Defense Technical Information Center Compilation Part Notice

This paper is a part of the following report:

- *Title:* Technology Showcase: Integrated Monitoring, Diagnostics and Failure Prevention. Proceedings of a Joint Conference, Mobile, Alabama, April 22-26, 1996.

- *To order the complete compilation report, use:* AD-A325 558

The component part is provided here to allow users access to individually authored sections of proceedings, annals, symposia, etc. However, the component should be considered within the context of the overall compilation report and not as a stand-alone technical report.

A NON-REDUNDANT SENSOR VALIDATION SCHEME FOR TRANSIENT AND STEADY-STATE CONDITION MONITORING

Michael J. Roemer

Stress Technology Incorporated
1800 Brighton-Henrietta Town Line Rd.
Rochester, New York 14623
Phone: (716) 424-2010
Fax: (716) 272-7201

Abstract: This paper presents a sensor validation scheme capable of detecting failed sensor hardware without sensor redundancy and during non-steady state monitoring conditions. The technical approach utilizes neural networks and fuzzy logic to accomplish the desired goal. Neural networks are used to recognize the non-linear, inter-relationships between the different types of sensors used in a transient or steady-state measurement environment. Fuzzy logic is used to pre- and post-process the measurement data in order to determine general characteristics about the state of the process being monitored. Different types of neural network architectures were developed and tested to determine their suitability to solving this problem. The feasibility of the method was proven through computer simulation utilizing gas turbine engine data as input to the validation system.

Key Words: Sensor Validation, Condition Monitoring, Neural Networks, Fuzzy Logic, Artificial Intelligence, Transient Conditions, Non-Redundant Sensors, Health Monitoring.

Introduction: Integrated sensor systems play a major role in the rapidly expanding area of on-line diagnostics and condition monitoring of all types of industrial, commercial and military equipment. After all, without accurate and reliable information on the equipment being monitored, it is impossible to diagnose the machines current condition or "health" in order to make informed maintenance and safety decisions.

Numerous sensor validation and recovery systems have been developed and tested over the years to separate failed sensor hardware from "real" equipment malfunctions [1-6]. In particular since 1980, when a ground test of the Space Shuttle main engine experienced erroneous combustion chamber pressure measurements that were used in the closed-loop thrust level control algorithm [7]. In this case, the failed pressure sensor led to running the engine in a severely abnormal operating condition, and nearly self-destroyed the engine. Following that incident, sensor validation and recovery research has focused primarily on utilizing sensor redundancy and knowledge-based systems that operate well under steady-state conditions [8-11]. Today, with the increased application of neural networks to solve non-

linear, pattern recognition problems, non-redundant equipment sensor patterns can be "learned" by dedicated neural networks to detect and isolate sensor failures. The additional advantage of utilizing neural networks is reducing the dependency on redundant sensors and steady-state operating conditions.

Algorithms designed to perform non-redundant sensor validation in transient monitoring environments are based on the principal that the various sensors used in a particular application are non-linearly related over a particular speed/power range. For example, if a machine is increasing in speed and power, then the temperatures, pressures, etc. are related to each other in some non-linear sense. Neural networks would appear to be a good tool for solving this type of problem because of their non-linear, pattern recognition and classification qualities. In addition, fuzzy logic was chosen as the best tool for deciding if the speed/power range of the equipment is increasing, decreasing, or in a steady-state condition and whether the sensor confidence level output of the neural networks depicts a good, bad, or marginal sensor condition.

Neural Networks for Non-Linear Pattern Recognition: Neural networks are systems of elemental processing units connected in a way analogous to how neurons are connected in the brain. Like the brain, neural networks exhibit learning and associative memory skills. A neural network is trained to perform a task by showing it examples of an input it will receive, paired with the output it is to deliver. The network learns the associations between these pairs of input examples and corresponding outcomes, and is able not only to reproduce these associations, but also to generalize these relationships for inputs that it has not encountered before. Neural nets are therefore capable of intelligent interpolation and therefore make them particularly well suited for this type of application.

The artificial neural network can be viewed as a collection of elemental processing units massively interconnected among themselves. Some of the processing units, sometimes called nodes, communicate with the outside environment. We distinguish between the different types of processing units with the following nomenclature:

1.) Input Nodes: Receive signal from the environment

2.) Output Nodes: Send signals to the environment

3.) Hidden Nodes: No direct contact with the environment

The processing unit or node is the component within neural networks where the computations are carried out. The input signal come from either the environment or other processing units, and form an input vector containing all the inputs. Figure 1 is an illustration of one processing unit or node.

