Determining and Improving the Fault Tolerance of Multilayer Perceptrons in a Pattern-Recognition Application

Martin D. Emmerson, *Student Member, IEEE,* and Robert I. Damper, *Senior Member, IEEE*

***Abstract*— Fault tolerance is a frequently cited advantage of artificial neural nets, yet it has rarely been the subject of specific study. In this paper, we investigate empirically the performance under damage conditions of single- and multilayer perceptrons (MLP's), with various numbers of hidden units, in a representative pattern-recognition task. While some degree of graceful degradation was observed, the single-layer perceptron was considerably less fault tolerant (at least, as far as the performance metric employed here indicates) than any of the multilayer perceptrons, including one with fewer adjustable weights. Our initial hypothesis that fault tolerance would be significantly improved for multilayer nets with larger numbers of hidden units proved incorrect. Indeed, there appeared to be a liability to having excess hidden units. A simple technique (called *augmentation*) is described, however, which was succesful in translating excess hidden units into improved fault tolerance. Finally, our results were supported by applying singular value decomposition (SVD) analysis to the MLP's internal representations.**

I. Introduction

Increasingly, it is becoming obvious that the advantages of artificial neural nets, such as self-learning ability and the capability to generalize from one learned pattern to another, will only be fully realized for VLSI implementations [1], [2]. Without the massive parallelism offered by VLSI, adequate speed for many intended practical applications will not be achieved. Fault tolerance is also frequently cited as an important property of neural networks. For instance, Beale and Jackson [3, pp. 89–90] write:

> "Multilayer perceptron networks are intrinsically fault tolerant, since they are distributed parallel processing elements, with each node contributing to the final output response. If a node or its weights are lost or damaged, recall is impaired in quality but the distributed nature of the information means that damage has to be extensive before a network's response degrades badly. The network therefore demonstrates graceful degradation in performance rather than catastrophic failure."

This fault-tolerant property will be crucial to the success of attempts to integrate usefully large nets onto silicon, when problems of yield assume significance.

Manuscript received July 8, 1991; revised December 5, 1992. The work of M.D. Emmerson was supported by a Co-operative Award in Science and Engineering from the U.K. Science and Engineering Research Council, in collaboration with Roke Manor Research Ltd., Romsey, U.K. (previously Plessey Research Ltd.).This work is based on a paper presented at IEEE ICASSP '91, Toronto, Canada, May 1991.

The authors are with the Department of Electronics and Computer Science, University of Southampton, Southampton SO9 5NH, U.K.

IEEE Log Number 9207791.

A small number of researchers has confirmed empirically the existence of graceful degradation in particular applications. In their now classic study of letter-to-sound conversion with the NETtalk multilayer perceptron (MLP), Sejnowski and Rosenberg [4], [5] randomly perturbed the weights of "a highly trained network" and claimed that "this had little effect on the performance of the net." However, fault tolerance was not the central focus of this study, nor did the authors consider how fault tolerance might be improved. Theoretical studies of performance in the presence of damage have also been undertaken by McClelland [6] on the Willshaw associative net and by Stevenson *et al.* [7] on the adaline net, reaching the same general conclusion about the graceful nature of degradation. Generally, however, these nets are unusual in the extent to which they are amenable to mathematical analysis. Given the importance of this aspect of neural network function, it is perhaps surprising that fault tolerance has so rarely been the subject of *specific* theoretical or experimental study. This paper explores the fault tolerance of the practically important MLP in a representative application, and proposes a possible way of minimizing performance degradation under damage conditions.

There is a close relation between fault tolerance and the concept of redundancy. According to Lala [8, p. 1]:

> "redundancy (i.e., additional resources) . . . *[is incorporated]* . . . into a system with the aim of masking the effects of faults."

The most obvious source of redundancy is excess processing units. Here, we investigate how the fault tolerance of the MLP varies with the size of the net's (single) hidden layer and, by presumption, the degree of redundancy in the net. Damage was introduced by cutting connections to/from the hidden units. A further issue is the method used to calculate the variable weights in the network. More specifically, we attempt to determine the extent to which standard back-propagation training [9] produces a fault-tolerant set of weights, while recognizing that this may be a function of the exact problem being tackled.

The particular application studied here is the classification of coins according to their acoustic emissions after striking a hard object. This problem was chosen for two main reasons. First, it is an important pattern-recognition problem of obvious commercial significance, representative of the sort of task for which neural nets are being employed. Second, as we shall

show, it is soluble by a single-layer perceptron (SLP), so allowing us to determine the impact of the hidden layer on fault tolerance.

We assess classification performance under fault conditions as a function of the number of neurons in the (single) hidden layer. Surprisingly, it was found that fault tolerance did not increase with the number of hidden neurons. Thus, back-propagation training does not lead to an internal representation which promotes fault tolerance, at least for this application. The question then arises: how might nets be designed and trained so as to achieve good fault tolerance?

