis report.
	8.1	Loss-of-coolant accidents.
	8.1(a)	Break in instrument line from primary system that penetrates the containment.
	8.2(a)	Rod ejection accident (PWR).
	8.2(b)	Rod drop accident (BWR).
	8.3(a)	Steam line breaks (PWR's outside containment).
	8.3(b)	Steam line breaks (BWR).
Class 9		Events more serious than class 8 events. They are highly improbable. For example, complete loss of on-site and off-site power concurrent with LOCA, sudden rupture of the pressure vessel, or containment rupture due to an aircraft impact.

The schematic diagram of a three-layered network is shown in Fig. 11. A network layer is composed of one or more nodes. Each node of a layer is linked to each node of the next layer by a weight. Thus, the outputs from nodes of a layer are weighed and passed as inputs to the nodes of the next layer. In the form used here, a node sums its weighted inputs and passes the result through a sigmoid function. A layered ANN structure consists of a set of network inputs, one output layer, and one or more intermediate (or hidden) layers that separate the inputs from output layer. Each network input is linked to each node of the first hidden layer by a weight. Thus, the inputs to the network are not directly connected to any nodes. However, it is customary to refer to the set of inputs as the "input layer". This convention was found to be convenient and will be used in the following discussion. During the training procedure of the network, the weights are adjusted according to the training algorithm in order to improve its performance. The network is said to "learn" its training data during training. Training continues until desired accuracy of learning is attained.

In the current work, training is performed using the BP technique. The inputs to the network are propagated forward from one layer to the next in a sequential manner through the interconnecting weights until the output layer is reached. The actual outputs are compared with desired outputs. The error in the outputs is used as a measure of accuracy with which the network has learned its training data. The errors are propagated back into the hidden layers. The weights are adjusted by an amount proportional to the errors they cause. The feed-forward, backpropagation, and weight adjustments are repeated until the errors are below a specified level. The square root of the mean of squared errors in the output layer, referred to as the rms-error, is used as the figure of merit for the network's performance. Thus, the BP training technique is an iterative gradient algorithm designed to minimize the

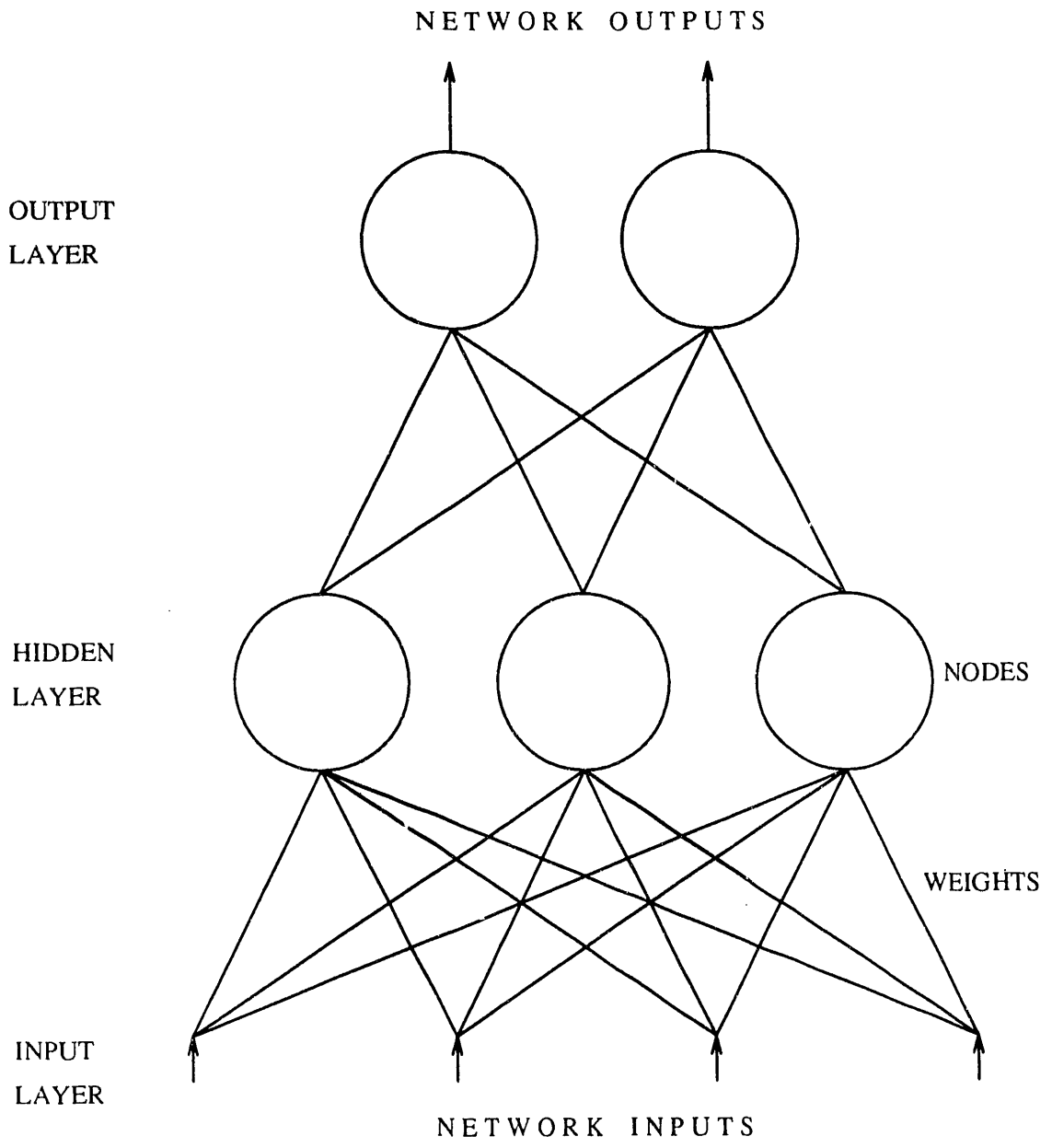

Figure 11: Schematic diagram of a three layer neural network. Nodes are represented by circles and weights by the interconnecting lines. This network has four inputs and two outputs represented by the arrows. It has a single hidden layer with three nodes in it.

rms-error.

In the preceding discussion, the network inputs and the desired outputs constitute a training pattern. Often several such patterns have to be used for training in order to obtain a network paradigm that performs the desired function. This set of training patterns is called the training set. A valid training set consists of one or more training patterns, each with the same number of inputs and same number of associated outputs. For a training set with more than one pattern, cumulative weight adjustments for the training set are determined and then added to the original weights during each iteration. This is known as batch training and is used in the following work.

A trained network will generate outputs with rms-error below the specified level when input patterns from the training set are forward propagated. This is implied by saying that the network has successfully "recalled" its training set. However, the real use of ANNs lies in its capability to successfully recall patterns that are close (but different from) the training set. This may be interpreted as a tolerance for noise in recalling data and is known as the generalizing capability of the network. The degree of generalization depends on the rms-error achieved during training. A high rms-error implies the network has not learned its training set properly. A very small rms-error means the network has "memorized" its training set and, therefore, will tolerate very little noise in its recalling data. In the following work, data normalized between 0 and 1 is used for training and recalling. Usually an rms-error between 0.01 and 0.1 for this data resulted in the desired degree of generalization.

4.3.4 Using ANNs in Plant Diagnostics

An artificial neural network uses examples of plant transients as its model. The network generalizes the characteristics of the examples during the training procedure. The examples are called transients known to the network. The network classifies a given situation based on its similarities with the examples. A particular transient is

identified if it is akin to one of the known transients. Any transient that is significantly different from all known transients cannot be classified successfully. Thus the domain of the network's "knowledge" is restricted only to its training examples. However, ANNs are capable of fast and accurate transient classification within their domain of knowledge. This will be exemplified by their performance discussed in later sections.

A transient example consists of some selected plant variables such as the reactor flux level, average coolant temperature, or others. The values of these variables are recorded at regular intervals, say, one second. Usually data is recorded to cover the entire transient and the safety actions or consequence following the transient. Raw data is not suitable for training purposes. The variables are normalized to take values between 0 and 1. The interrelationship between these variables gives an indication of the plant operating status during the transient. The ANNs are trained to generalize this relationship between the variables. The inputs of a training pattern consists of the values of the variables for a particular recording time. The outputs are chosen such that a unique ID may be assigned to each transient in the training examples. For example, if four transient examples are present in a training set, two outputs may be chosen with 00, 01, 10 and 11 as outputs for each example.

4.3.5 The ISU Accident Data Bank

An accident data bank for use by the ISU research team was set up at ISU. The data bank consists of examples of power plant accidents which can be used for training neural networks. Since this data forms an important component of the research work, it is briefly described in this section.

The accident data was obtained from the Nuclear Power Plant Simulator at the Duane Arnold Energy Center. Several examples of reactor abnormalities were chosen for simulation. These examples differ in type and severity. A summary of all the accidents simulated is included in Section 9. The ISU team held discussions with

the simulator personnel to decide upon what accidents to simulate and what plant variables to choose for inclusion in the data. The accidents chosen were generally ones with severe consequences since these are most important from safety point of view. The plant variables chosen are discussed in the next section.

4.3.6 Plant Variables

The term "plant variables" is used here to refer to all the variables the simulator keeps track of during simulation. The plant variables total several hundred. Of these, a subset (about three hundred) were chosen as possible inputs to the ANN based adviser. The variables of this subset , referred to as trend variables here, will be trended during accident simulation and written to a data file. The choice of these trend variables is based on their importance from safety point of view.

During each simulation a maximum of only 100 trend variables can be written to the data file. Since data for over three hundred variables is required, each simulation is slated for repetition with different trend variable set each time. Currently, data is being collected for the first set of trend variables. After data for the selected accidents is collected for this set, the simulations will be repeated for the other sets of trend variables.

Trend variables are of two types. The first type will be referred to as process variables. For the accidents being simulated now, these are ninety-seven in number. A list of all these variables is given in Table 2 (pages 9-12). The process variables are common to all simulations.

The second type of variables are commonly called the Boolean variables. Unlike process variables, the Boolean variables are not common to all accidents simulated. They represent the malfunction particular to the accident being simulated, and they take one of the two possible values of true and false. The malfunction takes effect when the Boolean is set to true. Because a maximum of 100 trend variables can be chosen,

there is room for three Boolean variables which is sufficient for the current work in progress. Usually, only one Boolean is required to insert the desired malfunction. But sometimes a safety feature can have an overriding effect on this inserted malfunction. Thus, space is provided for two more Booleans so that appropriate malfunction may be inserted for the overriding safety features.

4.3.7 Simulator Operation

A simulator that is "on" can operate in three basic states. It is in "RUN" if all the conditions existing are normal, i.e., the simulator is in steady state operating conditions. This state is assumed when the simulator is switched on, or when the reset command is given. A second possible state is freeze, abbreviated "FRZ". The simulator can be put on freeze at any desired point of time. All conditions existing at the time freeze is initiated are preserved until the simulator is reset. When abnormal conditions develop due to malfunctions inserted, a "reactor" scram can occur. This is the third possible state. The simulator continues to run in "SCRAM" state until reset. The freeze, reset, and run actions can be performed by either entering the command at the terminal or by pressing the appropriate buttons provided for the purpose. For the accidents simulated, the scram usually occurred as an automatic action.

Before the accident simulation starts, certain initial conditions are chosen. For example, conditions corresponding to the beginning, middle, or end of the fuel cycle can be selected. Since the ANN based adviser should be able to classify accidents independent of the initial conditions, data is being collected for each initial conditions. The next step is to load the trend variable list.

The simulation is performed with help of a batch command file. This batch file contains commands that puts the simulator on run and initiates a data collection procedure. A waiting time of about five seconds is included to obtain readings for

steady-state operation. Then the malfunctions as specified in the trend variable list are inserted. The simulation continues with these conditions until brought to freeze status through external command. This will mark the end of trend time. Typically, trend times of three to five minutes were used. This raw data consists of a brief description the accident followed by the values of the trend variables against time.

4.3.8 The Normalized Data

Computer programs were developed at ISU for reformatting and normalizing the raw data files as brought from the DAEC simulator. The minimum and maximum values each variable can take according to the instruments at the simulator were used to normalize the data. For normalized values between 0 and 1, the following formula was used:

$$Normalized\ \ value = \frac{Actual - Minimum}{Maximum - Minimum} \tag{10}$$

The variables in the simulator programs, termed "engineering variables", have no minimum or maximum limits imposed on them. It is possible for them to assume values beyond the range of the instruments. Hence, some normalized values can take negative values or values greater than one if the above equation is used. In such cases, the normalized values were hard limited to whichever limit was exceeded. Along with the normalized version of the accident data, information about each accident such as the number of Boolean variables, number of process variables, and the time at which malfunction is inserted is also generated.

4.3.9 Effect of Sensor Faults on Network Performance

A nuclear power plant has a complex arrangement of sensors which monitor various plant systems. When these sensors perform according their design specifications, the operating status of the plant may be assessed through their outputs. In the event of an abnormal condition at the plant, the sensor outputs will enable plant status to

be determined. Appropriate preventive measures may then be taken to counteract progress of undesirable events. In this light, a malfunctioning sensor is a potential safety concern because any deviation from normal in plant conditions may remain undetected. Providing sensor redundancy is, therefore, a standard industry practice. However, there is a small possibility that a total sensor failure will occur; and if it occurs, plant safety will be in jeopardy.

Conventional control systems monitor the sensor outputs and indicate the plant operating status through control room panel displays and CRT screens. Any abnormal conditions will be indicated to the operating personnel through annunciators, flags, or alarms. These control systems may also initiate automatic control actions if the type and/or severity of the abnormality warrant such an action. Generally, a degree of robustness is incorporated into these control systems so that a malfunctioning sensor will not adversely effect their performance. But the type and severity of the sensor malfunction can have a variable effect on their performance, such as misdiagnosing an event. The objective of the present work is to simulate a sensor fault and investigate its effect on the diagnosing ability of a neural network.

4.3.10 Training Data

The data used to perform the study was obtained from the training simulator at the San Onofre Nuclear Power Plant. The data consists of ten different simulated transient scenarios. For each of these transients, a total of thirty-three plant variables (common to all) were trended for about ten minutes. The variables are listed in Table 19 (page 81). From these ten data sets, two were chosen to study the sensor fault effects. These two data sets are:

1. Trip of all reactor coolant pumps

2. Trip of a single-reactor coolant pump

In each transient, of the thirty-three variables, five variables common to both sets remain constant throughout the trend time. They are:

1. Pressurizer relief steam flow
2. Pressurizer relief liquid flow
3. Source range counts (cps)
4. Reactor vessel head level
5. Reactor vessel plenum level

These variables do not contain information useful for classifying the two transients. Therefore, they have been eliminated from the data sets.

The data generated by the simulator cannot be used in its "raw" form for training the diagnostic network. Each variable was normalized with respect to the minimum and maximum values that variable takes in that data set, so that all values lie in the range 0 and 1. The normalized data retains the information content of the original data and is amenable for use in training the network. This is the data used in the training procedure of the network, as will be explained soon. In what follows, this data is referred to as "true data."

4.3.11 Network Architecture

The number of trend variables in the (true) data sets are twenty-eight. Accordingly twenty-eight input nodes to the network were chosen. The two data sets represent two distinct transients. Thus, to distinguish between the two, one node for output layer was chosen; this will enable a representation of the first transient with output equal to 0 and the second transient with output equal to 1. Other network architectures are possible. For example, two output nodes can be chosen with 01 and 10 as the

outputs representing the two transients. The input nodes can also be expanded with the additional inputs chosen as some primitives of other inputs.

The number of hidden layers and the number of nodes in each hidden layer can be varied from one to any higher number. After some initial trials, one hidden laver with twenty-five nodes in it was chosen. Thus the final network dimensions are 28 x 25 x 1. These dimensions remain fixed throughout our development here.

4.3.12 Training Procedure

The backpropagation algorithm was used for training the network on the true data. The RMS-error to be achieved was set equal to 0.1. This relatively high value was chosen with an intention of providing the trained network a degree of robustness against data contaminated due to sensor faults.

The training procedure consists of two distinct steps recursively executed until the network has achieved desired degree of generalization. The first step consists of training the network on training data set using backpropagation algorithm. The second step consists of using the network so obtained to "recall" or "feed-forward" the patterns in the true data set. If the desired cost function was not achieved for every pattern in the true data set, the training data set is modified in a manner explained below, and the above two steps are repeated. Once the desired cost function is achieved, the network parameters are saved and the training procedure ends.

Initially, the training data set is composed of four patterns. These four patterns are the first and the last pattern of each true data set. The first step is executed with this training set. The output generated by step two is analyzed, and some fairly uniformly-spaced peaks are chosen for inclusion in the training set. The objective is to train the network on those patterns which give high errors in the outputs. All peaks need not be included, since in practice it has been observed that one newly included peak tends to "pull down" its neighboring peaks after training. The expansion of the

training data set stops when all the patterns in the true data set are recalled with less than the specified RMS-error.

After the training procedure is complete, the network layout and weights are saved. This will be referred to as "trained network" in the following discussion and was used without any further modification to study the effects of simulated sensor faults.

