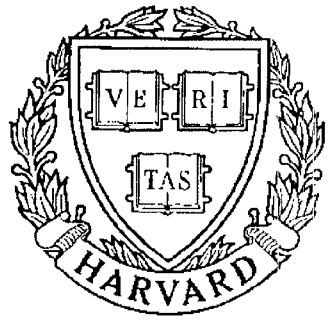
TECHNICAL RESEARCH REPORT

SYSTEMS RESEARCH CENTER



Supported by the National Science Foundation Engineering Research Center Program (NSFD CD 8803012), Industry and the University

Control System Sensor Failure Detection via Networks of Localized Receptive Fields

by S.C. Yao and E. Zafiriou

TR 90-24

1990 Amer. Control Conf.
San Diego, CA

Control System Sensor Failure Detection via Networks of Localized Receptive Fields

Sam Chuan Yao and Evanghelos Zafiriou*

Chemical Engineering Program
and Systems Research Center
University of Maryland
College Park, MD 20742

Abstract

This paper investigates the use of local receptive field networks (LRFN) in detecting sensor failures of a control system in the presence of model-plant mismatch. Simulation results indicate that LRFNs hold significant promise in sensor failure detection. Another issue discussed in this paper is a method to prune redundant nodes. A simple scheme which uses singular value decomposition (SVD) is developed to identify and remove excess nodes. Comparable classification performance is obtained using reduced and standard LRFN.

1 Introduction

Neural networks have been shown to be useful and successful in many areas of engineering applications, especially in the field of information processing. This paper investigates the use of local receptive field networks (LRFN) in detecting sensor failures in a control system and distinguishing them from the effects of model error. In previous work (Naidu *et al.*, 1989), we demonstrated that for a single-input single-output (SISO) example, a standard backpropagation network (BPN) gives a more accurate prediction of supercritical sensor failures than a robust detection algorithm with a finite integral squared error criterion and the Nearest Neighbor method. Despite these promising results, there are several unanswered theoretical problems that need to be addressed before standard BPN can be an effective paradigm for engineering applications. Such problems include the determination of the learning rate, momentum factor, number of neural nodes, and scaling of input and output space. As a result, heuristic trial-and-error approaches tailored to particular applications are often used. An alternative network structure that has been shown to perform equally or better than BPN in a variety of applications is the local receptive field network introduced by Moody and Darken (1988, 1989). An important reason for using LRFN is the simplicity of the network structure which makes it amenable to rigorous mathematical analysis.

In the following sections, a brief review of the important concepts of LRFN is given. Another issue discussed in the paper is the implementation of a simple scheme to remove redundant neural nodes. This scheme utilizes singular value decomposition (SVD) to determine the appropriate number of nodes in LRFN. A rigorous derivation of which nodes to prune is also demonstrated. Finally, a summary of simulation results for both training and test data sets is presented. The results indicate that LRFN can be a promising and useful neural network in detecting sensor failure in the presence of model-plant mismatch. The results also show that reduced LRFN and regular LRFN have about the same classification performance.

*Author to whom corresponce should be addressed. E-mail: zafiriou@cacse.src.umd.edu.

2 Localized Receptive Field Networks

The schematic diagram of the LRFN architecture is shown in Fig. 1. The LRFN has a single internal layer which maps a real valued function $f : R^n \rightarrow R^o$. The overall response function for the LRFN with one output, $o = 1$, is given by:

$$f(\vec{x}_i) = \sum_{j=1}^{m} w_j R_j(\vec{x}_i) \quad i = 1, 2, ..., p \qquad (1)$$

where R_js are radially symmetric functions, e.g., $R_j(\vec{x}_i) = exp(\frac{-\|\vec{x}_i - \vec{x}_{cj}\|^2}{(\sigma_j)^2})$. In (1), the LRFN has m nodes, $f(\vec{x}_i)$ is the predicted output for a given real input vector $\vec{x}_i$, and R_j is the j^{th} receptive field response function. This function is also known as a radial basis function where $\vec{x}_{cj}$ and σ_j are the cluster center and width, respectively. The variable, w_j, is known as the weighting function.