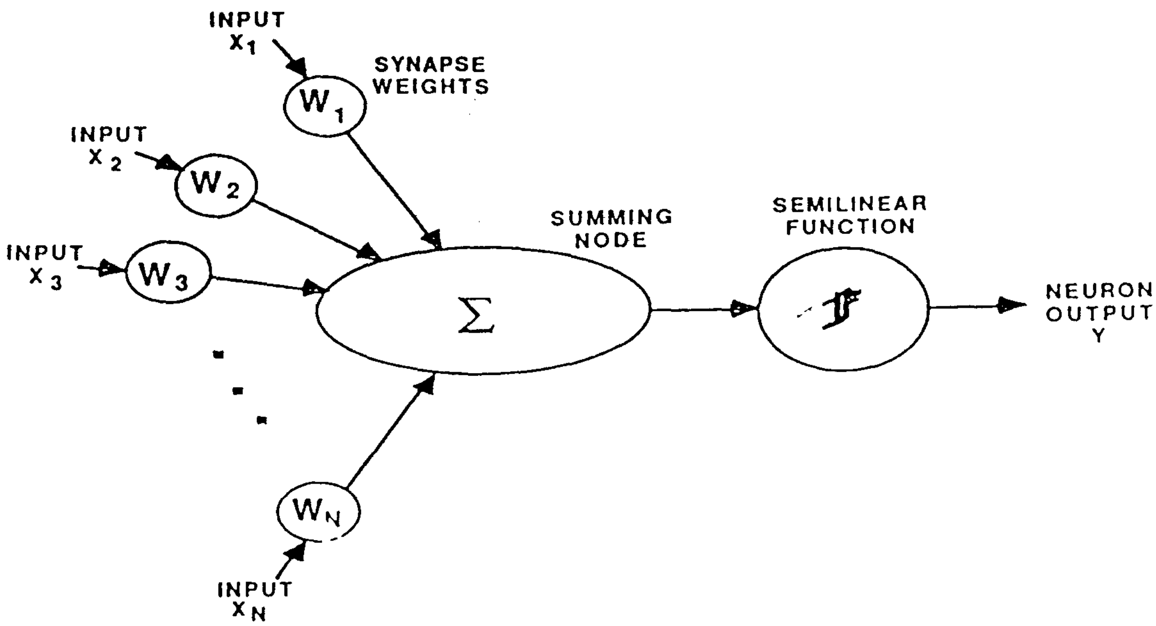

Figure 1 Neural Network Processing Unit

Figure 1 also shows weights corresponding to each input. These weights are used to compute the output value of the processing unit. This computation is performed by taking the product of each input value x_i and its corresponding weight w_i. These products are then summed together and "passed" through a sigmoidal activation function to determine its final output activation level. Other types of activation function can be used, but the sigmoid is the most commonly used function.

When we talk about neural network's abilities to learn cause and effect relationships, we are really discussing the supervised learning procedure. Supervised learning involves the task of teaching the network associated input/output pairs. The network is presented with data that shows what response (output) should be generated by a given stimulus (input). The network then self adjusts its internal parameters in order to represent this underlying relationship between the inputs and outputs. This is the basis for a neural network's ability to generate appropriate outputs for all other inputs of a similar category, even if these inputs have never been previously encountered.

Supervised learning is commonly implemented using a Generalized Delta Rule network architecture with backpropagation of error. During this procedure, the network architecture is specified in terms of the number of input and output nodes, as well as hidden layer nodes. The training set is then used to specify what target outputs should result from an input pattern, and the network automatically learns the set of parameters (weights and thresholds) that will generate this desired output. In this learning procedure, the network learns a single set of network parameters that satisfies <u>all</u> the training input/output pairs. The learning is not perfect, but is optimum on the basis of the least mean square error. In the consulting mode, the network is able to generalize and generate appropriate output patterns for any input pattern

applied to the network. This attribute is the principal advantage to utilizing neural networks in condition monitoring applications.