4.3.13 Simulation of Sensor Faults

A sensor can malfunction in several ways. The sensor output in each case can be different depending on, for example, the severity or the type of malfunction. In this preliminary work, it was assumed that the sensor output assumes its minimum value when it is malfunctioning. Thus the fault data was generated from the true data by setting the value of a variable to zero throughout its trend time. It is possible that more than one sensor of the twenty-eight (corresponding to twenty-eight trend variables) can malfunction simultaneously. For example, two sensors can malfunction due to a common cause failure. However, for now, it is assumed that only one sensor can go bad at any time.

Two separate cases of sensor faults were considered. In the first case, the input 4 corresponding to the Cold Leg 1A Temperature (^0F) was set to zero. Two fault data sets for the two transients were generated from corresponding true data sets. In the second case, input 14, which corresponds to Steam Generator Feedwater Flow (gallons per min), was set to zero, and fault data sets were generated.

To study the effect of the simulated sensor faults, the fault data sets were propagated forward through the network trained on the true data sets. The objective is to examine the effect of the contamination in the data arising through simulated sensor fault on the performance of the network. The outputs generated for various data sets through feed-forward process by the trained network are now compared.

4.3.14 Results

Figure 12 shows the network performance for true data set of accident (1). The transient is successfully classified during the entire trend time. Figures 13 and 14 show the network performance with the simulated faults. Considerable deterioration is seen: the transient is classified correctly only after 100-150 sec. However for accident (2), no such deterioration occurs (Figs. 15, 16 and 17).