Using the weights from a trained MLP, it is possible to produce an *augmented* network with n times as many hidden units (where n is a positive integer) that produces exactly the same input-to-output mapping. This is simply achieved by replicating the hidden nodes of the *standard* net and their connections as described below. Such a network is redundant by design; accordingly its fault tolerance relative to that of the original (standard) net from which it was derived, and relative to a standard net of the same size, is of some interest.

This paper is structured as follows. We first describe the coin-recognition system which has provided the base data for the study. Experiments are then detailed in which recognition performance is assessed as a function of the degree of damage for networks with differing numbers of hidden units. Results for both standard and augmented nets are described. We next report attempts to relate these findings to an analysis of each net's connection weights, using the technique of singular value decomposition (SVD). Implications of the findings for fault tolerance and its improvement are then discussed, before concluding.

II. Coin-Recognition System

Fig. 1 shows a schematic of the transduction system. A coin is dropped into the slot at the top of the system and falls until it strikes the anvil and bounces. While in the air, it vibrates or "rings"; this acoustic signal is picked up by the microphone above the anvil and passed to an analog-to-digital converter. The coin then lands and rolls down the inclined chute towards the coinbox, passing over a weighbridge with a sensor attached. As it rolls, a noncircular coin (such as the British 50p) will produce a peaky output from the sensor, while a circular coin (such as a 10p) will produce a much smoother output. This is important since some coins (the 10p and 50p for example) are made of very similar amounts of identical metals and would otherwise be hard to distinguish.

A. Microphone Output

When a coin hits the anvil, it vibrates in various modes, some of which are transient and highly random (depending on exactly how the coin struck the anvil). After a short time, the pattern of vibrations settles down into something more characteristic of the specific coin's denomination. Thus, there is a delay of 10 ms after the coin hits the anvil before the microphone output is sampled. Sampling is at 170 kHz, quantized to 8 b, using a window of approximately 3 ms. The

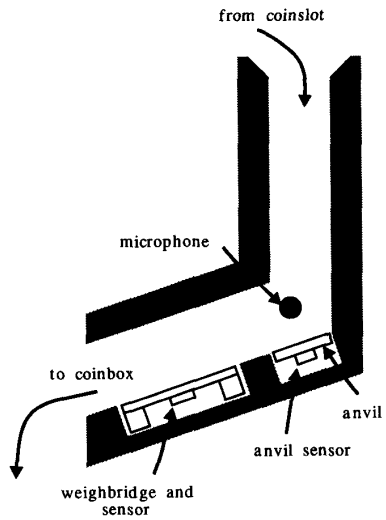


Fig. 1. The coin-classification system. Coins are dropped in at the top, bounce off the anvil, ring while in midair, and then roll down across the weighbridge. The microphone collects the acoustic data which are preprocessed for input to the neural net.

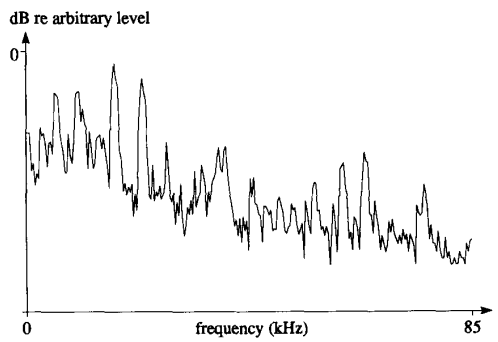


Fig. 2. Typical frequency spectrum of microphone-output signal for 50p piece.

acoustic signal is then frequency transformed using a 256-point fast Fourier transform. Fig. 2 shows a typical spectrum for a 50p piece.

B. Weighbridge Output

The weighbridge sensor is a piezoelectric device, incapable of reliably distinguishing coins by their relative weights alone. It is used to help discriminate noncircular coins, such as 20p or 50p pieces, from circular ones.

The weighbridge output signal is low-pass filtered to reduce noise caused by coins bouncing or rattling on their way down the chute. A peak-finding algorithm is then applied to derive 3 additional parameters to assist classification. These are the overall maximum of the weighbridge response, the sum of the significant peaks in the response curve, and the logarithm of this sum.[1] These 3 parameters are then combined with the 256

[1] Obviously, this latter value is not independent of the sum itself, but its inclusion did improve performance.

spectral values from the frequency analysis to yield a 259-element real-valued vector. This is passed on to the neural network for classification.

C. Neural Net Architectures

All nets were simulated in software (written in C++) on a Sun workstation. Each had 259 input nodes, one for each element of the input pattern-vector, and 6 outputs, one for each denomination of British coin to be recognized (2p, 5p, 10p, 20p, 50p, and £1). Initially, three MLP's were investigated for which the number of hidden neurons (h) were 5, 10, and 20; we refer to these as 259-5-6, 259-10-6, and 259-20-6, respectively.[2] Each net had full interconnection between layers. In addition, one single-layer network (259-6) was studied.