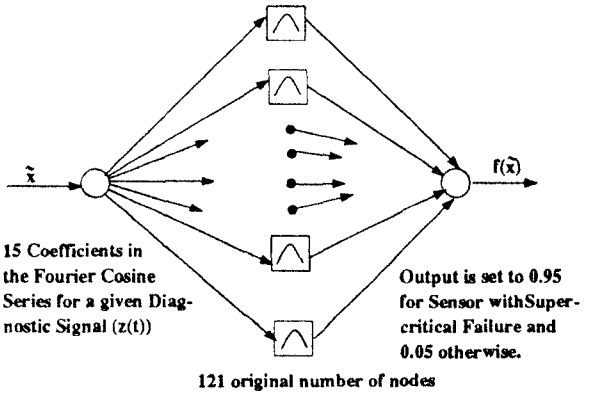


Figure 1. Schematic diagram of LRFN.

The training of LRFN is a hybrid of two learning methods, unsupervised and supervised, and it involves a 3-stage procedure. The first and second stages involve the determination of cluster centers and widths using unsupervised learning. A standard k-means clustering algorithm is used to determine m cluster centers of the LRFN. After the centers are obtained, cluster widths (σ_j) are calculated using the "P nearest neighbors" heuristic. This heuristic basically solves for cluster widths such that the equation $\tilde{E} = 0.5 \sum_{j=1}^{m} [\sum_{k=1}^{P_{nearest}} (\|\vec{x}_{cj} - \vec{x}_{ck}\|)^2/(\sigma_j)^2 - P]^2$ is minimized, where P is known as an overlap parameter and the $\vec{x}_{ck}$s are the P nearest neighbor to $\vec{x}_{cj}$. The purpose of this heuristic is to ensure that the radial basis functions form a smooth interpolation of those regions of the input space they represent. For a more detailed discussion of cluster center and width determination, the reader is referred to Moody and Darken (1988). In the third and final stage, supervised learning is used to determine the weighting functions. Moody and Darken recommended the use of a Quasi-Newton method to iteratively locate weights which would minimize the total sum of squared error between p actual and predicted output values given by:

$$E = 0.5 \sum_{i=1}^{p} [f^*(\vec{x}_i) - f(\vec{x}_i)]^2 \qquad (2)$$

where $f^*(\vec{x}_i)$ is the actual ouput for a given training pattern $\vec{x}_i$.

As a variation from the recommended method for the final stage, we use an approach similar to Renals and Rohwer (1989). This method has the advantage of being computationally more efficient than the former. No iterative minimization of error is required to calculate the weights. In addition, this approach enables us to develop a simple procedure that allows one to reduce network nodes by means of linear algebra operations. Rewriting equation (2) in matrix notation yields:

$$\min_{\vec{w}} E = 0.5(\vec{y} - R\vec{w})^T(\vec{y} - R\vec{w}) \qquad (3)$$

where $\vec{y}$ is the px1 vector of desired outputs $f^*(\vec{x}_i)$, R is the pxm matrix containing all the radial basis functions, and $\vec{w}$ is the mx1 vector of output weights. To minimize this overall error, the gradient of the error at that point must be zero and the Hessian must be positive definite. To satisfy the first condition, the gradient of the total error with respect to the vector weights is set equal to zero, i.e.,

$$\nabla E = -R^T\vec{y} + R^T R\vec{w} = 0 \qquad (4)$$

Note that the coefficient of $\vec{w}$ is the Hessian (H). In order for H to be positive definite ($\vec{w}^T H \vec{w} > 0$ for all $\vec{w} \neq 0$), it is necessary that R be nonsingular. It is worth noting that R also has to be nonsingular for (3) to have a unique solution. The assumption that R is nonsingular is generally valid when $p >> m$. In other words, when the number of training patterns is far greater than the number of nodes, the likelihood of having a singular R becomes small. Rearranging (4), we obtain

$$R^T R \vec{w} = R^T \vec{y} \tag{5}$$

Direct inversion of the Hessian in (5), yields

$$\vec{w} = (R^T R)^{-1} R^T \vec{y} = H^{-1} R^T \vec{y} = R^+ \vec{y} \tag{6}$$

Thus, the determination of the LRFN weights becomes a problem of computing the inverse of H or the pseudo-inverse R^+ of R. To take the inverse of H, we recommend the use of SVD, not only because it is a numerically good technique, but also because information obtained from it is helpful in estimating the appropriate number of nodes. In addition, the right singular vectors of H are useful in identifying nodes to prune.