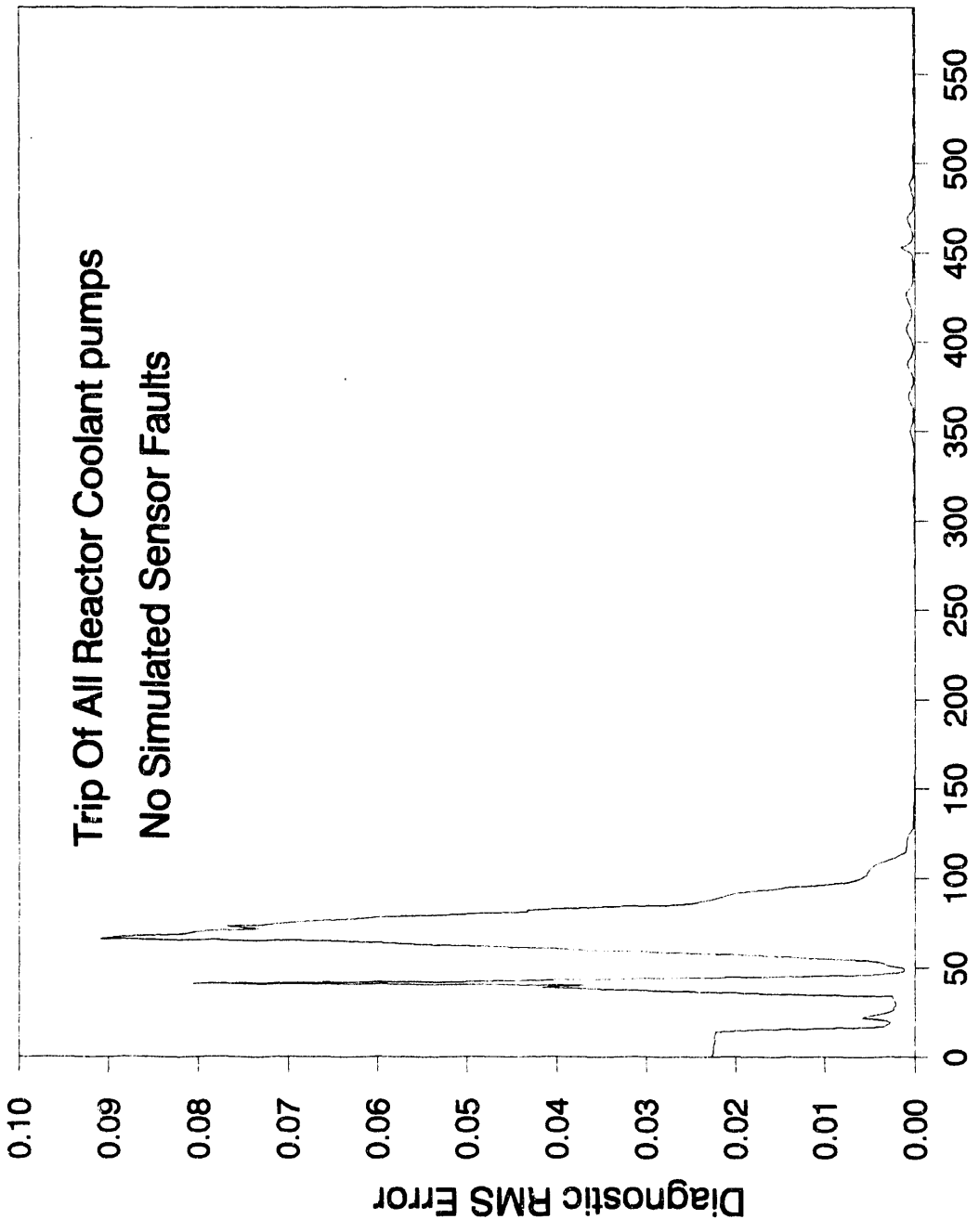

Figure 12: Network performance for accident (1) with true inputs

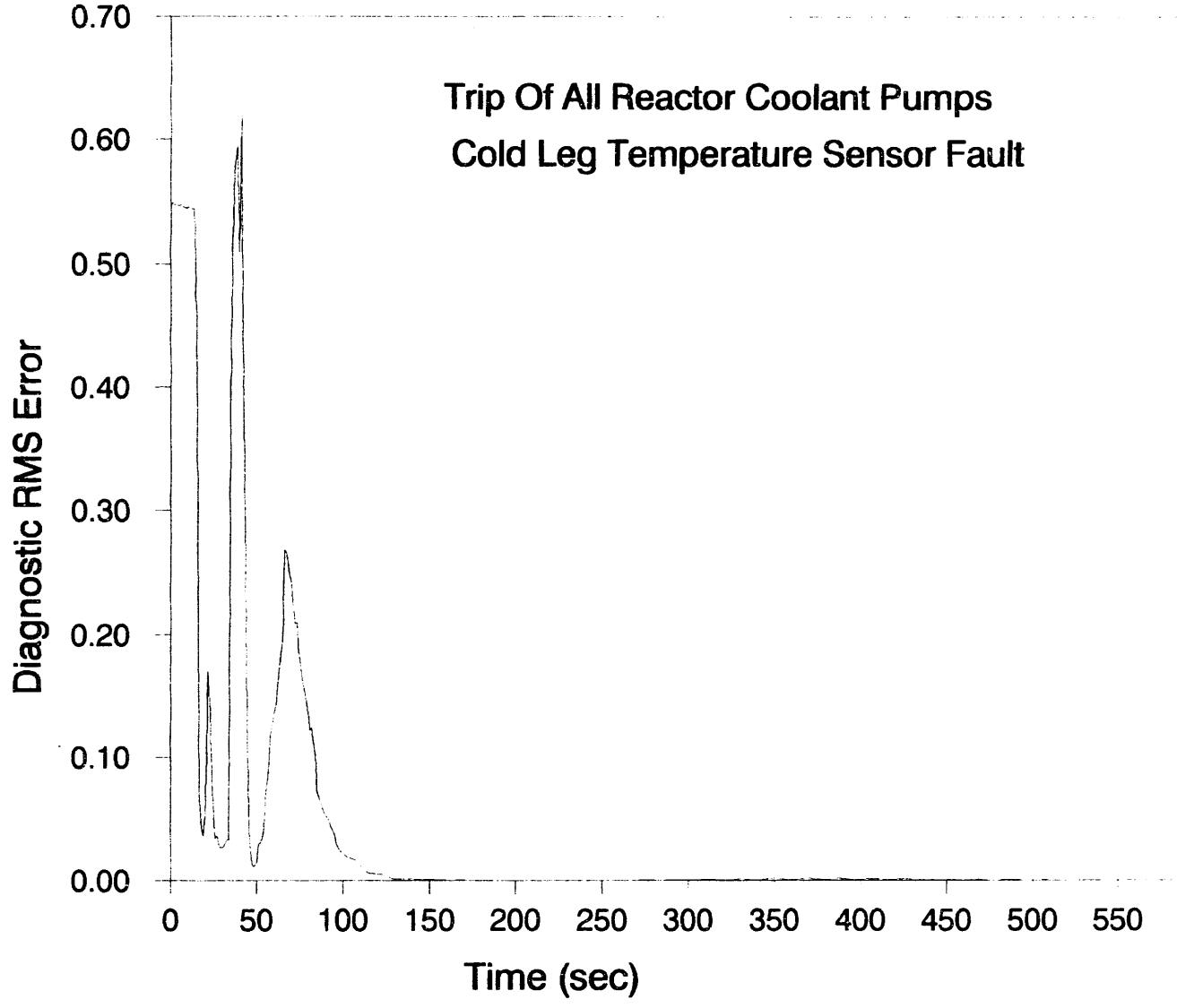

Figure 13: Network performance for accident (1) with fault in input 4

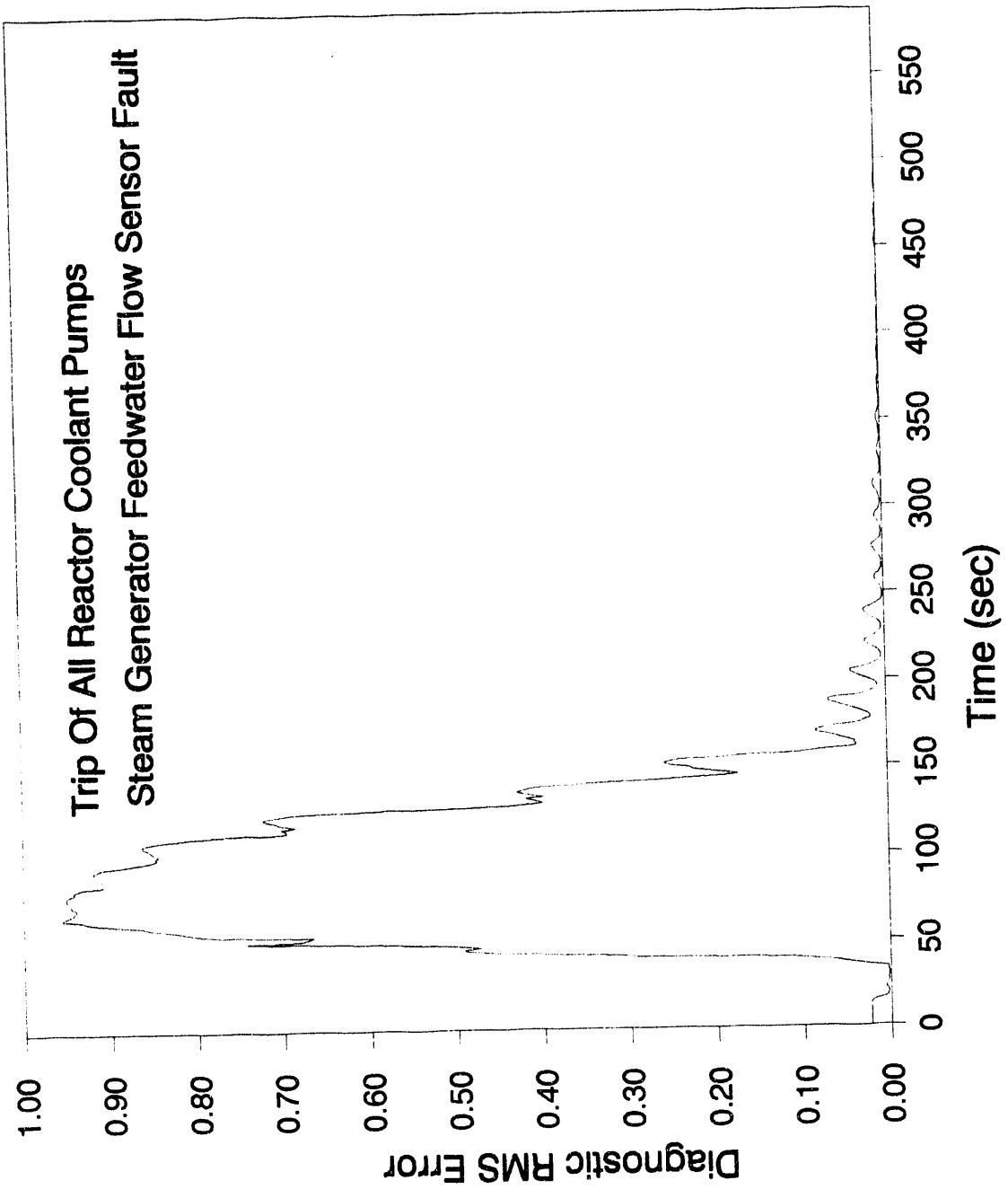

Figure 14: Network performance for accident (1) with fault in input 14

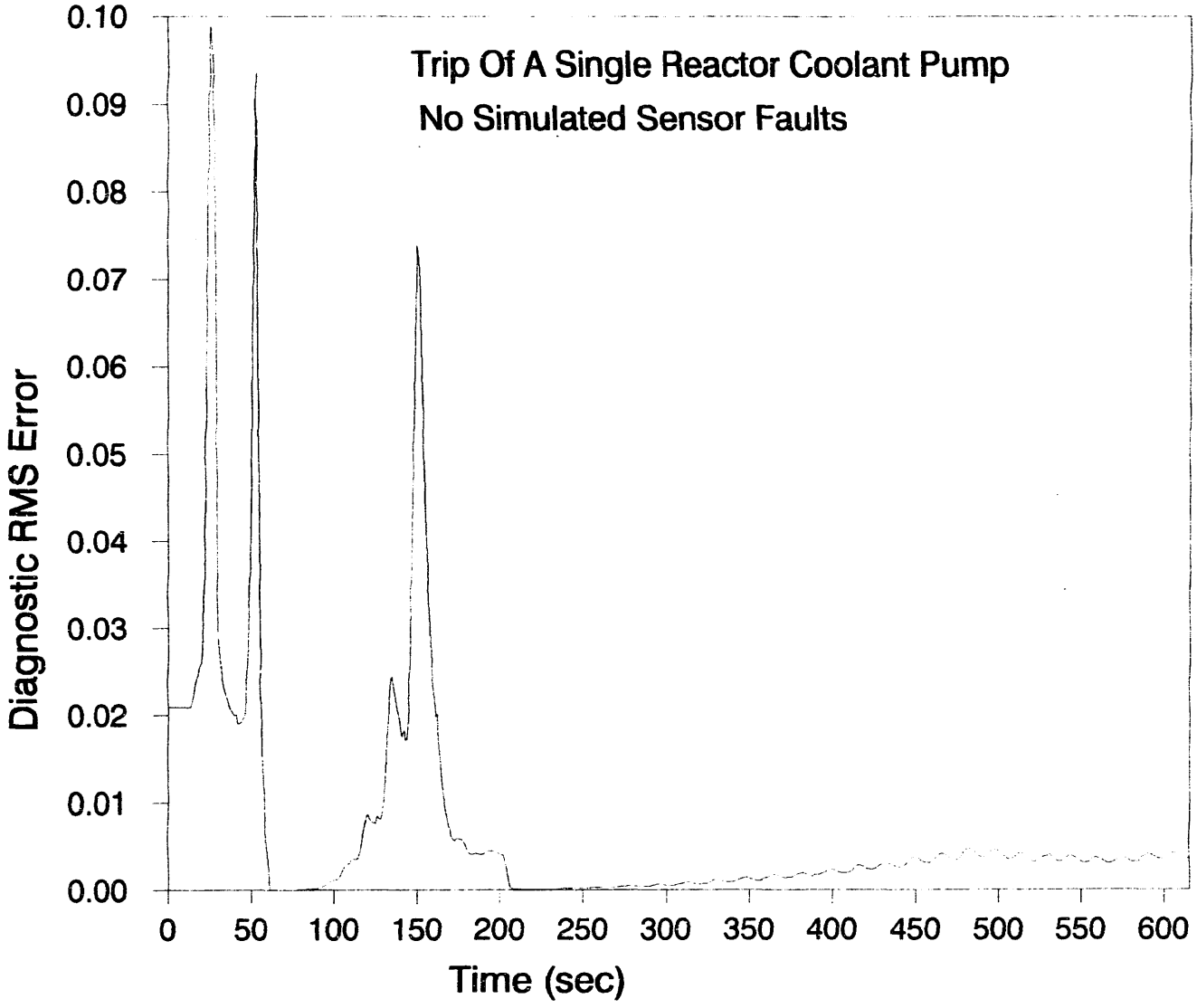

Figure 15: Network performance for accident (2) with true inputs

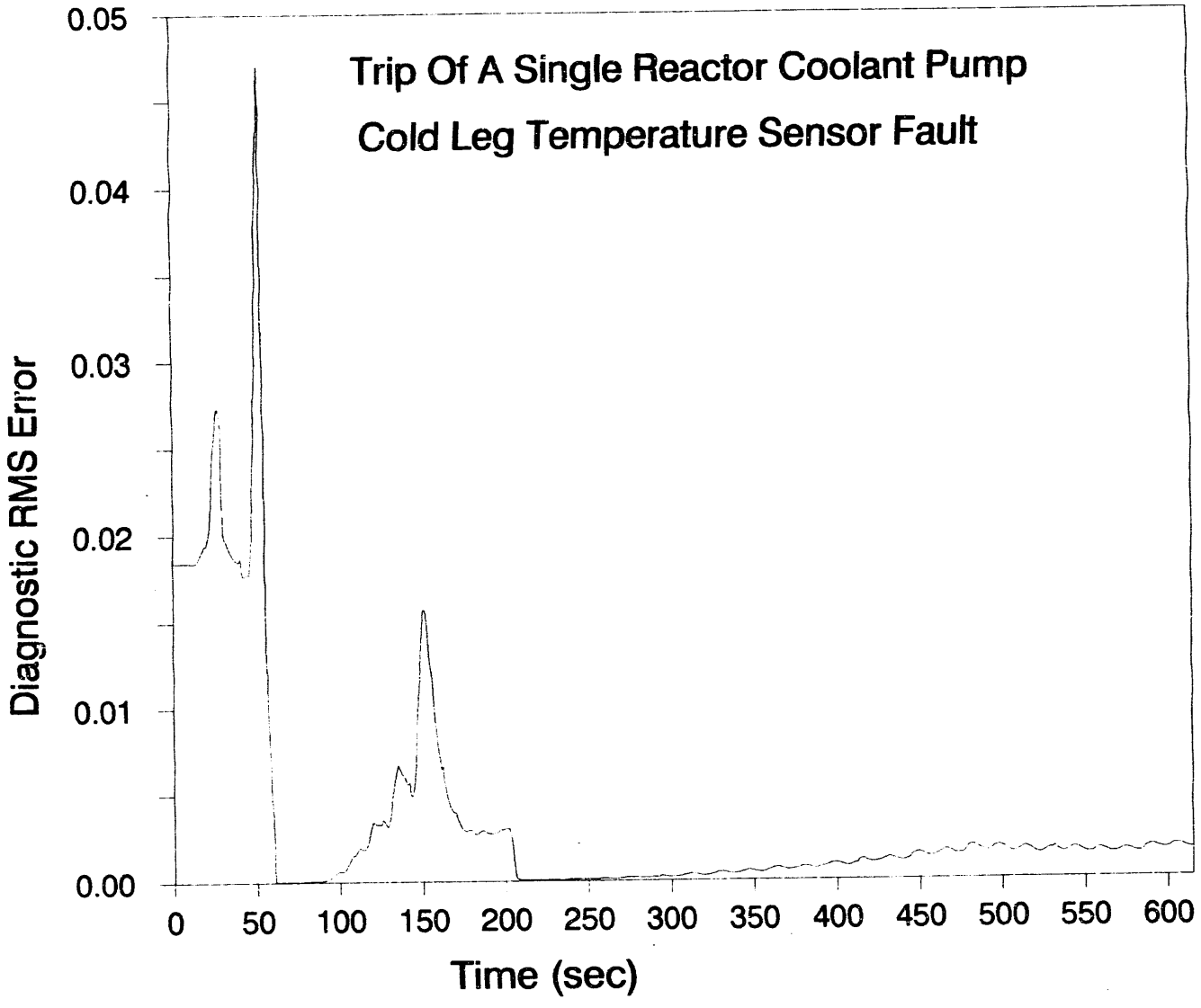

Figure 16: Network performance for accident (2) with fault in input 4

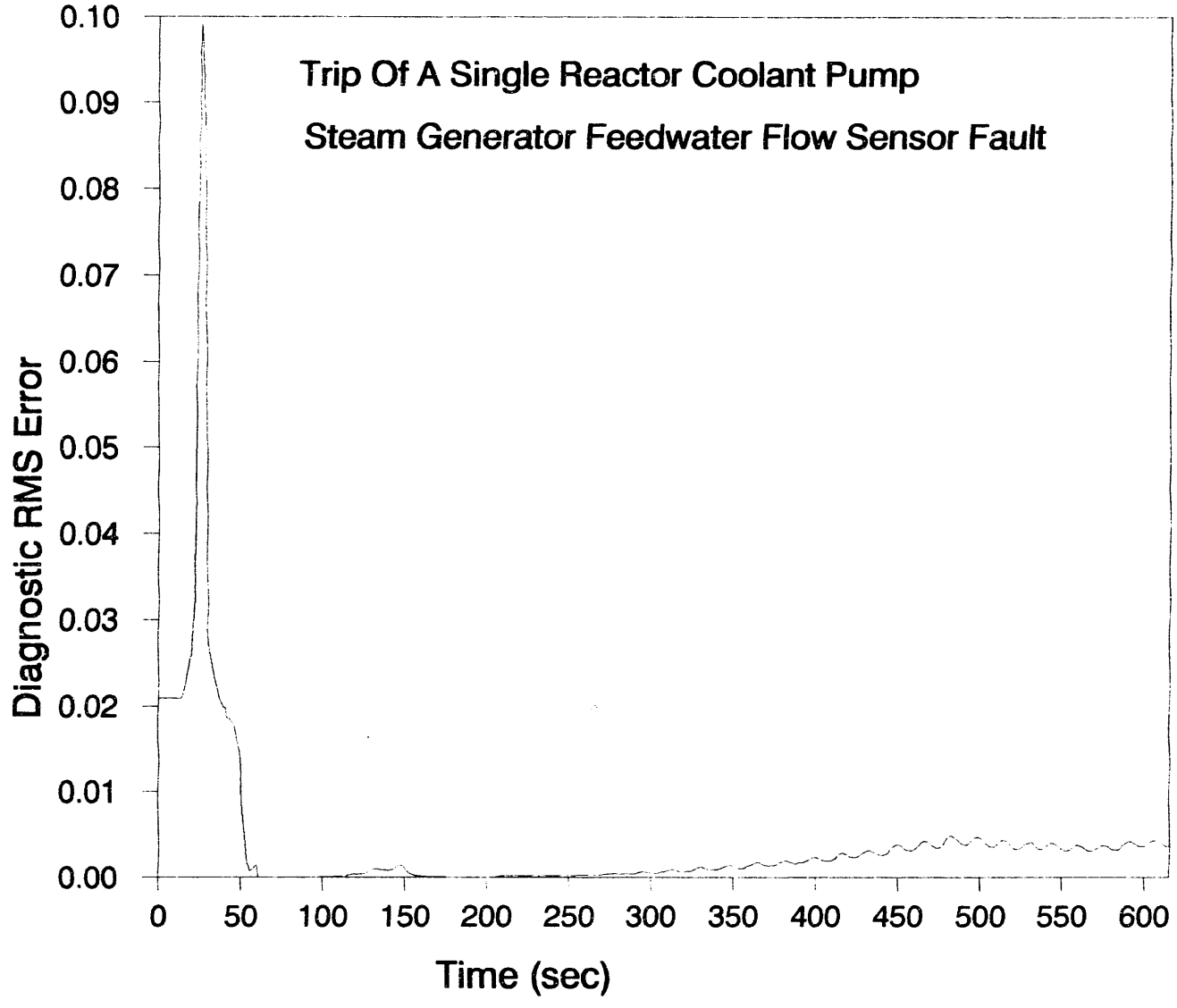

Figure 17: Network performance for accident (2) with fault in input 14

4.4 A Network Tree For Accident Classification

by Chalapathy V. Dhanwada

Neural network training paradigms suitable for use in a diagnostic adviser were developed earlier (see Sections 4.2, 4.6, 4.7.1, 10.5, 10.8 of this report; Bartlett 1990). As a demonstration, these methods were applied to a small set of accidents to test the ability of the networks to classify them in a timely manner and with as high an accuracy as possible. Typically, a satisfactory performance was reported. However, the total number of accidents in their training sets were usually ten or less. The ability of artificial neural networks to classify a larger set of accidents remained untested. Developing such a network will be a step towards building a diagnostic adviser capable of classifying a range of accident conditions. Accordingly, in this work an attempt was made to classify forty-one transient scenarios which included twenty-three distinct transient conditions. A significant departure from previous works was the use of network tree structure (net-tree) in place of a single diagnostic network. The results show that a successful classification of all the forty-one scenarios can be achieved in $3\frac{1}{2}$ minutes, or less.

4.4.1 Introduction

Earlier works cited above used a single network to achieve the classifications. This was usually found to be sufficient because of the small number of the accidents that were used for classification. However, initial trials to classify forty-one scenarios using a single network revealed several inadequacies of such an approach. The network size tended to become large as the training proceeded towards achieving better diagnosis time. This is to be expected because an expanded set of accidents would require a larger network for their successful classification. Because of the range of accident conditions, all ninety-seven available plant variables in the data sets were used as

inputs to the network. The increased number of accidents required more number of training patterns to be included in the training set. This resulted in long training times. As a remedy, a network tree structure was used in the current work. This allowed the problem to be divided into smaller independent parts. Despite an increase in the complexity of training procedure, this approach provided some advantages such as independent training of smaller networks for a relatively shorter amount of time. The following sections describe this approach.

4.4.2 Network Tree

In an attempt to alleviate the network size and training time requirements, a simple network tree structure was used to replace the single network. A schematic diagram of the net-tree used here is shown in Fig. 18. The net-tree is composed of several small networks similar to the ones used in previous works. These individual networks are independent of each other and each of them is trained to perform a distinct desired function. It is intended that, once these smaller networks are trained and used in the net-tree, a diagnostic adviser capable of classifying the accidents will be formed. Since the scope of function of the smaller networks is limited, a small size is expected to be adequate. Also, since they are independent of each other, all the networks can be trained simultaneously[1]. This is expected to take significantly lower training time than would a single network that carries out the entire task. In Fig. 18, these

[1] During the course of this work, simultaneous access to five DECstation 3000/5000 machines was available. The net-tree has a total of six independent networks which shared these five machines during their training stage. Overall, this resulted in the faster development of the net-tree.

networks are represented as rectangular boxes. Each of them is labeled according the function performed. Further description follows in the sections below.

4.4.3 The Accident Set

The set of forty-one accident scenarios considered for classification are listed in Table 15. A brief description of the nature of each accident is also given. Further description can be found in Section 9. The data was obtained from the training simulator at Duane Arnold Energy Center (Vest & Berchenbriter 1991). During each simulation, data for ninety-seven plant variables was collected. These variables are described in Table 2. In this work, all these ninety-seven variables are used as inputs to the networks in the net-tree. Distinct binary coded outputs were assigned to each accident. The twenty-three distinct accident conditions required five outputs to be used for each accident. Some of these accidents were simulated at various severities. For example, fw18a, fw18a_2, fw18a_3 are were simulated at 100%, 60%, and 30% severity of the same malfunction. These different severity simulations of the same accident condition were assigned the same output (See column Binary ID, Table 15). It is expected that, if the net-tree establishes the nature of the accident condition, another network to determine its severity (100%, 60%, or 30%) can be developed. This later task is relatively easier due to the small set to be classified.

4.4.4 Root Network

The individual components of the net-tree are described in this and the following sections. The function of the root network is to distinguish between normal (no-accident) condition and an accident condition. The outputs of the node 1-5 networks collectively establish the nature of the accident condition.

The root network has a single output. The output for a normal condition is 0, and for an accident condition output is 1. Thus, for all the forty-one accident scenarios,

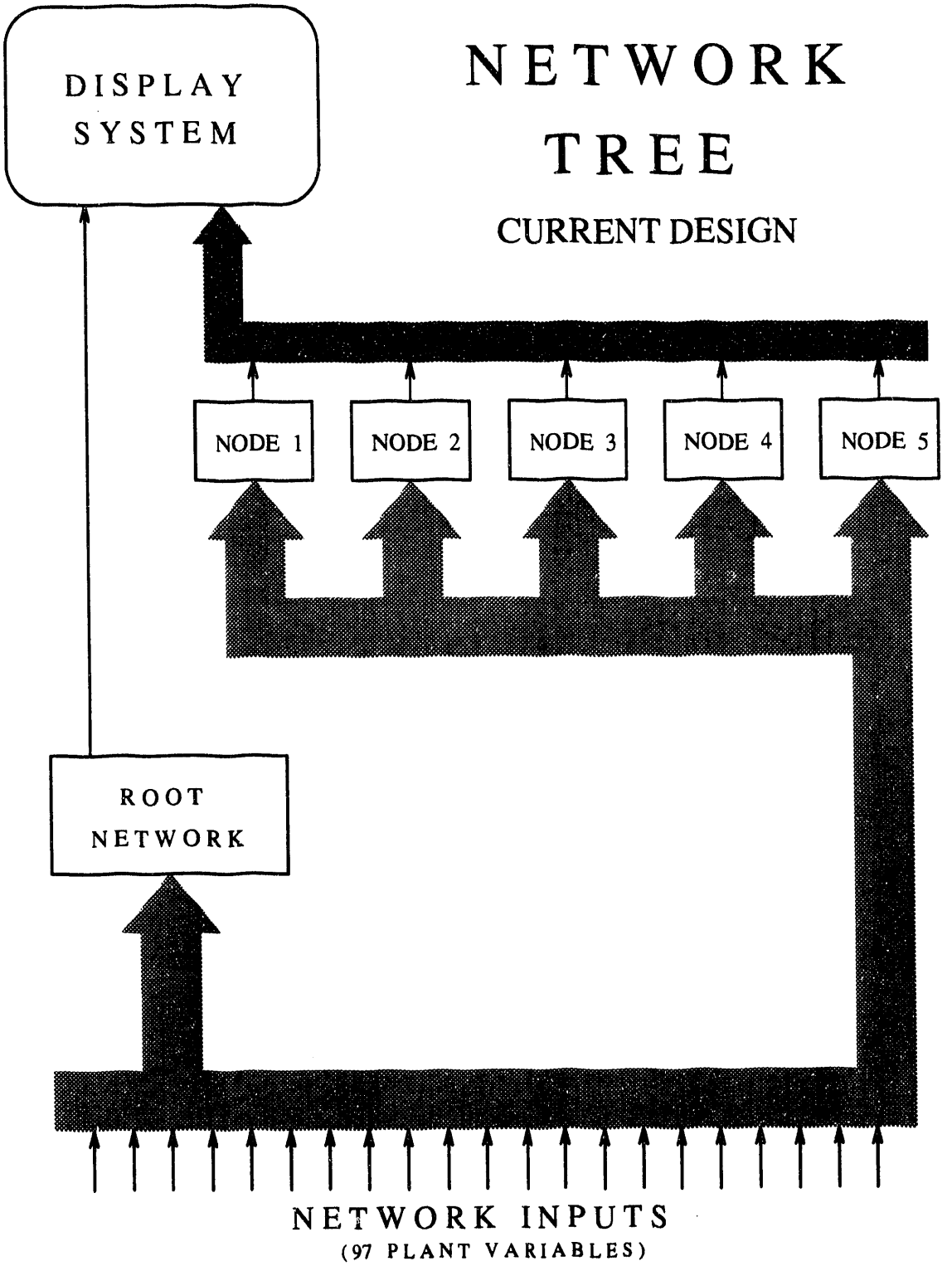

Figure 18: Network-tree design used for the current adviser

Table 15: List of transient scenarios considered for classification by the net-tree and their corresponding binary IDs.

S. #	Scenario	Binary ID	Description
1	cu10	00001	Coolant leakage outside primary containment 100% severity.
2	cu10gp5	00010	Malfunction cu10 with automatic group 5 isolation overridden
3	fw04a	00100	Condensate filter demineralization resin injection, 100% severity: 5% resin release.
4	fw09a	00101	Reactor feedwater pump trip
5	fw17a_2	00110	Main feedwater line break inside primary containment, 60% severity
6	fw17a_3	00110	Main feedwater line break inside primary containment, 30% severity
7	fw18a	00111	Main feedwater line break outside primary containment, 100% severity
8	fw18a_2	00111	Main feedwater line break outside primary containment, 60% severity
9	fw18a_3	00111	Main feedwater line break outside primary containment, 30% severity
10	hp05_2	01000	High Pressure Coolant Injection (HPCI) steam supply line break in HPCI room, 60% severity.
11	hp05_3	01000	HPCI steam supply line break in HPCI room, 30% severity
12	hp08_2	01001	HPCI steam supply line break in torus room, 60% severity
13	hp08_3	01001	HPCI steam supply line break in torus room, 30% severity
14	ic14scra	01010	spurious scram with initial condition IC14: 100% power, Beginning of Cycle (BOC)
15	ic20hp05	01011	HPCI steam supply line break in HPCI room, supply line break in HPCI room, 100% severity.
16	ic20hp08	01100	HPCI steam supply line break in torus room, 100% severity.
17	ic20scr1	01101	spurious scram with initial condition IC20: 100% power, End of Cycle (EOC). Inadequate operator action, scram occurs.
18	ic20scr2	01101	spurious scram with IC20, 100% power, EOC. Adequate operator action, scram prevented.
19	ic20scrm	01101	spurious scram with IC20, 100% power, EOC. No operator action.