Fuzzy Logic for Approximate Reasoning: Fuzzy logic is a programming tool that is capable of incorporating imprecise or ambiguous information into algorithmic expressions. However, contrary to its name "fuzzy", the mathematics involved are based on precise and rigorous calculations with respect to fuzzy sets or membership functions. The four basic processes required to develop fuzzy logic systems are fuzzification, rulebase development, inference, and defuzzification. The fuzzification process begins with the development of membership functions which relate linguistic variables like "cool", "hot", and "cold" to particular numerical ranges used in the "fuzzy" calculations. For instance, "cool" might have a membership value of 1.0 (the highest degree of membership) for 60 degrees F, a membership of 0.6 for 50 degrees F, a membership of 0.25 for 40 degrees F, and a membership of 0.0 (no degree of membership) for 30 degrees F.

The rulebase development is typical of any if/then rule set implemented in standard expert systems, except the rules now incorporate the "fuzzy" linguistic variables that have membership functions associated with them. An example of a rule would be; If *temperature* is <u>cool</u>, Then *velocity* is <u>medium</u>. In this rule, *temperature* is the *input variable* and *velocity* is the *output variable*, both or which have membership functions associated with them that include <u>cool</u> and <u>medium</u> respectively.

The strategies for "inferring" conclusions/decisions from cause-and-effect relationships provided by the rulebase and membership functions (knowledge base) are often called fuzzy inference techniques. Some inference techniques include; Product-Sum, Max-Min, and Min-Sum. The first expression of the inference technique name refers to the method for scaling the membership function variables. The second expression refers to the technique for combining the scaled membership function variables. A more complete description of the inference techniques is given in Reference [12]. The final process of calculating a single value from the scaling and combining of the variables described in the membership functions is called defuzzification. Techniques such as Centroid, Max-height, and Max-moment are used to determine the value that best represents the outcome of the fuzzy rule evaluations.

Sensor Validation System Architecture: The sensor validation system architecture involves the integration of the neural networks, fuzzy logic, and miscellaneous arithmetic and logic operations. A block diagram of the basic system architecture is given in Figure 2. The speed/power sensor data is first accepted by two parallel fuzzy logic modules. The first module determines the state of the speed/power condition (i.e increasing, decreasing, or steady-state) and the second verifies the validity of speed/power sensor itself. The output of the speed/power condition module triggers a particular neural network module that was specifically trained to know the sensor relationships for either increasing, decreasing or steady power output. Only one neural network module is triggered at a time, depending on the outcome of the prior fuzzy logic decisions. The sensor confidence values predicted by the

neural networks are trended over time and passed through another fuzzy logic module to interpret the results. These extra steps are used to ensure that false alarms do not occur.

For the gas turbine engine application discussed in this paper, there are four primary performance related sensors that are monitored during turbine operation. These sensors include; fuel flow (Wf), HP compressor delivery pressure (P3), LP compressor delivery temperature, and Jet Pipe temperature (T6). The outputs of the neural networks yield a confidence factor associated with the probability of a failed sensor. A confidence factor near one represents proper sensor operation, while a confidence factor near zero indicates a faulty sensor mode. A fuzzy logic module is used at the output of these neural networks to decide whether the sensor is good, bad, or somewhere in between. For instance, if a hard decision was utilized to alert the crew when a sensor confidence factor reached a level less than 0.80, false alarms would likely occur even though a sensor confidence factor of 0.78 might still indicate a properly working sensor.

The same reasoning applies for using fuzzy logic as a pre-processor for the neural network in terms of determining the speed direction. When examining the difference in speed change based on several different time differences, speed changes approximately near zero would indicate a steady-state operating condition. By implementing fuzzy logic, the "approximately near zero" term can be accounted for in algorithmic expressions.

Neural Network Architecture and Training: Two different neural network architectures were examined for this application. Both networks utilized a multi-layered, feed-forward architecture with five input nodes and four output nodes. The first network contained one hidden layer with 13 nodes and the second used 2 hidden layers with 10 and 5 nodes respectively. Figure 3 is an illustration of the neural network with only one hidden layer and 13 nodes.

Determining the "optimal" number of hidden layers and nodes for each network is a non-trivial task and depends on many factors, some of which include; number of input/output nodes, quantity and accuracy of training data, complexity of problem, and resulting network generalization performance. The "standard" feed-forward architectures used for this problem were picked due to the large quantity of training data available and the resulting network generalization performance required.