3 Reduction of Network Nodes

One of the main objectives of this research is the optimal determination of neural nodes. If the number of nodes is small, its ability to generalize the map of the function may not be sufficient. On the other hand, when the number of nodes is too high, computational efficiency of the network is significantly reduced. Furthermore, the network may need a larger database than necessary in order to satisfy the condition that $p >> m$. Our goal is to develop a mathematically rigorous method to determine the necessary nodes. To accomplish this, we have developed a simple scheme by means of linear algebra methods. In particular, we use SVD to determine and remove redundant nodes. Recently, Xue *et al.* (1990), also attempted to address the problem of removing redundant nodes in a BPN by using SVD to approximate the rank of the output covariance matrix of the BPN. A problem with their approach is the analysis of the covariance matrix of the hidden unit outputs when the neurons are nonlinear (sigmoidal) functions. Also, the identification of redundant nodes is not straight forward and a a semi-heuristic approach has to be used.

As pointed out earlier, $H = R^T R$ was assumed to be nonsingular when taking the inverse of H. Although H is full rank, from our experience it is highly ill-conditioned. An ill-conditioned matrix is nearly singular. As a result, (3) has essentially infinite solutions. Suppose that H is singular with $rank[H] = r$ and $r < m$. One can closely approximate and determine the rank of H by checking its singular values. Since H is not full rank, $R^T R \vec{w} = R^T \vec{y}$ is underdetermined. Note that singularity of H implies singularity of R and $rank[R] = r$. This means that the set of linear equations has more unknown variables than the number of independent equations. Thus the solution to the set of linear equations has infinitely many possible solutions, given by

$$\vec{w} = \vec{w}_0 + N_H \vec{z} \tag{7}$$

where $\vec{w}_0$ is a particular mx1 vector solution. A particular solution $\vec{w}_0$ can easily be calculated from an SVD of R. For instance, let the SVD of R be $R = U_1 \Sigma_1 V_1{}^T$, where U_1 and V_1 are pxr and mxr unitary matrices ($U_1^T U_1 = V_1^T V_1 = I$). Σ_1 is a diagonal matrix of dimension r whose elements are the non-zero singular values of R. Then using the properties of U_1, V_1, one can easily verify that the following $\vec{w}_0$ is a solution of (5):

$$\vec{w}_0 = V_1 (\Sigma_1)^{-1} U_1{}^T \vec{y} \tag{8}$$

The variable N_H is an mx$(m-r)$ matrix whose columns form a basis of the null space of the rank deficient H. An orthonormal basis for the null space of H can be formed by the last $(m-r)$ columns of the right singular vectors of H (see, e.g. Horn and Johnson, 1985). The variable $\vec{z}$ is any $(m-r)$x1 vector.

Our next task is to determine values of $\vec{z}$ such that $(m-r)$ elements of $\vec{w}$ will be set to zero, thus removing the corresponding redundant nodes.

Consequently, we need to find any $(m-r)$ rows of N_H that are linearly independent. This task can be accomplished by indexing all rows of N_H, and applying Gaussian elimination (keeping track of the corresponding indices) until the row-echelon form of N_H is obtained. All index numbers which correspond to nonzero rows in the row-echelon form are one of the many set of independent rows of N_H. Using the set of linear independent rows, we solve the following equation:

$$N_H{}^I \vec{z} = -\vec{w}_0^I \tag{9}$$

where the superscript I indicates the selected set of $(m - r)$ rows that are linearly independent. By solving the above equation for $\vec{z}$ and using it in (7), we get a new $\vec{w}$ that has $(m - r)$ zero elements, making the corresponding nodes of LRFN redundant.

4 Classification Performance of LRFN

4.1 Control System Studied

To verify the LRFN's ability to classify decision functions, the SISO control system investigated by Naidu *et al.* (1989) is used as an example. The block diagram for this SISO control system with sensor failure detection scheme is shown in Fig. 2. The plant is a stable, first order linear time invariant system with high model uncertainty. It should be pointed out that the controller is designed using the standard Internal Model Control (IMC) procedure and it is equivalent to a PI feedback controller. Although the control system used in this investigation is simple, it has all the necessary features to test the efficacy of LRFN for monitoring sensor failure in the presence of plant-model mismatch. In fact, the main objective of this work is to test the feasiblity of using LRFN as an alternative algorithm in discerning diagnostic signals caused by plant-model mismatch from sensor failures. The diagnostic signal is given by:

$$z = [I + (P - \tilde{P})Q]^{-1}[\phi + (P - \tilde{P})Qr] \tag{10}$$

where P, $\tilde{P}$, Q, r, ϕ correspond to the plant, model, IMC controller, setpoint change, and sensor failure respectively. For the purpose of simplicity, only 40% uncertainty in the process gain is considered for the plant-model mismatch. It should be noted that the method can be applied to other uncertainty descriptions between the plant and the model.