Table 15: . . . Continued

S. #	Scenario	Binary ID	Description
20	ic22scrm	01110	spurious scram with initial condition IC22, 25% power, beginning of cycle (BOC).
21	ic23scrm	01111	spurious scram with initial condition IC23, 75% power, BOC.
22	ic24scrm	10000	spurious scram with initial condition IC24, 100% power, Middle of Cycle (MOC).
23	ms02_2	10010	RCIC line break inside primary containment. 60% severity.
24	ms02_3	10010	RCIC line break inside primary containment. 30% severity.
25	ms03a	10011	Main Steam Line (MSL) rupture inside primary containment, 100% severity.
26	ms03a_2	10011	MSL rupture inside primary containment, 60% severity
27	ms03a_3	10011	MSL rupture inside primary containment, 30% severity
28	ms04a	10100	MSL rupture outside primary containment, 100% severity.
29	ms04a_2	10100	MSL rupture outside primary containment, 60% severity.
30	ms04a_3	10100	MSL rupture outside primary containment, 30% severity.
31	ms19ab	10101	Spurious group I isolation. Feedwater relays A & B logic failure.
32	rp05tc01	10110	Trip of Main Turbine & failure to scram
33	rp5actc1	10111	Trip of Main Turbine, failure to scram & Alternate Rod Insertion (ARI) failure
34	rr10	11000	Recirculation pump speed feedback signal failure
35	rr15a_2	11001	Recirculation loop rupture, 60% double-ended shear
36	rr15a_3	11001	Recirculation loop rupture, 30% double-ended shear
37	rr30	11010	Coolant leakage inside primary containment, 100% severity.
38	rr30_2	11010	Coolant leakage inside primary containment, 100% severity. 60% severity.
39	rr30_3	11010	Coolant leakage inside primary containment, 100% severity. 30% severity.
40	rx01	11011	Fuel cladding failure, 100% severity.
41	tc02	11100	Electro Hydraulic Control (EHC) pump trip

patterns corresponding to time prior to malfunction insertion are assigned an output 0, and the rest of the patterns are assigned an output 1.

The training of the root network was carried out in conventional manner. Two patterns from each scenario, one picked from the start of the simulation (which corresponds to no-accident condition) and other from the end of simulation form the initial training set. After the desired training accuracy (0.1 RMS) was achieved, a recall was performed for each of the accident scenarios. Patterns resulting in high recall error were included progressively in the training set, starting with the ones near the end of simulation, and the training procedure was repeated. The aim of the training was to achieve output 0 for normal conditions and output 1 for any accident condition as soon as possible after the malfunction insertion. The results of this training are shown in Table 16. It is seen that the desired output is obtained in less than a minute after malfunction insertion for all the accident scenarios. Thirty-four of the forty-one scenarios are diagnosed in less than thirty seconds. Thus, independent of the next level networks, the presence of accident conditions is established by the root network in less than one minute following the initiating event.

As will be seen later, the purpose of using a root network is to reduce the size of training sets for the group of networks which perform the accident classification. The function of these "node networks" is described next.

4.4.5 The Classification Networks

Each accident condition has ninety-seven inputs and a distinct five digit binary code assigned to it as the output. If a single network is used to train the accidents, a $97 \times X \times 5$ dimension network would be required, where X represents the number of hidden layers chosen and the nodes within each hidden layer. However, such an approach would result in large network size and long training time requirements. Hence, in the net-tree structure, five independent $97 \times X \times 1$ networks replace the single

Table 16: Diagnosis times for the forty-one accident scenarios for each individual network in the net-tree. The last column lists overall net-tree diagnosis time for each accident

Scenario	Root	Node 1	Node 2	Node 3	Node 4	Node 5	Net-tree
cu10	31	14	1	84	19	1	84
cu10gp5	13	193	78	202	1	106	202
fw04a	52	211	144	181	146	1	211
fw09a	7	23	13	1	4	1	23
fw17a_2	1	154	17	116	3	128	154
fw17a_3	2	158	138	25	45	1	158
fw18a	8	1	1	1	1	1	8
fw18a_2	17	1	39	34	10	65	65
fw18a_3	14	31	4	1	52	1	52
hp05_2	23	36	49	87	1	1	87
hp05_3	32	43	51	85	1	1	85
hp08_2	37	94	1	1	1	1	94
hp08_3	4.	134	1	1	1	1	134
ic14scra	2	78	135	107	78	1	135
ic20hp05	14	72	72	73	1	1	73
ic20hp08	15	72	74	74	1	1	74
ic20scr1	1	3	88	55	84	1	88
ic20scr2	1	3	1	124	42	1	124
ic20scrm	1	23	85	55	79	1	85
ic22scra	2	78	1	117	1	2	117
ic23scrm	1	116	53	79	78	80	116
ic24scrm	2	78	94	99	104	97	104
ms02_2	4	70	41	74	32	74	74
ms02_3	5	72	44	81	35	93	93
ms03a	8	100	136	79	3	13	136
ms03a_2	1	164	7	65	4	54	164
ms03a_3	1	147	14	71	29	69	147
ms04a	4	145	1	34	4	58	145
ms04a_2	4	137	1	35	4	58	137
ms04a_3	3	128	1	34	4	58	128
ms19ab	18	95	1	50	20	74	95
rp05tc01	22	102	106	19	49	16	106
rp5actc1	23	169	125	18	22	140	169
rr10	41	140	1	54	1	146	146
rr15a_2	2	81	157	179	171	15	179
rr15a_3	2	2	86	89	80	140	140
rr30	12	83	57	103	84	98	103
rr30_2	9	103	78	110	104	119	119
rr30_3	5	133	122	140	135	146	146
rx01	2	16	47	2	75	78	78
tc02	27	45	1	49	29	23	49

network. While all these smaller networks have common inputs, the single outputs are the bits of the five-digit binary output assigned to the accidents. These networks are labeled node 1, node 2, node 3, node 4 and node 5 in Fig. 18. Node 1 is trained on the least significant bit of the five-digit binary output; node 5 network is trained on the most significant bit. Node 2-Node 4 networks are trained on bit 2-bit 4 respectively. The net-tree is said to give correct diagnosis of an accident condition if the output of the five networks together forms the desired binary ID for that accident. Because of the reduced number of outputs, these smaller networks individually take lesser training time than a single larger five-output network.

Since the root network establishes the presence of an accident, the node networks are relieved of performing this task. Thus, only the patterns corresponding to presence of a malfunction were used to train these networks. The initial training set for a node network consists of one pattern for each accident at the end of its simulation. The output assigned to these patterns depends on the accident and the particular node network being trained. Thus, for node 1 network the least significant bit of the accident binary IDs were assigned as outputs to their corresponding input patterns. Similarly, training sets for the other node networks were formed. These networks were then trained independently to achieve an RMS error of 0.1. A recall was performed for all the accidents on patterns occurring after malfunction insertion. Those patterns with high RMS error and nearer to the end of simulation were included in the training set and the next phase of training was carried out. In this manner, a progressively refined diagnosis time for each of the node networks was obtained.

4.4.6 Results

The self-optimizing stochastic learning algorithm was used to train the networks (Bartlett & Uhrig, 1990a,b). Since this a dynamic node architecture scheme, an optimum network size is obtained. In this work, a single hidden layer is used for all

Table 17: The dimensions of the root network and the node networks for the results given in to Table cdtab2. The training set sizes are also shown. The number of patterns used during recall, all accidents inclusive, are 9363. Of these, 269 correspond to normal operating conditions.

Network	Inputs	Outputs	Network Dimensions	Training set size
Root	97	1	97×19 × 1	113
Node 1	97	1	97× 4 × 1	69
Node 2	97	1	97× 7 × 1	63
Node 3	97	1	97×11 × 1	57
Node 4	97	1	97× 2 × 1	58
Node 5	97	1	97× 1 × 1	60

the networks. The network sizes and number of training patterns required for the networks are summarized in Table 17.

The performance of the root and the node networks is summarized in Table 16. The last column labeled "Net-tree" is the network tree diagnosis time for each accident. The root network gives an indication of accident condition in fifty-two seconds or less for all the accidents. For thirty-four scenarios, the correct output is obtained in less than thirty seconds. Despite absence of a full diagnosis in these early stages, output of the root network is suitable as an early alarm to notify accident conditions to the operators.

The diagnosis times for the node networks typically show a large variation for each accident. As an example, for ms04a, node 1 network gives appropriate output 145 seconds following malfunction insertion, while node 2 is turned on instantaneously and the root network takes 4 seconds. Thus for this accident, the diagnosis time is 145s. This value is listed in the last column of Table 16.

4.4.7 Conclusions

As can be seen from this table, most of the long diagnosis times occur for the node 1 network. This is because output changes most rapidly across the accidents for this network. An improvement in this network can result in a better performance of the

overall net-tree. Several avenues are available for investigation to obtain such an improvement. Some of these possibilities are enumerated below.

1. The network tree structure shown in Fig. 18 is not very elaborate. Due to this simple structure, the task to be performed by each network in the tree is still complex resulting in long diagnosis times. Thus, further sub-division of the tasks can result in a better diagnostic adviser. For example, the root network can be retained but the node networks replaced by one network per accident. Each of these new networks is trained to recognize one particular accident condition, while simultaneously giving no output in the presence of other accident conditions. This approach would result in forty-one networks in place of the five node networks. There is a corresponding increase in training complexity. However, due to the simplification in the task to be performed by these networks it will be easier to train and thus possibly result in better diagnosis time.

2. The plant variables in the data collected during the simulations are only ninety-seven. This small set of variables may not be adequate or suitable to sufficiently represent the conditions for some accident scenarios. Thus, a larger set of data variables is desirable in view of the range of accident conditions being considered.

3. A major limiting factor in developing large networks, or, group of networks, is their long training time requirements, which can be prohibitively long. ANNs are amenable for implementation on a parallel computer. Training the networks on such a computer can result in significant cuts in training times. Large networks can be trained on MASPAR-II, a massively parallel computer available on Iowa State University campus. A code suitable for use on this machine has been developed (Section 4.7.3).

4. In this work, an information theory analysis (Section 11.2) was not performed on the accident data. Such an approach is useful to decide on the plant variables needed to train the networks. The corresponding reduction in the size of training data will result in lower training time requirements.

5. The ANNs used in this work are simple multilayer feedforward networks. Other network paradigms using interconnections between nodes of the same layer or between nodes separated by a hidden layer (recursive networks) are more powerful. Their suitablility for use in a diagnostic adviser can be investigated in future work. The processing elements in current networks are relatively simple which use a non-linear (sigmoidal) transfer function. Processing elements capable of performaing advanced functions such as integration are available. These enhanced networks have the potential for use in a diagnostic adviser.

4.5 Accelerating the Training Process

by Keehoon Kim and Taher Aljundi

4.5.1 Introduction

Training of ANNs can be supervised or unsupervised. In the supervised training, the network uses information available from the outside environment to adjust the network's parameters such as learning rate, number of iterations, and weights. These adjustments are related to the difference between the network output and the desired output. The unsupervised training, on the other hand, relies only on information available within the inputs to modify the network's parameters and has no known desired outputs. The backpropagation network employs the supervised training to optimize its parameters and the user decides, to a certain extent, the way the network's parameters are modified. The supervised training of a network requires presenting the network with known examples from a training set of input-output data and adjusting the interconnecting weights between the individual nodes until the desired result is obtained. Once the ANN learns the functional relationship between inputs and outputs, subsequent inputs need not be identical to the examples in the training set, and good results are still obtained. However, training of a network to perform a useful task can be computationally expensive (Judd 1987). New methods for increasing learning rates and reducing mathematical and computational complexity are being vigorously investigated (Bartlett 1990, Bartlett and Uhrig 1991a, 1991b). In this area, we have investigated several new techniques to accelerate the convergence and to optimize the architecture of the backpropagation neural network. These techniques include:

1. Rescaling of Variables (Rigler, Irvine and Vogl 1991);
2. Dynamic Learning Rate (Vogl et al. 1988)
3. Dynamic Weight Architecture (DWA).

We used the first two techniques to accelerate the training process, and we developed the third technique to select the optimal network architecture that can learn the given input-output pairs. Selecting the optimal network architecture that is capable of solving the problem will significantly improve the learning speed as well as the generalization capability of the network.

4.5.2 The Backpropagation Method

The backpropagation algorithm (Rumelhart 1986) is a gradient descent method that determines the weights in a multilayer, adaptive neural network. The weights are initially chosen to be small random numbers. Learning is accomplished by successively adjusting the weights based on a set of input patterns. During the learning process, an input pattern is presented to the network and propagated through the network to determine the resulting output. The difference between the resulting output and a predetermined desired output represents an error. A second pattern is then presented and an other error is obtained, and this process continues until all the patterns are presented. The sum of the resulting errors is backpropagated through the network in order to adjust the weights. The learning process continues until the network gives outputs whose RMS of errors is less than a preset value. When an input pattern p is presented to the network, the activation of each node is determined using the sigmoidal activation function

$$O_{pj} = \frac{1}{(1 + exp\{-(\sum w_{ji} O_{pi})\})} \tag{11}$$

where O_{pj} is the activation of node j as a result of presenting the pattern p, W_{ji} is the weight from node i to node j. The summation of errors at the output node j is backpropagated to update the weights in the network according to the following equation :

$$\Delta W_{ji}(n+1) = \mu * \delta_{pj} * O_{pi} + \alpha * \Delta w_{ji}(n) \tag{12}$$

where n is the iteration number, and α is the momentum factor. The error δ_{pj} for an output node j is calculated from the difference between the actual output and the desired output for that particular node

$$\delta_{pj} = (O_{pj} - d_{pj}) * O_{pj} * (1 - O_{pj}) \tag{13}$$

where d_{pj} represents the desired output. The error for a hidden node j is a function of the error of the nodes in the next higher layer connected to j and the weights of those connections.

$$\delta_{pj} = O_{pj} * (1 - O_{pj}) * \sum \delta_{pk} * W_{kj} \tag{14}$$

where the summation is carried over K and k is a node in the next higher layer. Currently, this algorithm is employed with constant values for μ and α. Assuming μ and α are appropriately chosen, the backpropagation process will generally converge with a set of weights that satisfies the criterion imposed by the user. Usually this criterion is satisfied when the RMS of errors of the output nodes falls below a preset value. Unfortunately, when applied to many practical problems, the number of iterations required before convergence can be very large. Consequently, the computations will be complex and very expensive. To overcome this drawback, several modifications of the above algorithm have been investigated to accelerate the convergence process.

4.5.3 Rescaling of Variables in Backpropagation Learning

The derivative of the activation function of the backpropagation network is one cause of the ill-condition of the Backward Error Propagation (BEP) technique. This inherent ill-condition of the BEP process causes the learning algorithm to be painfully slow. The sigmoidal activation function $y(x)$ of each node in the network satisfies the differential equation:

$$y' = y * (1 - y) \tag{15}$$

where x is the sum of weighted inputs to the node. When modifying the weights of the network according to the backpropagation learning algorithm, the factor $y*(1-y)$ occurs once at the first layer preceding the output nodes, and two such factors appear at the second layer preceding, and so on until the input nodes are reached. Because $0 < y*(1-y) < 1/4$, for the sigmoidal function, it is apparent that a major cause of the ill-conditioned nature of the BEP is the gradient element y'. This element cannot exceed 1/4 at the first layer preceding the output nodes, 1/16 at the second layer preceding , 1/64 at the third, and so on. The fact that this element is getting smaller reduces the step size of weights modification as we backpropagate the error from the output layer to the input layer. The rescaling technique suggests that a compensatory correcting factor for the ill-conditioned BEP should be introduced. A random behavior of the derivative can be assumed, and one can employ powers of the expected value $E[y*(1-y)]$, $E[y*(1-y)]^2$,..., $E[y*(1-y)]^n$, where n denotes the nth layer counting backwards from the output. These values are 1/6, 1/36/, 1/216,... The rescaling factors are the reciprocals 6, 36, 216,..., applied as a multiplier of each partial derivative in layers counted backward from the output nodes.

4.5.4 Dynamic Learning Rate

One of the reasons that slows down the convergence of the backpropagation network was discussed in the previous section. Another reason is that the learning constant might not be appropriate for all portions of the error surface. Several modifications of the backpropagation algorithm described by Rumelhart (Rumelhart, Hinton and Williams 1986) can greatly accelerate convergence . Among these modifications is the implementation of a dynamic learning rate so that the algorithm utilizes a near optimum learning rate μ (Vogl et al. 1988). Also the momentum factor a is set to zero when the RMS error does not decrease. Only after the network makes an iteration that reduces the RMS error, a assumes a non-zero value.

The value of μ, which modulates the step size of the change of weights produced by the backpropagation, is sensitive to local shape of the error surface. If a steep V-shaped valley is being followed, too large a value of μ will cause steps that bounce between the two opposite sides of the valley rather following the contour of its bottom. On the other hand, too small a value of μ will prevent the algorithm from making reasonable progress across a long flat slope in the error surface. Choosing a suitable learning rate for a particular problem requires experimenting with different values to see which value achieves the fastest convergence. Unfortunately, even though a one value of μ may be optimum at one stage of the learning process, there is no guarantee that the same learning rate will be appropriate at any other stage of the learning process. To accelerate the convergence of the backpropagation algorithm, a dynamic learning rate technique with the appropriate changing of the momentum factor a was used. The learning rate μ and the momentum factor α were varied according to whether or not an iteration decreases the RMS error. If an update results in reduced RMS error, μ is multiplied by a factor $\phi > 1$, and α is set to a non zero value for the next iteration. If a step produces a network with an RMS error more than the previous value, all changes to the weights are rejected, μ is multiplied by a factor $\beta < 1$, α is set to zero, and the step is repeated.