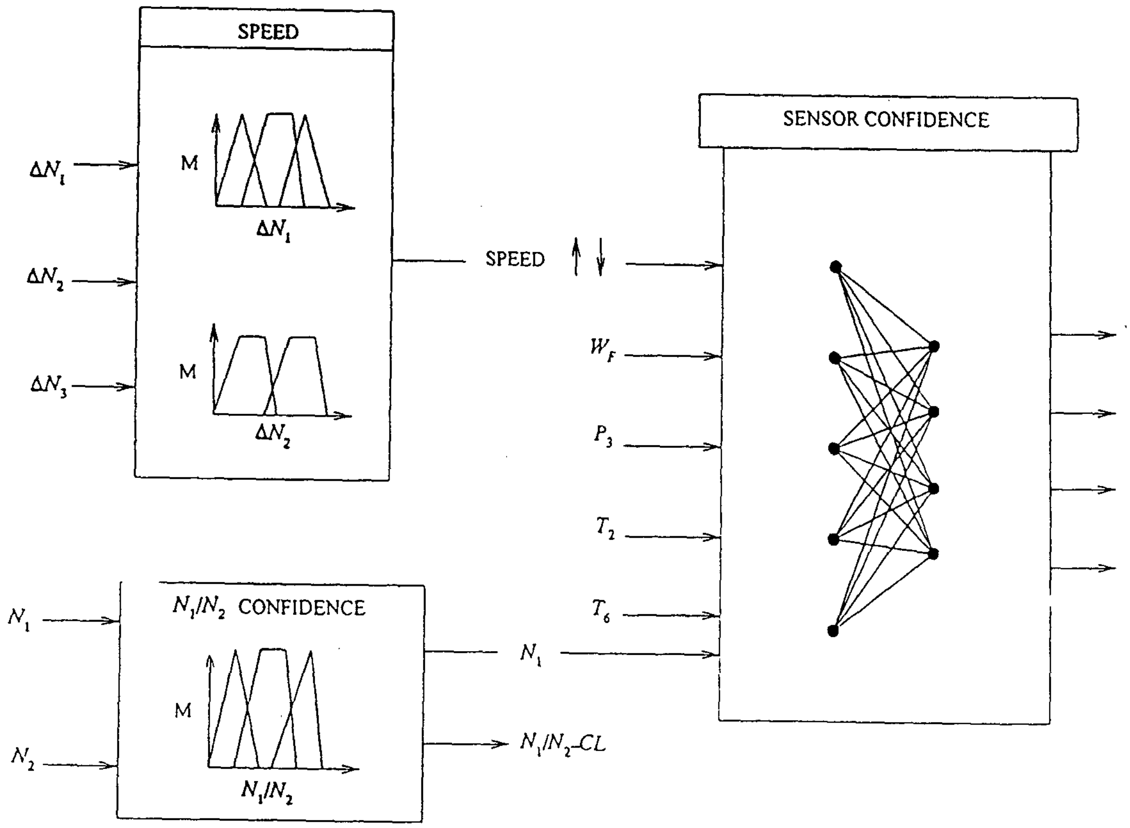

Figure 2 **Sensor Validation System Architecture**

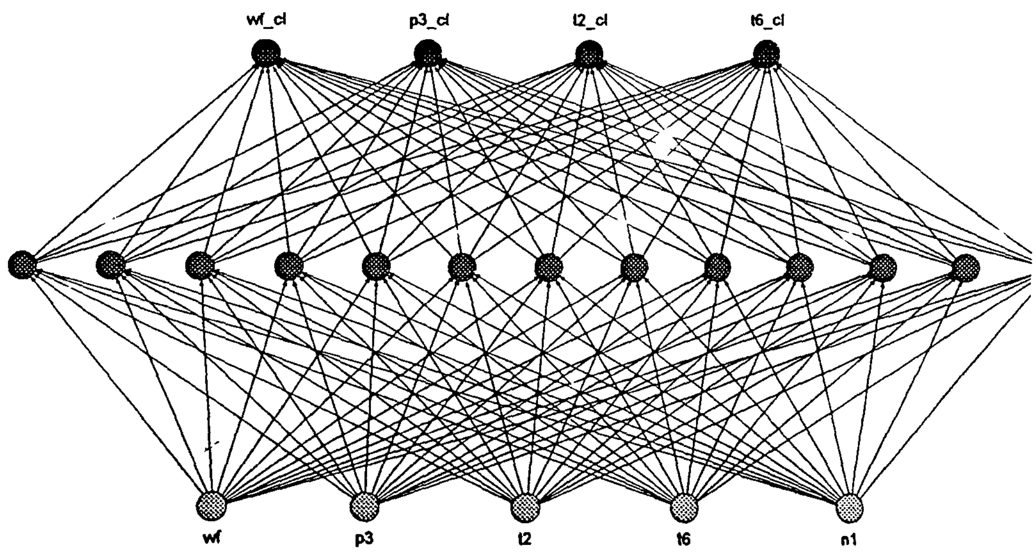

Figure 3 **Neural Network Architecture**

The simulated gas turbine engine data used to train the neural network architectures is given in Figure 4. The data set represents sensed fuel flow, pressure, and temperature readings during a start-up condition. Simultaneously measured data staying within the illustrated confidence limits for each sensor would represent properly operating sensors. Data going outside these limits would indicate a failure mode associated with the particular sensor. For training purposes, any measurement within the confidence limits of each sensor for a particular engine speed would indicate a sensor confidence level of 1.0 (highest confidence). As a sensor measurement moves outside the confidence limits, the neural network output confidence level decreases from 1.0 towards 0.0 indicating the graduating sensor failure mode. Each network architecture was subjected to the same training data set consisting of 300 input/output pairs.

Training the sensor validation networks was accomplished with a supervised learning procedure. Each of the 300 training pairs or patterns used duri , the training process consisted of 5 sensor input signals and it's corresponding set of 4 outputs sensor confidence factors. The input and output training data was normalized to values between 0 and 1. An error-back-propagation algorithm was used to minimize the mean-squared error between the actual network output and the target values set by the training set. Training parameters such as the learning rate, gain of the activation function, and momentum coefficient were adapted during the training session to aid in minimizing the error. A final RMS error associated with all training pairs was reduced to 0.199.