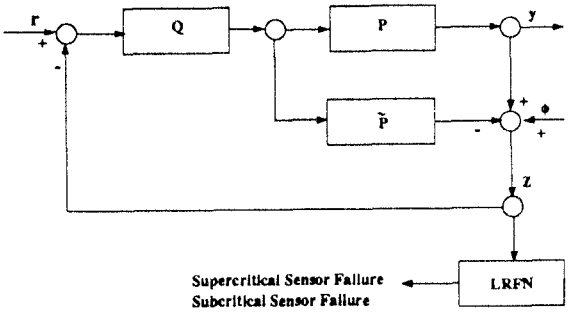


Figure 2. Block diagram of IMC controller with sensor fault detection.

According to the degree of deviation from desired value, the sensor failure detected by LRFN from the diagnostic signals is classified into two classes: supercritical sensor failures and subcritical ones. A supercritical failure occurs when its magnitude of exceeds the predetermined critical value of 0.03. This value is the result of the specification of no more than 5% steady state offset, a reference value of 1 for the plant output, and a ±40% operating range. This type of failure is not tolerated and requires action to eliminate the sensor fault. When the magnitude of the sensor failure is less than 0.03, the signal caused by sensor failure is considered to be subcritical. This type of failure is considered acceptable and allowed to persist in the system operation.

4.2 Training and Results of LRFN

To train the LRFN in classifying supercritical sensor failures from subcritical ones, 7500 diagnostics signals uniformly distributed on the sample space were used. The five parameters

of this space are: a) magnitude of sensor failure, b) time, within the window of observation, when the sensor failure occurs, c) magnitude of setpoint change, d) time when the set point changes, and e) steady state gain of the "uncertain" plant. For more details, see equations (3) to (7) in Naidu *et al.* (1989). The coefficients of the cosine Fourier series of the diagnostic signal serve as input vectors for the LRFN. In this work, only 15 Fourier coefficients are used as inputs instead of 24 Fourier coefficients employed in Naidu *et al.* (1989). We found that the smaller number is sufficient and it also helps reduce the computational load in training the LRFN. In defining the output of LRFN, a value of .95 is set for supercritical sensor failures while a value of .05 is set for subcritical sensor failures. In locating cluster centers using k-mean clustering algorithm, 125 initial clustering centers are arbitrarily used. After the algorithm converged, 4 clusters are found to contain no members. This implies that the 4 clusters are in a region where data are unlikely to be located; thus these 4 cluster centers are removed. In applying the "P nearest neighbor" heuristic, a value of $P = 1$ or $P = 2$ is typically used. In this work, we have chosen $P = 2$ to calculate the necessary cluster widths for the LRFN. After both cluster centers and widths are obtained, weights of the LRFN are calculated by minimizing (2). The classification performance of LRFN on the training set is shown in Table 1. Note that p_{ij} is a measure of classification performance, where i and j represent the actual and predicted fault status of the control system. In particular, p_{11} is a performance measure of correctly predicting supercritical sensor failures, while p_{00} corresponds to correctly predicting subcritical sensor failure. The last two performance measures, p_{01} and p_{10}, represent a false alarm (raising the alarm for subcritical failures) and a miss on supercritical sensor failure, respectively. To check LRFN classification performance with data different from the training set, 5000 randomly generated diagnostic signals within the given sample space are used as test data set. The corresponding simulation results are also shown in Table 1. The total error using LRFN is only around 7%, which is about one third of the error observed with standard BPN in Naidu *et al.* (1989). Also, no significant change in LRFN predictive ability is observed when training data set and test data set are used. To further validate the classification performance of LRFN, Learning Vector Quantization (LVQ), introduced by Kohonen (1987), is also considered in this study. Learning Vector Quantization is a traditional nonparametric classifier, for which LaVigna (1989) showed that when some conditions are satisfied, the classification error can be made arbitrarily small.