4.5.5 Dynamic Weight Architecture

Optimizing the network architecture for a specific problem is a vital part of the training objective. Too many nodes with too many weights will significantly slow down the learning process and will also make it easy for the network to fit the noise of the training data, thus failing to generalize. The minimal network strategy states that if several nets fit the data equally well, the simplest one will on average provide the best generalization (Weigand, Rumelhart and Huberman 1991). The simplest network that fits a particular set of data is the network with the least number of nodes and the

least number of weights. A traditional method of determining the optimal network architecture is by starting with a very large network architecture and eliminating the least important nodes as the learning process proceeds (Weigand, Rumelhart and Huberman 1991). This approach, however, requires a significant computer time because there is no limit on the initial network size. Our approach, on the other hand, starts with the smallest possible network architecture. Weights and nodes are then added according to their importance as the training process proceeds. Since the network is established using the least unit of a weight connection, a resultant network will have a sparsely connected architecture. The DWA is a different scheme from DNA (Dynamic Node Architecture) using a node as the least unit of connection.

Start with a fully connected network with only one hidden node in each hidden layer. The number of input and output nodes is usually pre-determined by the nature of the specific problem. Typically, since we started with a very small architecture, the network will not be able to learn the data set. After a certain number of iterations have been completed without any reduction in the RMS error, we pick the most important node in each layer according to the following:

The importance of an output node is given by

$$I(j) = E(O_j) * SD(O_j)(6) \tag{16}$$

where:

$I(j)$ is the importance of output node j.

$E(O_j)$ is the expected value of the outputs of the node j.

$SD(O_j)$ is the standard deviation of the outputs of the node j.

The importance of an input or a hidden node i is given by

$$I(i) = E(O_i) * SD(O_i) * [\sum I(j) * W_{ji}] \tag{17}$$

where W_{ji} is the weight connecting the i-th node to the j-th node in the above layer,

and the summation is over all the nodes in the layer above. The DWA then connects the most important nodes via weights with arbitrary initial values of .001. If the important node in a hidden layer is already connected to the important nodes in the layers above and below, the DWA adds a new node in that hidden layer and connects it to the important nodes in the layers above and below. The process is repeated until the convergence criterion is satisfied, which is that the RMS error falls below a preset value. The resulting network architecture will be the smallest network that can learn that specific data set in a reasonable number of iterations. Hence, this network will have generalization capabilities better than any other network trained with the same data set.

4.5.6 Results

Dynamic Learning Rate and Rescaling Factor Results

The rescaling technique along with the dynamic learning rate were implemented into the backpropagation algorithm with varying results. The dynamic learning rate has significantly increased the convergence speed of the network, whereas the rescaling factor has less impact on the convergence speed of the network. Table 18 summarizes the results of implementing both the dynamic learning rate and the rescaling technique into the same backpropagation architecture to solve the XOR problem.

Figures 19 and 20 compare the results of applying the backpropagation neural network with dynamic learning rate and with constant learning rate to solve the Gaussian Distribution Separator (GDS) problem. In this problem we have two overlapped data sets, and we wish to separate them. The same network architecture, 5 x 18 x 1, was used in both cases. Figure 21 shows the way the learning constant μ is changing during the training process.

Table 18: The momentum coefficient α=0.5 if the RMS error is decreasing, zero otherwise. Convergence criterion is satisfied when the RMS error is $\leq$ 0.006. Backpropagation network architecture is 2 x 3 x 1.

	Technique Used	# of Iterations Before Convergence
1.	Backpropagation with learning constant = 0.4	4942
2.	Backpropagation with Rescaling factor = 6	4500
3.	Backpropagation with Dynamic learning rate	180
4.	Backpropagation with Dynamic learning rate and Rescaling factor = 6	106
5.	Backpropagation with Dynamic learning rate and Rescaling factor = 5	108

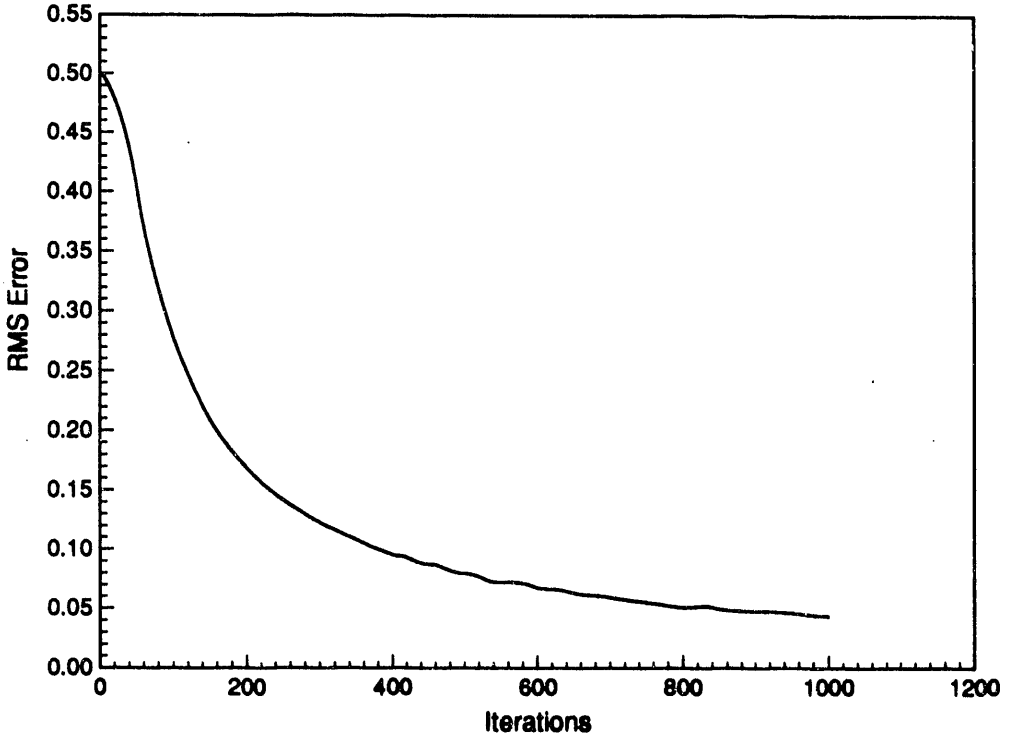

Figure 19: RMS error vs. number of iterations using the backpropagation network with dynamic learning rate

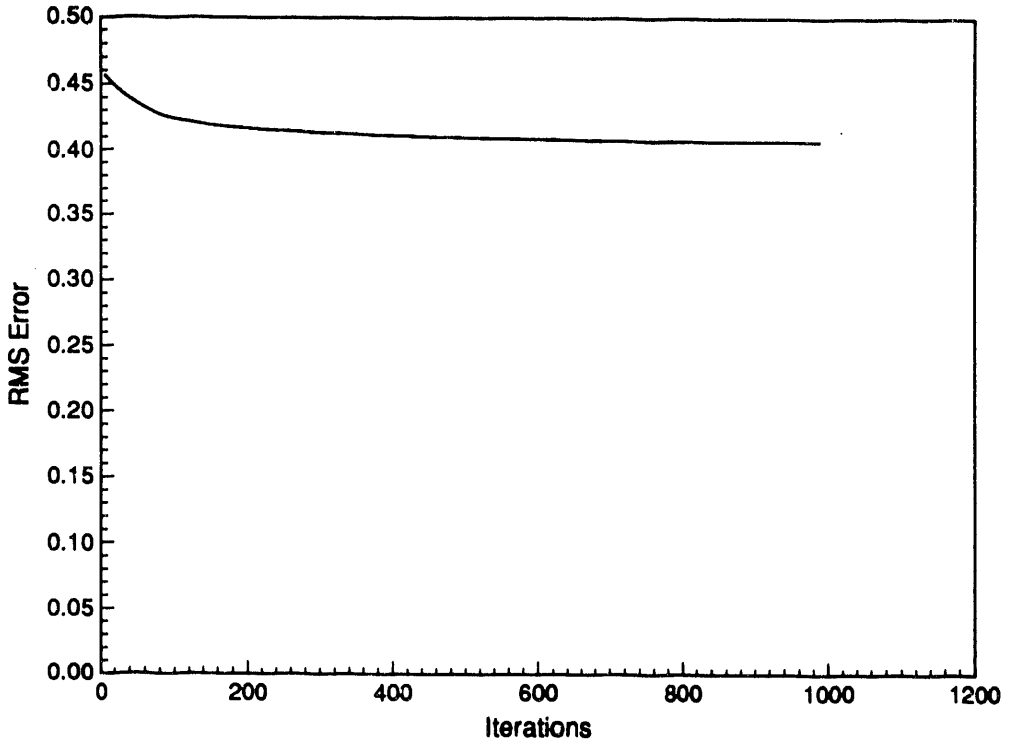

Figure 20: RMS error vs. number of iterations using the backpropagation network with constant learning rate

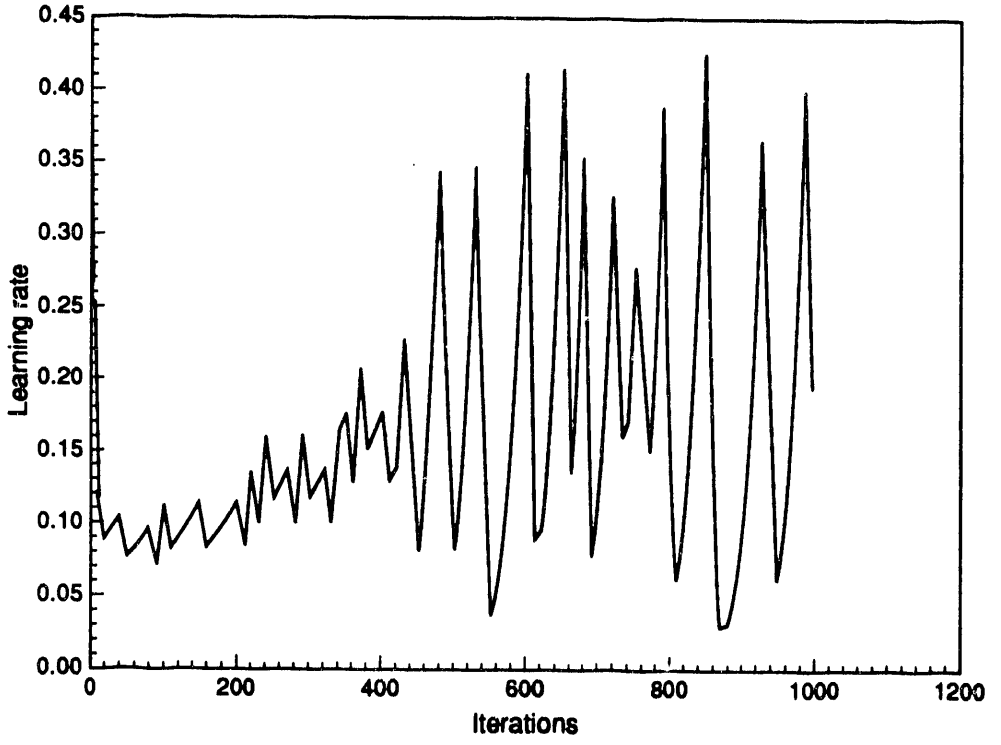

Figure 21: The dynamic learning rate technique optimizes the learning rate according to the local topography of the error surface.

Dynamic Weight Architecture Results

We have applied the DWA technique to find the optimal architecture to solve the XOR problem. The resulting architecture was 2 x 3 x 1 which agrees with the results of the Dynamic Node Architecture (Bartlett and Basu 1991). Actually this network architecture gives generalizing capabilities better than any other architecture when tested with noisy data. The DWA was also applied to find the optimal network architecture to solve the GDS problem, and the resulting architecture was 5 x 15 x 1.

4.5.7 Conclusions

We have implemented the convergence accelerating techniques into the backpropagation learning algorithm. These techniques have significantly reduced the training complexity and the computation time required to train the network during the development of the nuclear power plant fault diagnostic adviser. This work shows the feasibility of the DWA scheme as a network architecture optimizing technique. This technique, however, will be further investigated, and more variables will be included in the node importance function.

4.6 Diagnostic Error Prediction

by Keehoon Kim and Taher Aljundi

4.6.1 Introduction

Many scientists, as well as the public, are concerned with nuclear power plant safety. The diagnosis of nuclear power plant transients must be swift and accurate. Early and accurate transient diagnosis can give the reactor operators extra time to formulate and perform the necessary actions to prevent a transient from developing into a potentially dangerous accident. ANNs have many characteristics that make them a suitable tool to achieve fast and accurate diagnosis of nuclear power plant transients. These characteristics include fault and noise tolerance, generalization capabilities, and the ability to quickly respond to changes in a plant's conditions (Bartlett 1990). One disadvantage of ANNs, however, has been the relative difficulty of assigning error bounds on its results. Resolving the uncertainties associated with the ANN diagnoses is a very important step towards assuring the reliability of the adviser's diagnoses. In this work, an ANN fault diagnostic adviser was developed to diagnose nuclear power plant transients. The ANN adviser was developed by training a backpropagation neural network to diagnose ten distinct nuclear power plant transients. This ANN was trained using computer generated data obtained from the San Onofre Nuclear Power Station training simulator (James and Rogers 1992). The data simulates the plant's conditions during the ten distinct transients. We also applied the Stacked Generalization technique (Wolpert 1992) to estimate the predicted error associated with the adviser's diagnoses. When a transient or an accident occurs in a nuclear power plant, the ANN adviser will inform the operators in the control room about the cause of the system instability in a timely manner. At the same time the adviser will also provide them with a prediction of its diagnosis accuracy. The development

of an ANN nuclear power plant fault diagnostic adviser that can classify transients and provide error bounds on its diagnoses is another new contribution of this project to the science of ANNs and its applications.

4.6.2 Neural Networks and Stacked Generalization

The lack of validation capabilities of the ANNs' results has been a major limiting factor on the applications of these techniques. Stacked generalization, however, is a method that can be used to address the validation problem and can be implemented in several different ways (Wolpert 1992). When used with multiple generalizers, it can provide an estimate of the average generalizing accuracy (Wolpert 1990c, 1990d) of each individual generalizer. Hence, one can pick the generalizer that has the highest estimated generalization accuracy to map a particular data set. When used with a single generalizer, it can provide an estimate of the error associated with the outputs of a neural network that has been recalled on novel input data. In this work we have used the backpropagation neural network as a single generalizer to develop a fault diagnostic adviser that can classify nuclear power plants instabilities. The stacked generalization was used along with the backpropagation generalizer to assign error bounds to the adviser's outputs.

4.6.3 Stacked Generalization

Stacked generalization is a technique whose purpose is to achieve the highest possible generalization accuracy. In this work, we have applied this technique for estimating the errors of a single generalizer when working on a particular learning set. The stacked generalization can also be used to correct for these errors and, therefore, maximize the generalization accuracy.