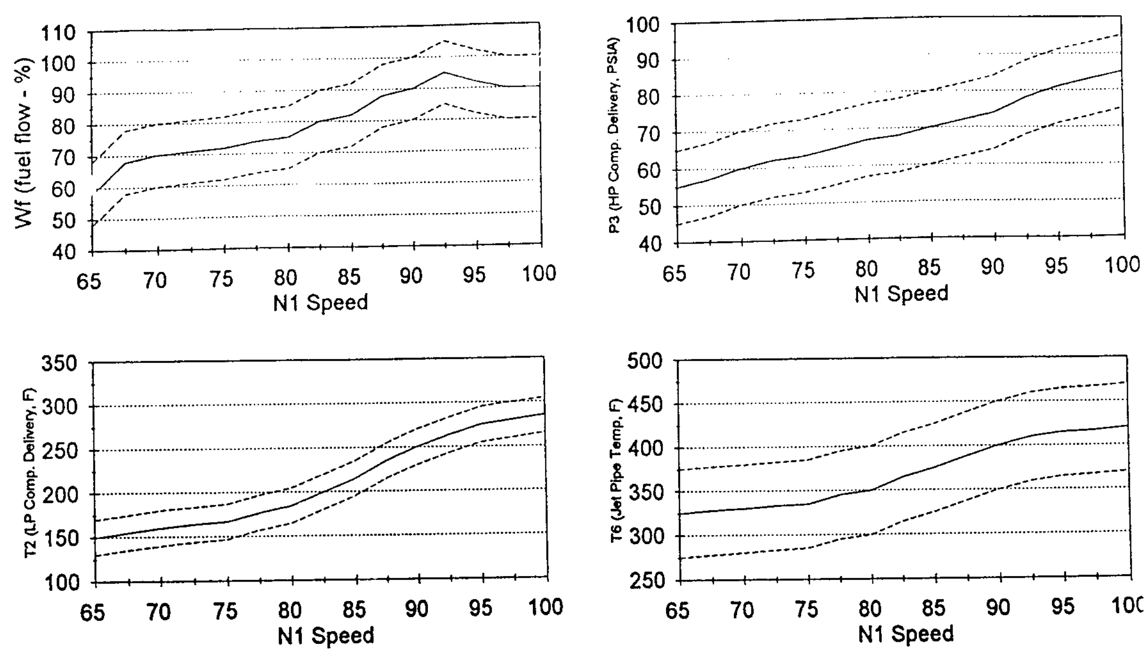

Figure 4 **Gas Turbine Measurement Data**

Neural Network Results: A computer generated data file simulating normal and faulty sensor measurements was developed to test the accuracy of the two neural network architectures. Figure 5 is an illustration of a small section of that file including a range of fuel flow measurement data. The data represented by the + signs are all within the confidence limits of normal operating sensor patterns. In this case, sensor confidence levels predicted by the neural network should all be close to one. The data indicated by X's and O's are outside the confidence limits and therefore indicate worsening sensor operation. The X's are just outside the confidence limits and should predict sensor confidences between zero and one. The O's are significantly outside the sensor confidence band and should predict sensor confidence levels close to zero. The results from this data file are given in Table 1 below. Note, the other sensor measurements including temperatures, pressure, and speed were all within the confidence limits shown in Figure 4.

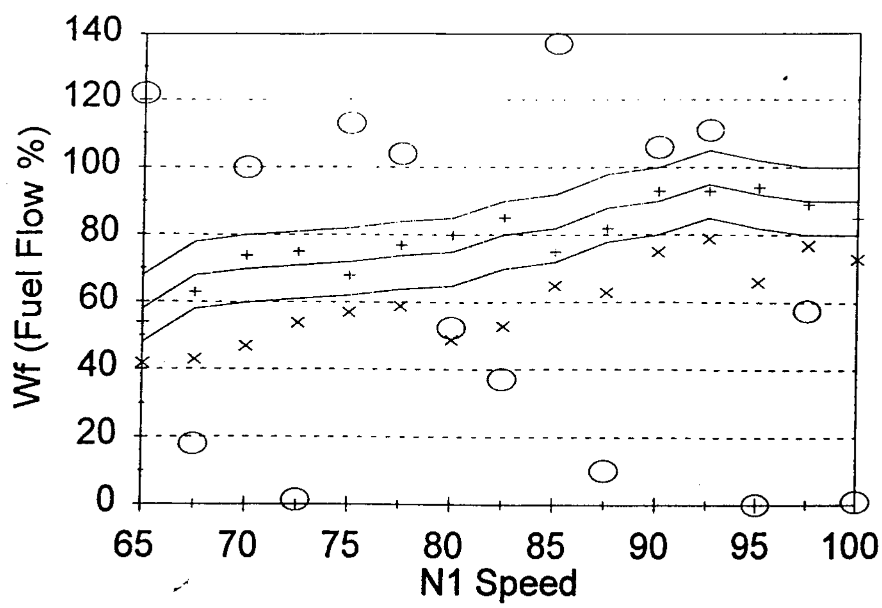