Method	p_{11}	p_{01}	p_{00}	p_{10}
Training				
a) QNM	0.563	0.040	0.360	0.037
b) DIH	0.566	0.032	0.368	0.034
c) LVQ	0.546	0.023	0.377	0.054
Test Set				
a) QNM	0.483	0.058	0.442	0.017
b) DIH	0.484	0.056	0.444	0.016
c) RNN	0.483	0.063	0.437	0.017
d) LVQ	0.445	0.027	0.473	0.055

Table 1: Summary of results for different methods. a) Quasi-Newton method (121 nodes) b) Direct Inversion of the Hessian (121 nodes) c) Reduced Node Network(101 nodes) d) Learning Vector Quantization (121 nodes)

Table 1 shows that LRFN has the ability to distinguish between supercritical and subcritical sensor failures. Moreover, it illustrates that no significant classification performance degradation is observed when the original 121 nodes were reduced to 101 nodes, according to the technique in section 3. Finally, comparable results are obtained using Quasi-Newton (QNM) and direct inversion of the Hessian (DIH) for LRFN.

Table 2 shows the performance of the various algorithms on a test space created from the same monitored signal sample space with one exception; the failure parameter is set to zero, i.e., all signals are failure free. It is clear that the percentage of false alarms ($p_{01}/(p_{00} + p_{01})$ falls drastically. In fact, both standard and reduced node LRFN correctly predict over 99% that no sensor failure occured. This indicates

Method	Number of Patterns	p_{00}	p_{01}
Test Set			
a) QNM	5000	0.995	0.005
b) DIH	5000	0.998	0.002
c) RNN	5000	0.991	0.009
d) LVQ	5000	0.999	0.001

Table 2: Summary of results when no sensor failure occurs.

that most of the false alarms of Table 1 are not true false alarms in the sense that the neural network triggered the alarm in cases where a subcritical failure (but still a failure) occured, which it is not required to catch. A final comment is needed on the usefulness of reducing the number of nodes, after a larger network has already been designed. In sensor failure applications, our plan is to train the network off-line with simulations of both failures and failure-free data and then continue the training on-line with presumably fault-free data from a real plant. This strategy was emulated with success for the BPN in Naidu *et al.* (1989). Our objective is, after the off-line training, (for which no significant computing limitations are imposed) to reduce the number of nodes so the computations during the subsequent on-line training are speeded-up.

5 Conclusion

An attractive alternative neural network to standard BPN, the LRFN, has been used to detect control system sensor failures in the presence of plant-model mismatch. Simulation results indicate that LRFNs hold significant promise in sensor failure detection. These results show no significant degradation of classification performance when a random test data set is used instead of the training set.

An important advantage of LRFN over standard BPN is the simplicity of the network structure. It is this simplicity that allowed us to develop a rigorous method to prune redundant nodes. The proposed method uses SVD to identify and remove such nodes. The results show that reduced and standard LRFN have comparable predictive ability, which is similar to that of LVQ.

References

[1] R. Horn and C. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, 1985.

[2] T. Kohonen, *Self-Organization and Associative Memory*, Springer-Verlag, 1987.

[3] A. LaVigna, *Nonparametric Classification using Learning Vector Quantization*, Ph.D. Thesis, University of Maryland, College Park, 1989.

[4] J. Moody and C. Darken, "Learning with Localized Receptive Fields", Proceeding of the 1988 Connectionist Model Summer

[5] J. Moody and C. Darken, "Fast Learning in Networks of Locally-Tuned Processing Units", *Neural Computation*, 1, pp. 281-294, 1989. School, Morgan Kaufmann Publishers, 1988.

[6] M. Morari and E. Zafiriou, *Robust Process Control*, Prentice-Hall, Englewood-Cliffs, NJ, 1989.

[7] S. Naidu, E. Zafiriou and T. Mc Avoy, "Application of Neural Networks on the Detection of Sensor Failure During the Operation of a Control System", pp. 1336-1342, *Proc. Amer. Control Conf.*, Pittsburg, PA, 1989.

[8] S. Renals and R. Rohwer, "Phoneme Classification Experiments Using Radial Basis Functions", 1, pp 461-467, *Int. J. Conf. Neural Networks*, Washington, D.C., 1989.

[9] Q. Xue, Y. Hu, and W. Tompkins, "Analyses of the Hidden Units of Back-Propagation Model by Singular Value Decomposition (SVD)", 1, pp 739-742, *Int. J. Conf. Neural Networks*, Washington, D.C., 1990.