The first step in employing stacked generalization is choosing a set of (r) partitions, each of which splits the learning set (L) into two (usually disjoint) sets. Label such

a set of partitions as L_{ij}, where $0 \leq i \leq r$, and $j \in \{1,2\}$. We have arbitrarily chosen the cross-validation partition set (Li 1985), where $r = m$ and m is the number of patterns in the training set. For all i, L_{i2} consists of a single pattern of L, the corresponding L_{i1} consists of the rest of L, and $L_{i2} \neq L_{j2}$ for $i \neq j$.. Since $r = m$, this last requirement of distinctness of the L_{i2} means that the set of all L_{i2} covers L. Wolpert defines the original learning set L as the "level 0" space. So any network when applied directly to L in the "level 0" space is called a "level 0" generalizer, and the original learning set L is called a "level 0" learning set. Figure 22 describes the partition used in stacked genealization. A learning set L is represented by the circle, and a partition of L into two portions is also shown. Given this partition, we train the "level 0" generalizer on the portion $\{L-(x,y)\}$. Then we ask the "level 0" generalizer the question x, and note both its guess, g and the vector (Euclidean distance) from x to its nearest neighbor in $\{L-(x,y)\}$, k. Since the "level 0" generalizer has not been trained with the pair (x,y), g will generally differ from y. Therefore when the question is x, and the vector from x to the nearest neighbor in the learning set is k, the correct answer differs from the "level 0" generalizer guess by $(g-y)$. This information can be cast as input-output information in a new space, the "level 1" space. The input is the pair (x,k) and the output is $(g-y)$. Choosing other partitions of L gives other such points. Taken together, these points constitute a "level 1" learning set L'. We then train a "level 1" generalizer on the "level 1" training set. We now ask the "level 0" generalizer the question q. Then we take the pair q and the vector from q to the nearest neighbor in L, and feed that pair as a question to the "level 1" generalizer which has been trained on L'. This "level 1" generalizer's guess is our guess for "level 0" generalizer's error in guessing what output corresponds to q. This is a brief description of one of the applications of the theory of stacked generalization. There are several other applications and details concerning the theoretical and empirical

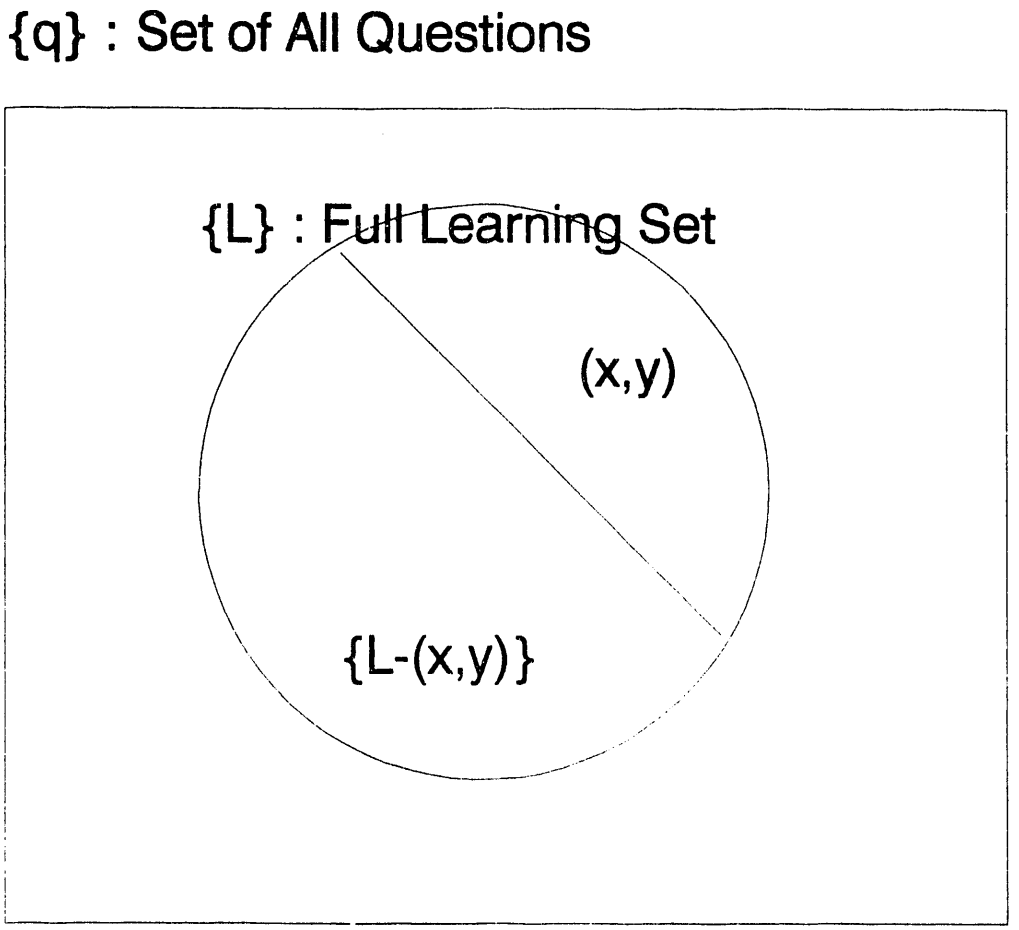

Figure 22: Description of the partition in the stacked generalization to predict the error of a "level 0" generalizer. The rectangle $\{q\}$ represents the set of all questions. The circle $\{L\}$ represents the full learning set. $\{l\}$ is partitioned into two subsets,$\{L-(x,y)\}$ and (x,y). A "level 0" generalizer is trained using $\{L-(x,y)\}$ and then ask (x,y). The error, y minus the guess of the "level 0" generalizer, will constitute the desired output of a "level 1" generalizer. The input to the "level 1" generalizer is the question x plus the vector from x to its nearest neighbor in $\{L-(x,y)\}$, k.

issues of stacked generalization. It is important to note that many aspects of stacked generalization are, at present, not completely understood. For example, there are no rules to how many inputs there should be in "level 1" generalizer. In practice, one must rely on prior knowledge to make intelligent guesses for how to set the details of stacked generalization.

4.6.4 Method

A backpropagation neural network with 33 x 22 x 10 x 4 architecture was used to diagnose the ten distinct transients. Thus, the network has 33 nodes in the input layer, 22 in the first hidden layer, 10 in the second hidden layer , and 4 in the output layer. The 33 input nodes represent 33 of the plant variables, which are shown in Table 19, and four output nodes distinguish each of the 10 transient conditions with a distinct 4-bit code. The two hidden -layers architecture was found to be the most suitable for this particular problem among trying several different architectures that were investigated. Training of the network was accomplished in steps (Bartlett, Uhrig 1992, 1991b). The initial training set composed of 20 patterns, two from each transient. These two patterns were the first pattern, corresponding to normal operating conditions at time = 1s, and the last pattern in each transient, corresponding to the time during the transient's onset. In all of the 10 transients, the data collection was terminated around time = 600 seconds. Training on this data was performed until an RMS error of .0135 was attained. The next step was to recall the network on the entire data set for each of the ten transients. The RMS error obtained from the recall set was plotted against time for each transient. There were several peaks where the RMS error was very high. These peaks correspond to patterns very different from those chosen in the initial training set. These patterns were included in the training set of the network for the next training phase. The training, recalling, and expanding of the training set was repeated, and the cycle continued until all peaks in the recall

Table 19: Nuclear power plant variables used for the San Onofre diagnoses described in the text.

1)	Power (flux)
2)	Average Temperature (Degrees F)
3)	Hot Leg 1 Temperature (Degrees F)
4)	Cold Leg 1A Temperature (Degrees F)
5)	Cold Leg 1B Temperature (Degrees F)
6)	Hot Leg 2 Temperature (Degrees F)
7)	Cold Leg 2A Temperature (Degrees F)
8)	Cold Leg 2B Temperature (Degrees F)
9)	Pressurizer Pressure (psia)
10)	Pressurizer Level (%)
11)	Pressurizer Temperature (Degrees F)
12)	Steam Generator 88 Narrow Range Level (%)
13)	Steam Generator 88 Water Level (%)
14)	Steam Generator 88 Feed Water Flow (gpm)
15)	Steam Generator 88 Feed Water Flow (Lb/sec)
16)	Steam Generator 88 Steam Flow (Lb/sec)
17)	Steam Generator 88 Pressure (psia)
18)	Steam Generator 89 Narrow Range Level (%)
19)	Steam Generator 89 Water Level (%)
20)	Steam Generator 89 Feed Water Flow (gpm)
21)	Steam Generator 89 Feed Water Flow (Lb/sec)
22)	Steam Generator 89 Steam Flow (Lb/sec)
23)	Steam Generator 89 Pressure (psia)
24)	Containment Pressure (psig)
25)	Containment Temperature (Degrees F)
26)	Pressurizer Relief Steam Flow
27)	Pressurizer Relief Liquid Flow
28)	Core Inlet Flow
29)	Saturation Margin
30)	Surge Line Temperature (Degrees F)
31)	Source Range Counts (counts per second)
32)	Reactor Vessel Head Level
33)	Reactor Vessel Plenum Level

set fell below .1 RMS error. The final training set consisted of 113 patterns and is equivalent to L in Fig. 22. The next step was to apply the cross-validation partitioning on this final training set. The "level 0" generalizer was the backpropagation neural network with 33 x 22 x 10 x 4 architecture. The "level 1" training set was composed of the "level 0" input (x) plus the vector from (x) to its nearest neighbor k. The "level 1" desired output was the difference between the "level 0" guess and the desired "level 0" output. The "level 1" network was another backpropagation neural network with 66 x 30 x 20 x 10 x 4 architecture. Again this architecture was chosen after several attempts to find the smallest architecture which could be trained with this particular data set and still give good results. The training of these networks was carried out and then the stacked generalization was applied as explained in the above section.

4.6.5 Results

Table 20 shows that the fault diagnostic adviser is capable of classifying the ten distinct transients and providing error bounds on its classification in a timely manner. The second column in Table 20 describes the time needed for the network to make a decision about what particular transient is taking place. The third column represents the time spent before the accuracy of the classification (the confidence level) reached an acceptable level. The acceptable level was arbitrarily taken as the time spent before the estimated error on the diagnoses drops below 0.1 RMS. Notice that the network responds very rapidly to the changes in the plant conditions. One exception is the Trip of a Single Reactor Coolant Pump transient, where the time to classification is 60 seconds.

Figures 23 through 32 represent the diagnoses of the ten distinct transients and the error predictions on these diagnoses. The peaks at the beginning of each of the diagnoses figures indicate that the adviser has detected an instability in the plant's

Table 20: List of the San Onofre 10 distinct scenarios and the associated diagnosing time and the time needed after correct diagnosing to assure the diagnosis.

Name of transient	Time needed after the initiating event to diagnose the transient(s)	Time needed after correct diagnosis to assure the diagnosis(s)
1. Turbine trip/reactor trip	30	1
2. Loss of main feedwater pumps	3	0
3. Closure of both main steam isolation valves	28	0
4. Trip of all reactor coolant pumps	2	0
5. Trip of a single reactor coolant pump	62	0
6. Turbine trip from 50% power.	1	Fail to make an assured diagnosis.
7. Loss of coolant accident with loss of off-site power.	14	0
8. Main steam line break	4	31
9. Stuck open pressurizer safety valve with high pressure injection inhibited diagnosis	Immediate diagnosis	63
10. Single turbine governor valve closure	16	0

conditions but it did not decide on the cause of this instability. After few seconds, the adviser has decided that this particular transient is responsible for the instability in the plant. The error prediction figures can be interpreted as the level of confidence in the decision obtained from the diagnoses figures. The value (1- the predicted error) can be considered as the level of confidence in the diagnosis. Column 3 in Table 20 indicates the time from the point where a correct diagnosis has been made in the diagnoses figures, RMS error drops below 0.1, until the confidence level in the diagnosis is reasonable. The time for this reasonable confidence level was taken as the time when the predicted error drops below another arbitrary value of 0.1.

Figure 28 indicates that although the classification of the Turbine Trip From 50% Power is correct, the network confidence in this classification is low. The estimated error on diagnosing this transient is large for the entire time period. This indicates that the diagnosis is unreliable even though it is correct. That looks like an unwanted result of the stacked generalization. But it justifies our experiment in the view point of the generalization characteristic. Since all the data were collected from 100% power except for this particular transient where the data were collected from 50% power, the network was not able to make an assured diagnosis on this particular transient. The level of confidence in the diagnosis of this transient can be increased by training the ANN adviser with more transients from this power level. Figure 31 shows the results obtained for the Stuck Open Pressurizer Safety With High-Pressure Injection Inhibited transient. This transient is a computer simulation of TMI-2 type accident. The ANN adviser was able to correctly diagnose the transient immediately; however, the adviser took about 65 seconds to assure its diagnosis.

4.6.6 Conclusion

This chapter has demonstrated the feasibility of using ANN technology coupled with stacked generalization technique as a diagnostic tool for nuclear power plant tran-

sients. The stacked generalization technique was used to measure the accuracy of the diagnoses. The results of the stacked generalization agreed with what we would expect. The diagnoses of nine of the transients was correct and accurate, while the accuracy of diagnosing the Turbine Trip from 50% Power transient was low because this was the only transient where the data was collected from 50% power level. Future work should include more transients and should test the performance of a prototype of such an adviser in a nuclear power plant. Implementing such an adviser in a nuclear power plant will provide continuous and accurate monitoring of the plant's integrity. It will also provide fast and reliable diagnosis of any system instability and therefore will significantly enhance the safety of nuclear power plants.

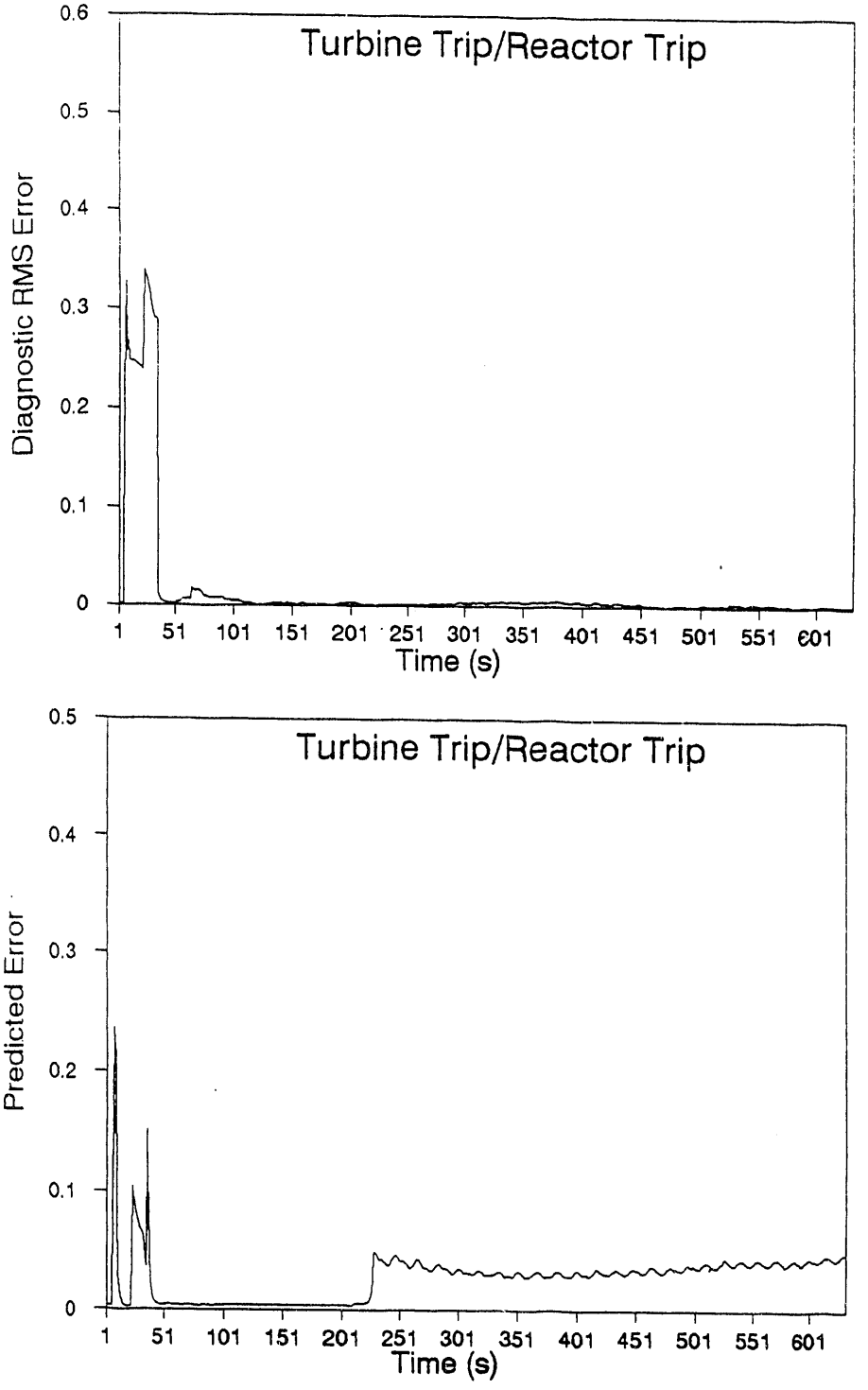

Figure 23: Time vs. diagnostic RMS error of the ANN adviser (above) and the predicted error (below) for the Turbine Trip/Reactor Trip transient. Notice that the value one minus the predicted error can be interpreted as the level of confidence in the diagnosis.

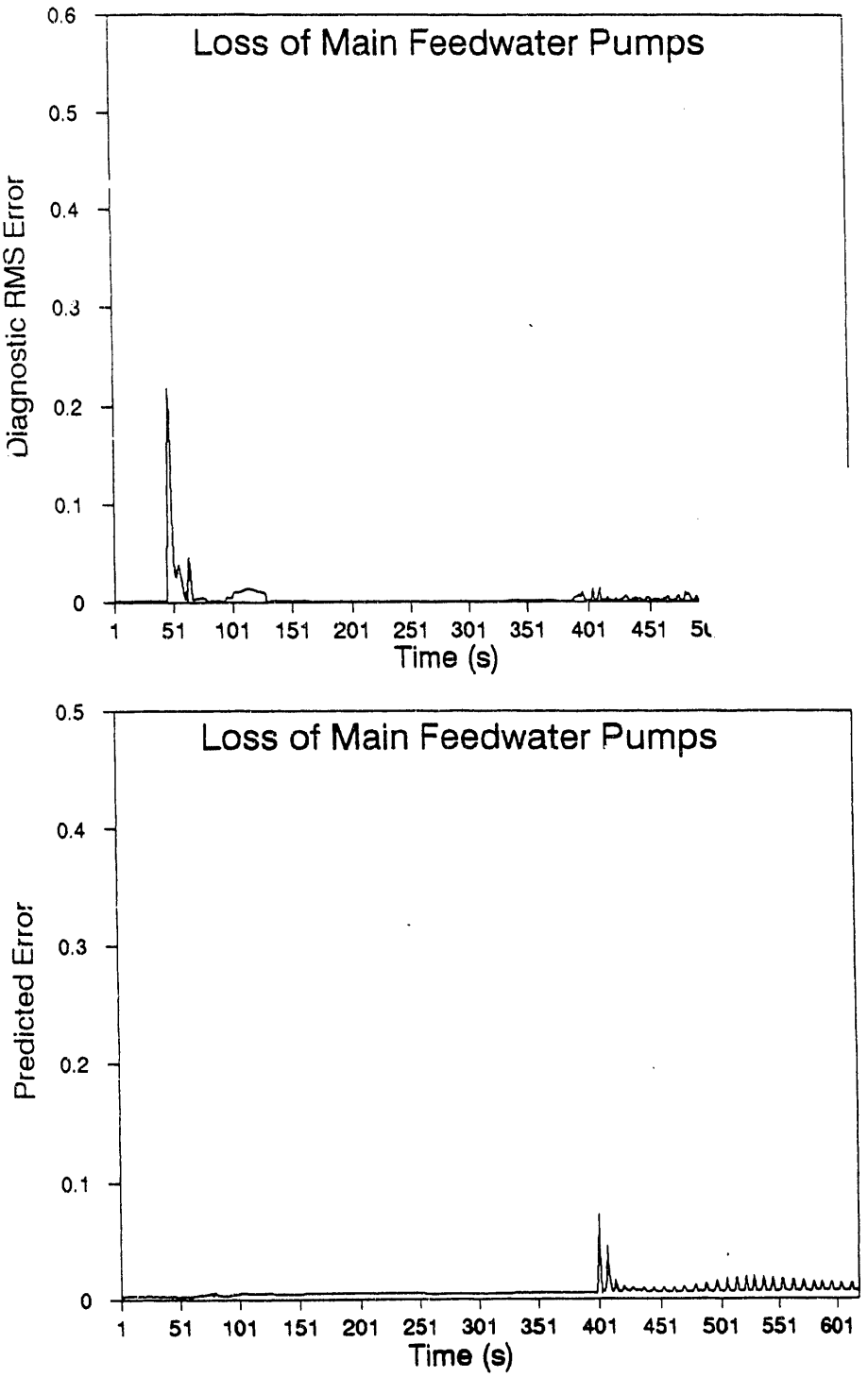

Figure 24: Time vs. diagnostic RMS error of the ANN adviser (above) and the predicted error (below) for the Loss of Main Feedwater Pumps transient. Notice that the value one minus the predicted error can be interpreted as the level of confidence in the diagnosis.