Figure 5 **Simulated Fuel Flow Test Data**

Table 1 Neural Network Results

Case #	"+"	"x"	"o"
1	0.9197	0.2711	0.0755
2	0.9990	0.1674	0.0543
3	0.9990	0.2625	0.0923
4	0.9861	0.7777	0.0500
5	0.9990	0.7700	0.0923
6	0.9990	0.7852	0.1061
7	0.9756	0.2378	0.4670
8	0.9112	0.1263	0.0380
9	0.9734	0.8133	0.1389
10	0.9723	0.3508	0.1736
11	0.8831	0.9244	0.4835
12	0.9259	0.9549	0.4616
13	0.8556	0.6185	0.2459
14	0.9345	0.9057	0.0980
15	0.9057	0.7111	0.1866

Note: The network output confidence levels for the other sensors were all above 0.975 for "+" test cases, above 0.927 for the "x" test cases, and above 0.946 for the "o" test cases.

Several test cases similar to the results described in Table 1 were conducted for the other sensor measurements, all yielding similar results. Although the trained network yielded good results in terms of accuracy and generalization capabilities, overtraining was a concern that was monitored carefully. Initially, 600 training patterns were used to train the network with output error similar to the 300 training pattern case. The resultant trained network had much worse generalizing capabilities than the network trained with only 300 patterns.