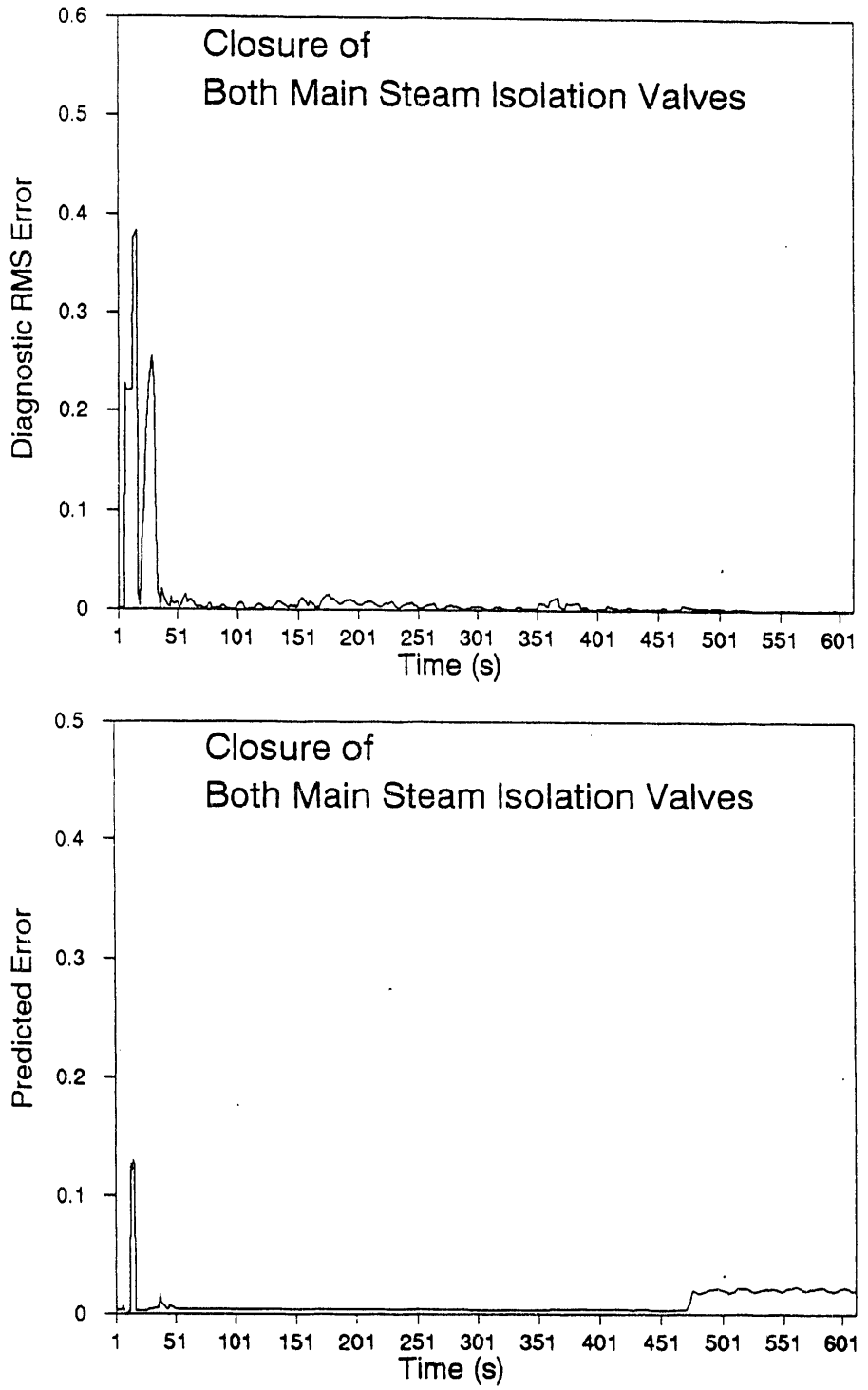

Figure 25: Time vs. diagnostic RMS error of the ANN adviser (above) and the predicted error (below) for the Closure of Both Main Steam Isolation Valves transient. Notice that the value one minus the predicted error can be interpreted as the level of confidence in the diagnosis.

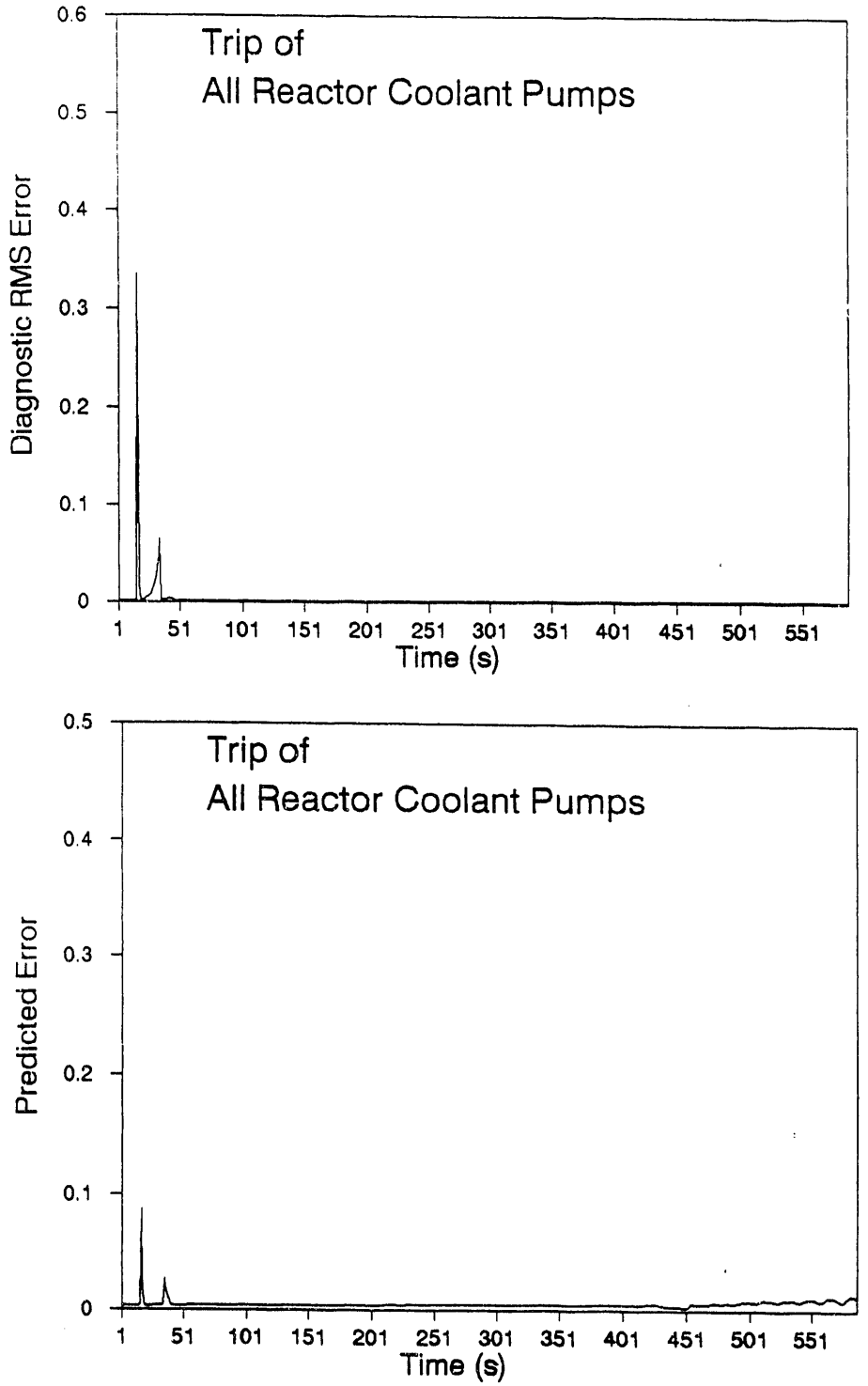

Figure 26: Time vs. diagnostic RMS error of the ANN adviser (above) and the predicted error (below) for the Trip of All Reactor Coolant Pumps transient. Notice that the value one minus the predicted error can be interpreted as the level of confidence in the diagnosis.

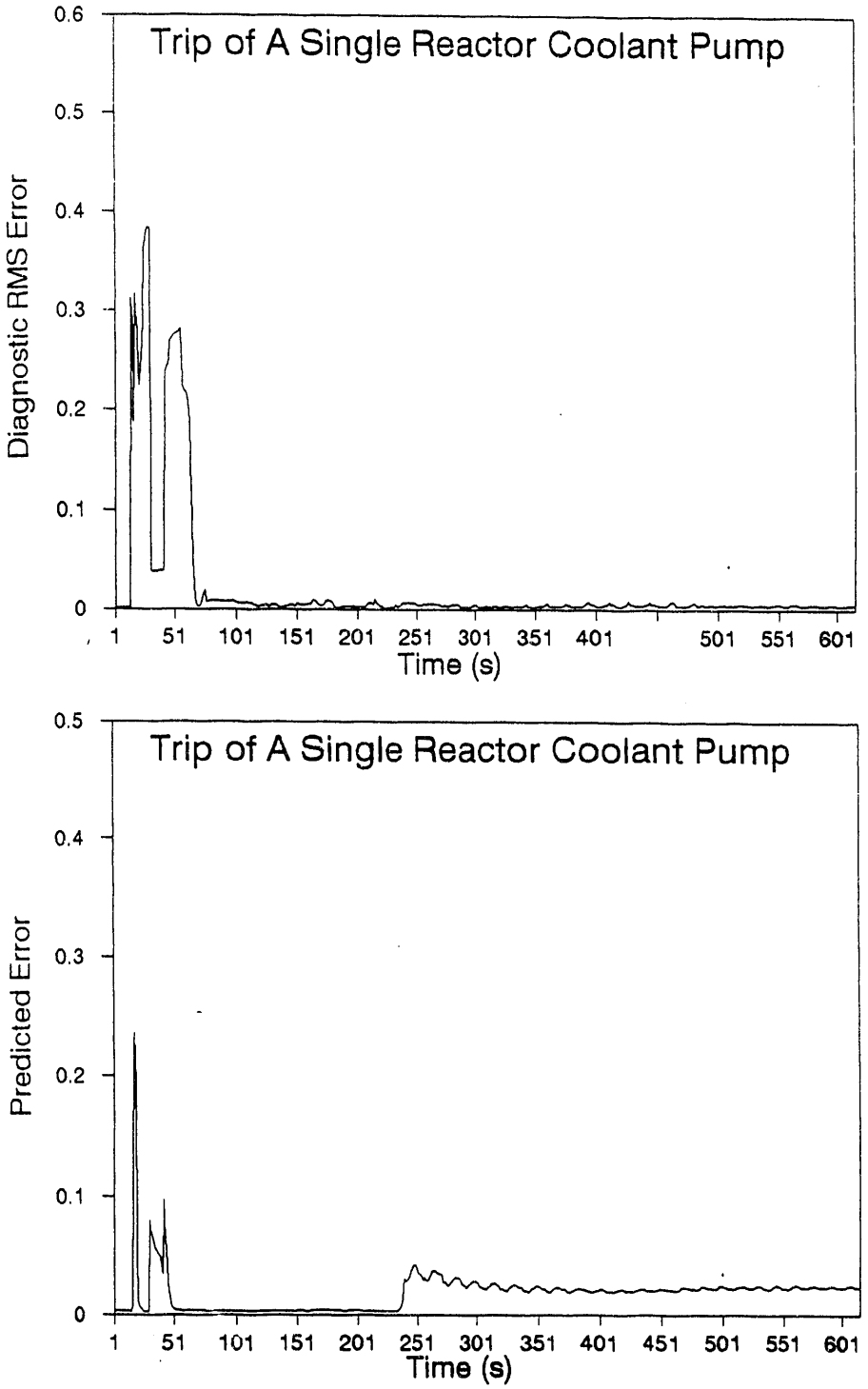

Figure 27: Time vs. diagnostic RMS error of the ANN adviser (above) and the predicted error (below) for the Trip of a Single Reactor Coolant Pump transient. Notice that the value one minus the predicted error can be interpreted as the level of confidence in the diagnosis.

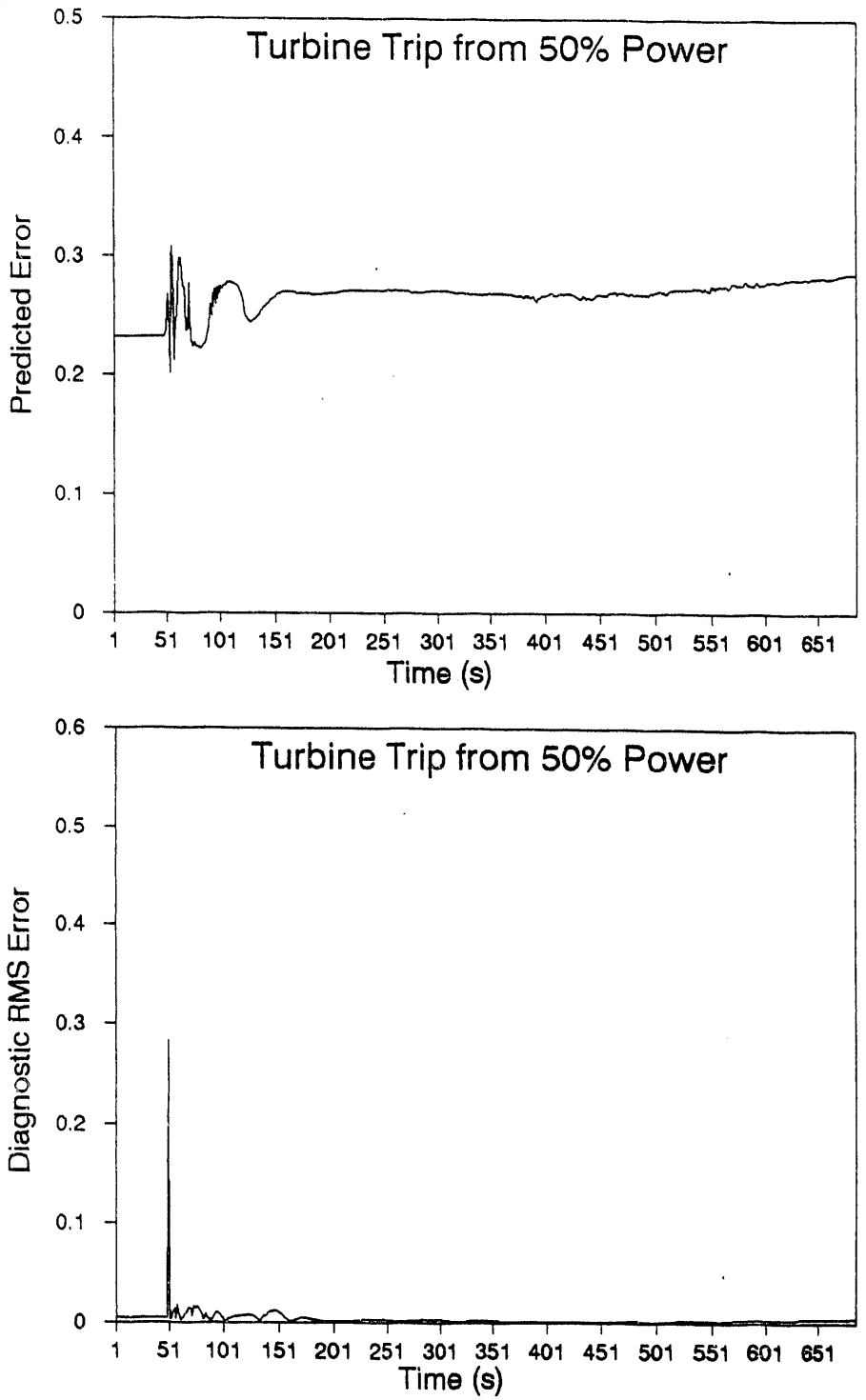

Figure 28: Time vs. diagnostic RMS error of the ANN adviser (above) and the predicted error (below) for the Turbine Trip from 50% Power transient. Notice that the value one minus the predicted error can be interpreted as the level of confidence in the diagnosis.

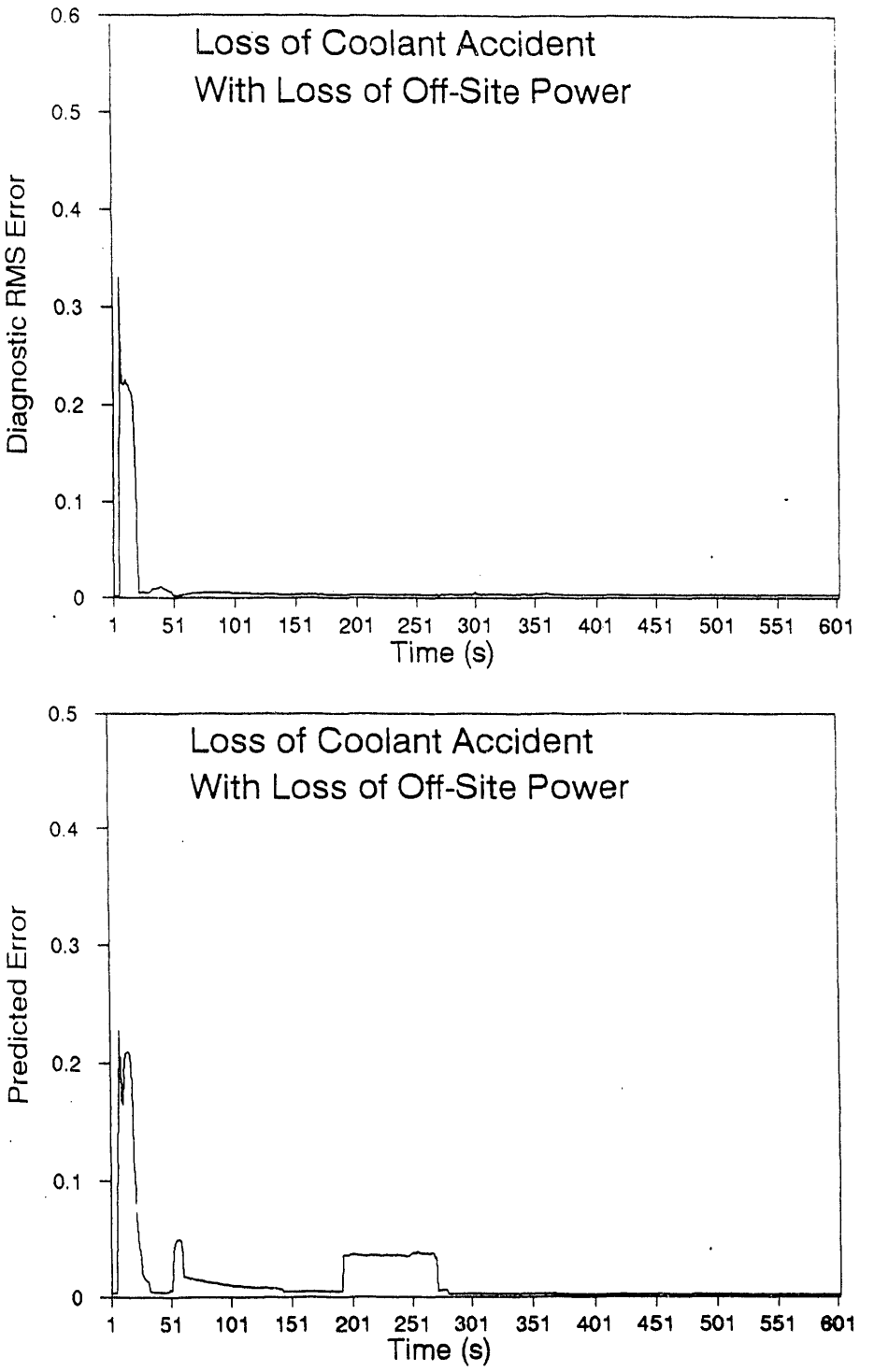

Figure 29: Time vs. diagnostic RMS error of the ANN adviser for (above) and the predicted error (below) the Loss of Coolant Accident with Loss of Off-Site Power transient. Notice that the value one minus the predicted error can be interpreted as the level of confidence in the diagnosis.

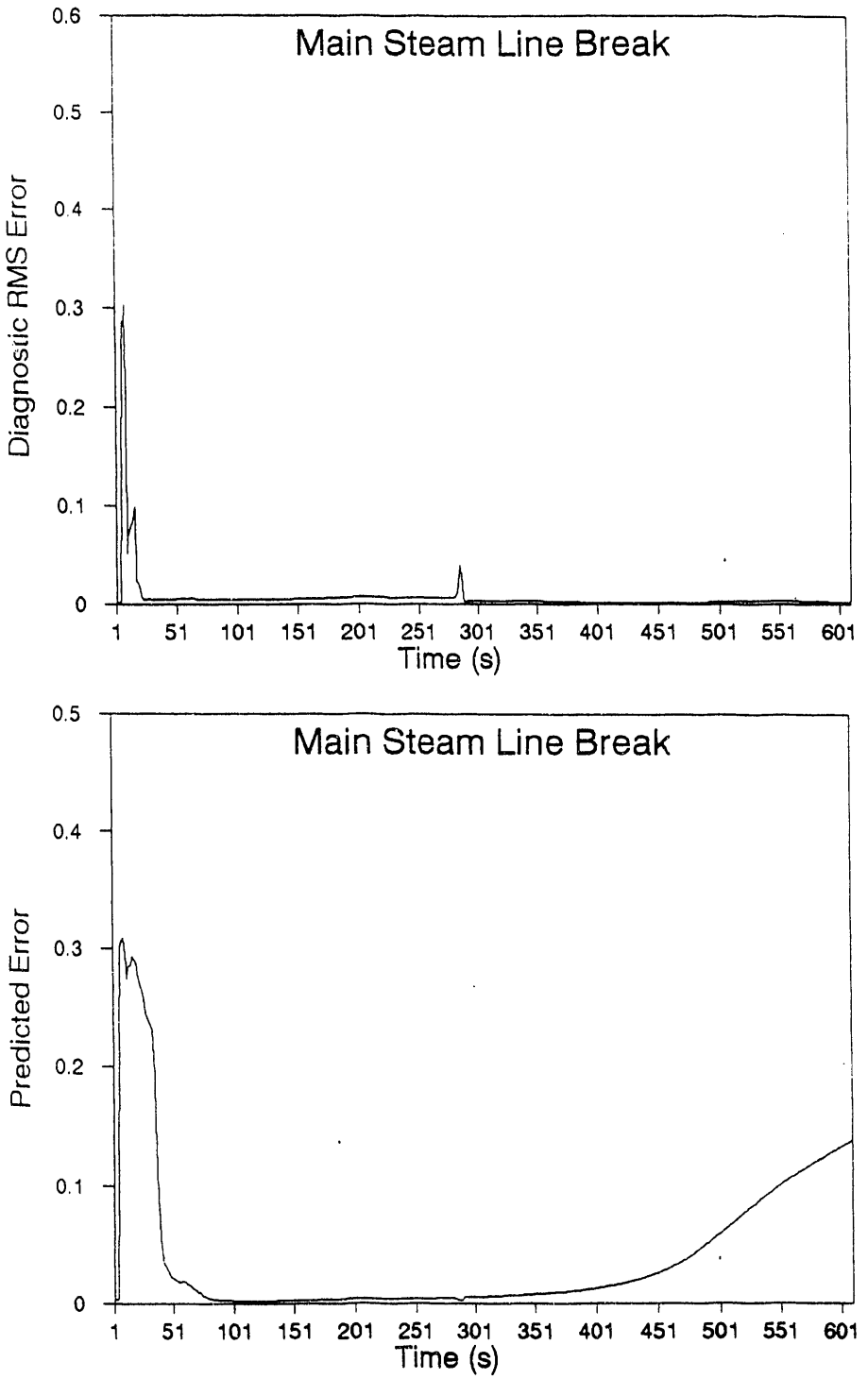

Figure 30: Time vs. diagnostic RMS error of the ANN adviser for (above) and the predicted error (below) the Main Steam Line Break transient. Notice that the value one minus the predicted error can be interpreted as the level of confidence in the diagnosis.

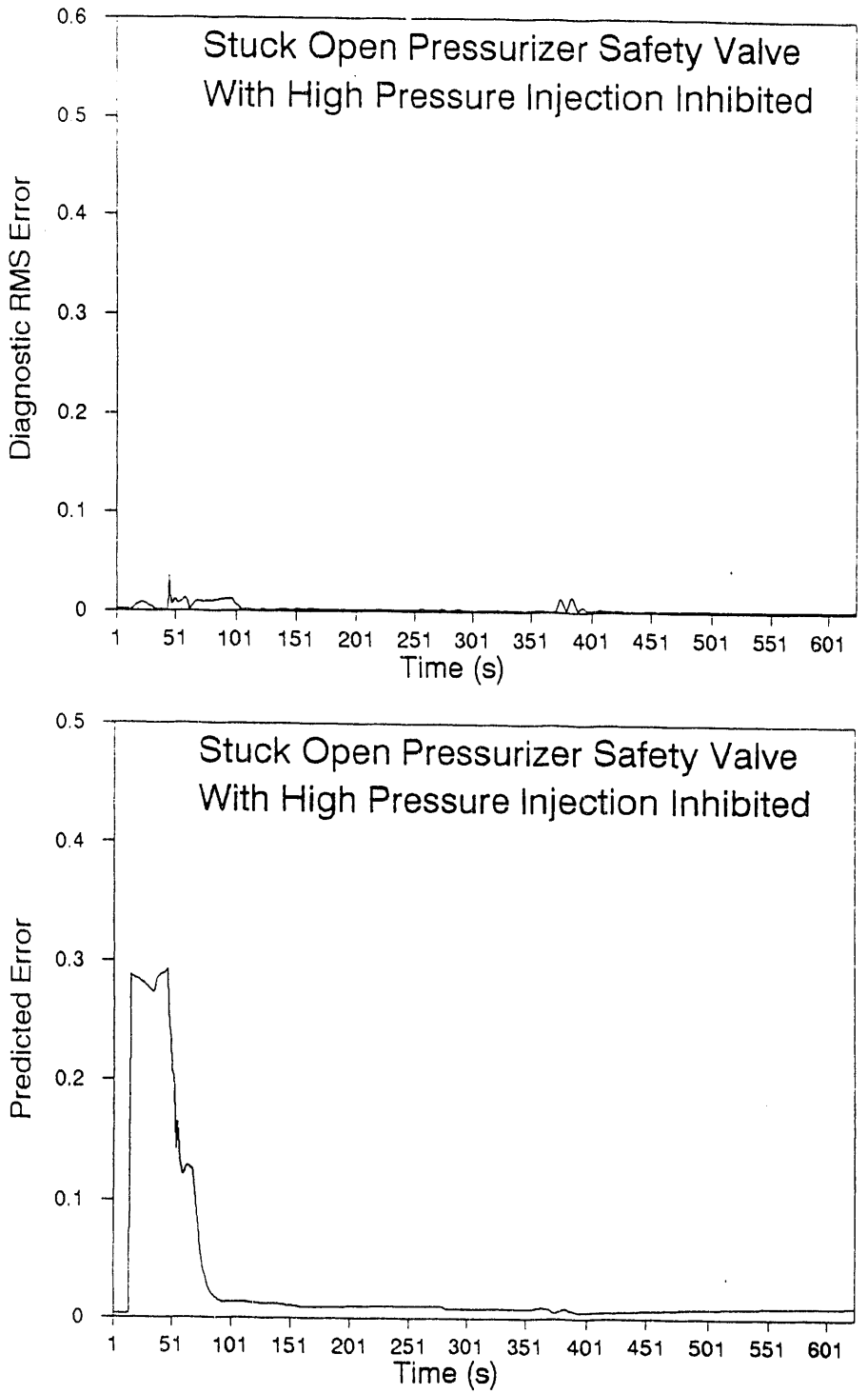

Figure 31: Time vs. diagnostic RMS error of the ANN adviser (above) and the predicted error (below) for the Stuck Open Pressurizer Safety Valve With High Pressure Injection Inhibited transient. Notice that the value one minus the predicted error can be interpreted as the level of confidence in the diagnosis.

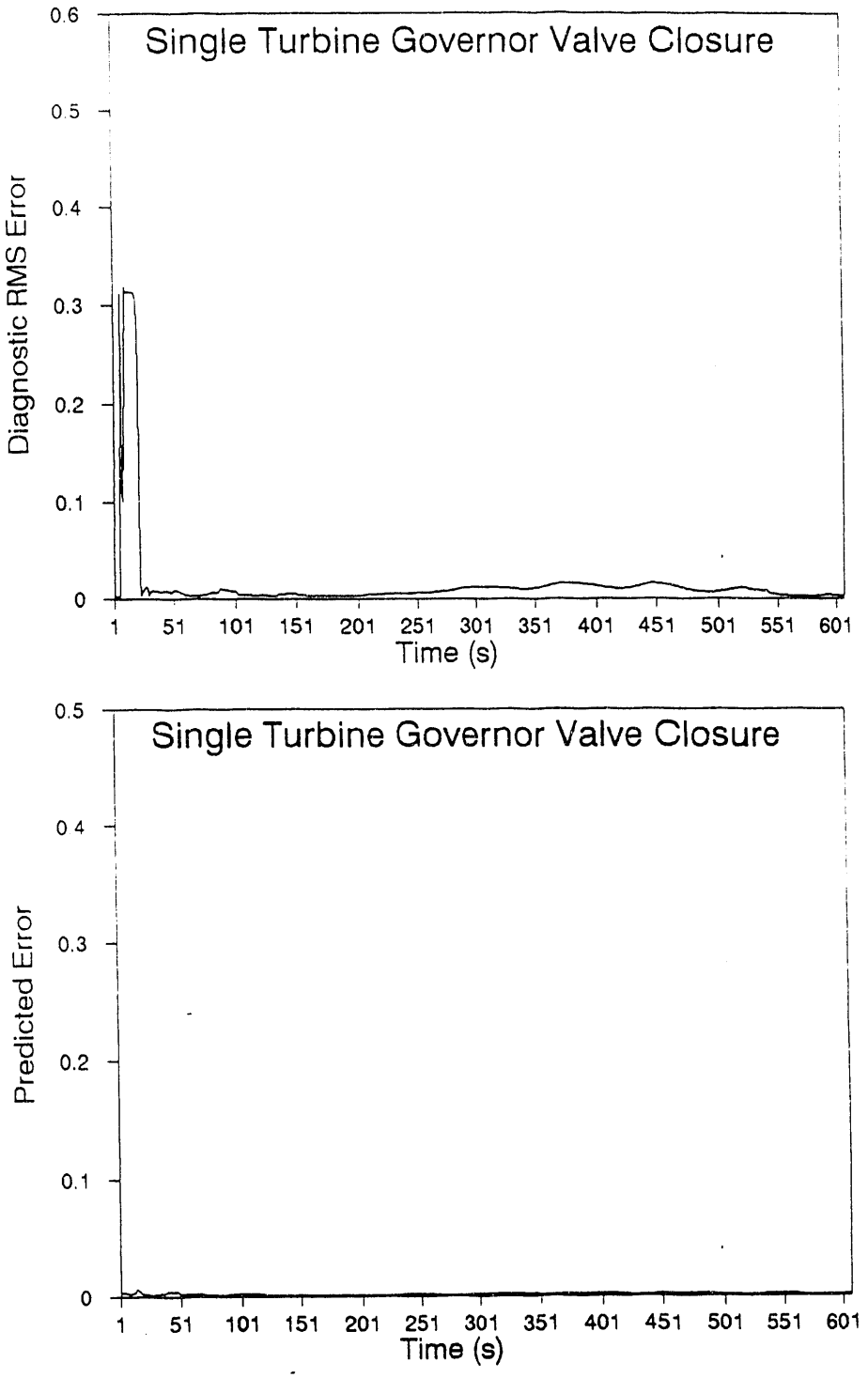

Figure 32: Time vs. diagnostic RMS error of the ANN adviser (above) and the predicted error (below) for the Single Turbine Governor Valve Closure transient. Notice that the value one minus the predicted error can be interpreted as the level of confidence in the diagnosis.

4.7 Selections from thesises completed under the current funding

This section contains the main bodies of the various thesises complted by graduate students under supervision of the PI during the current funding. The work undertaken in these thesises are either directly connected to this project (Sections 4.7.1, 4.7.2 & 4.7.4) or will contribute to the project different methodologies developed for other work (Section 4.7.3).

Theses removed and cycled separately.

10 Articles Published During the Period

This section contains the journal papers, abstracts, and conference papers that were published or accepted for publication during the current funding.

Removed and cycled separately.

END

DATE
FILMED
5/12/93