The network architecture with two hidden layers was trained and tested with the same data as used for the previous single hidden layer network. In this case, the output RMS error was only reduced to 0.289 and the network generalization capability degraded. The worsened generalization capability can be explained by the additional degrees of freedom that were introduced by the additional nodes (neurons) in the hidden layers. The higher network RMS error is most likely due to finding a "local" minimum associated with the gradient decent BPE algorithm. In theory, the error should have been reduced to at least the level of the previous single hidden layer network.

Conclusions: A sensor validation scheme was presented for checking sensor hardware operation without the need for sensor redundancy and during transient equipment monitoring conditions. Fuzzy logic and neural networks were applied to meet the non-linear and non-exact nature of this problem. Fuzzy logic modules were used to determine the transient operating condition of the machine and to interpret the neural network sensor validation network outputs in terms of normal, faulty or marginal sensor operation. Neural networks were developed and trained for increasing, decreasing, and steady-state operating conditions. These trained networks accepted the non-redundant sensory data and interpreted the non-linear relationships between them in order to recognize when a sensor reading did not match the patterns used in the training process. The output of the networks included the sensor confidence factors which ranged from zero (faulty sensor) to one (normal sensor). An output fuzzy logic module interpreted the sensor confidence values to determine when the equipment operator should be notified of a sensor hardware problem. Trending the sensor confidence factors over time is an additional step used to ensure accurate diagnosis of failed sensor hardware.

Results of the sensor validation scheme when subjected to computer simulated data representing gas turbine engine sensor hardware measurements was encouraging. The neural network never predicted sensor failures when measurements stayed within the trained sensor confidence bands. Also, sensor confidence levels were always predicted to be less than 0.5 if a sensor measurement drifted more than 50% outside the sensor confidence bands.

References

1. Walker, N. D. and Wyatt-Mair, G. F., "Sensor Signal Validation Using Analytical Redundancy for an Aluminum Cold Rolling Mill", **Control Engineering Practice**, Vol. 3, No. 6, June 1995, pp.753-760.

2. McAvoy, T. J., "Sensor Data Analysis Using Autoassociative Neural Nets", **World Congress on Neural Networks**, Vol. 1, 1994, pp.161-166.

3. Holbert, K. E., Heger, A. S., and Alang-Rashid, N. K., "Redundant Sensor Validation by using Fuzzy Logic", **Nuclear Science and Engineering**, Vol. 118, No. 1, Sept. 1994, pp.54-64.

4. Harrison, P. R., and Harrison, P. A., "Validating an Embedded Intelligent Sensor Control System", **IEEE Expert**, Vol. 9, No. 3, June 1994, pp. 49-53.

5. Ahmadi-Echendu, J. E., and Hengjun, Zhu, "Detecting Changes in the Condition of Process Instruments", **IEEE Transactions on Instrumentation and Measurement**, Vol. 43, No. 2, April 1994, pp.355-358.

6. Lee, S. C., "Sensor Value Validation Based on Systematic Exploration of the Sensor Redundancy for Fault Diagnosis", **IEEE Transactions on Systems, Man and Cybernetics**, Vol. 24, No. 4, April 1994, pp. 594-605.

7. Aguilar-Crespo, J. A., de Pablo, E., and Alaman, X., "A Fuzzy Logic Approach for Sensor Validation in Real-Time Expert Systems", **Advanced Methods in Artificial Intelligence**, 1993, pp.330-337.

8. Uhrig, R. E., "Use of Neural Networks in the Analysis of Complex Systems", **Third Workshop on Neural Networks - WNN92**, 1993, pp.16-24.

9. Upadhyaya, B. R., Eryurek, E., and Mathai, G., "Neural Networks for Sensor Validation and Plant Monitoring", **Proceedings of the 1990 International Fast Reactor Safety Meeting**, Vol. 3, 1990, pp. 349-358.

10. Lee, S. C., Ratliff, M., Peterson, C., and Lollar, L. F., "AMPERES: A Real-Time Fault Monitoring and Diagnosis Knowledge-Based System for Space Power Systems", **Applied Artificial Intelligence**, Vol. 3, No. 5, July 1991, pp. 281-308.

11. Boger, Z., "Applications of Neural Network Techniques for Fault Detection", **Nuclear Societies of Israel Transactions**, Vol. 16, 1990, pp. 217-2[illegible].

12. Ray, A., and Luck, R., "An introduction to sensor signal validation in redundant measurement systems", **IEEE Control Systems Magazine**, Vol. 11, No. 2, Feb. 1991, pp. 44-49.