Subsequently, we investigated the performance under damage conditions of MLP's augmented by the addition of redundant nodes after training, as described in Section IV below.

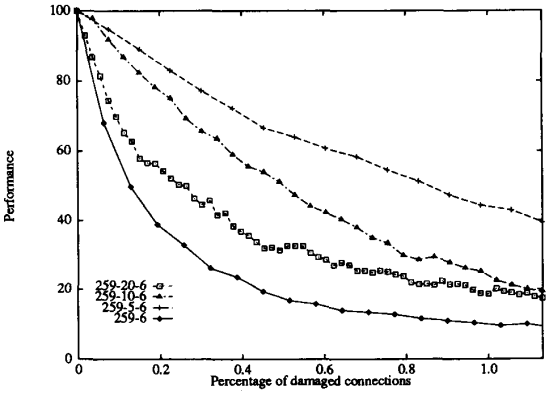


Fig. 3. Performance of the standard 259-6, 259-5-6, 259-10-6, and 259-20-6 networks plotted against percentage of cut connections. This shows the inferior fault tolerance of the SLP relative to the 259-5-6 net, having *fewer* connections. Also, the performance of these MLP's does not improve with increasing redundancy in terms of additional hidden units. In fact, there seems to be a positive advantage in having the smallest number of hidden units.

III. Fault Tolerance of Standard Nets

A. Network Training

To reduce the size of the data set and so speed up training, prototype vectors were prepared for each of the 6 coin classes as follows. First, a mean 259-element vector was calculated from data for 100 different coins of that type. Any vector more than 2 standard deviations from this original mean was discarded and a new vector calculated.[3] This was done because coins may bounce badly and rattle against the sides of the slot instead of ringing clearly.

Each of the 4 standard networks (259-6 and h = 5, 10, and 20) was then trained by back-propagation on the appropriate prototype in two stages. "Coarse" training was performed for 4000 iterations with gain term $\eta = 0.05$ and momentum term $\alpha = 0.8$. Subsequently, "fine" training continued for a further 2000 epochs with $\eta = 0.01$ and α unchanged. The target activity values were 0.9 for the (single) correct output node and 0.1 for the other 5 output nodes.

B. Damaging the Networks

As specific hardware implementations for the MLP have not been discussed here, it is impossible to define precisely those faults which might occur in the network. In order to remain as general as possible, cutting connections between pairs of neurons was chosen as a physically plausible type of damage to study. A cut was simulated by setting the relevant connection weight to zero.

For each network, a number of connections were cut at random with recognition performance assessed after each cut. The nets were compared on the basis of the percentage of damaged connections. Because of the random nature of this damage and the fact that some connections will be more important than others, the process was repeated 500 times for each of the networks and results averaged.

C. Performance Metric

The performance metric chosen for assessing fault tolerance was the percentage of the 500 damage trials in which the net correctly identified all 300 of the test input patterns. (The decision rule was simply to select the pattern class corresponding to the output node with the highest activation.)

D. Results

Test data were obtained by dropping 50 of each type of coin into the system. These test data had not been seen by any of the nets during training. Preliminary tests with the trained, fault-free networks showed that most could correctly classify the 300 input vectors with 100% accuracy. Those that could not were retrained, starting from a different random set of weights, as in subsection A above. This was repeated until all nets could classify the test set perfectly, thus giving a reference level of performance for the assessment of fault tolerance which was the same for all nets.

It would have been desirable to study a number of different versions of each net, trained to the 100% criterion from different initial, random weight values. Unfortunately, however, limits of available computational resources made this impractical.

Fig. 3 shows a composite graph of the performance for the 4 standard networks (259-6, 259-5-6, 259-10-6, and 259-20-6) assessed against the percentage of damaged connections. This plot reveals a number of interesting features.

First, performance degradation is in some sense "graceful." For instance, the 259-5-6 network is capable of suffering

[2] Previous experimentation had shown that, for a network with a single hidden layer, a minimum of 5 hidden neurons appeared necessary in order to train the net to classify the input data without error.

[3] We believe this means of producing prototypes by averaging is admissible for the present situation because there is a large measure of consistency between coins of the same denomination. It would probably not be justifiable in applications like signature verification, where the net would need to capture some knowledge of intraclass variability in its internal representation.

8 damaged connections (0.6%) and still sustaining perfect performance 63.2% of the time.[4]

Second, the 259-6 (SLP) net is clearly inferior in terms of fault tolerance to all the MLP's. Thus, the addition of a hidden layer to the perceptron structure gives an improvement in fault tolerance. Since we have assessed performance in terms of *percentage* of damaged connections, it is clear that this improvement is not merely a result of increased redundancy in terms of net size.

Third, and most importantly, fault tolerance (as assessed here and in this application) does not increase as the number of hidden units increases. This finding is counterintuitive, since excess or redundant hidden nodes might be expected to improve fault tolerance. In fact, the *smallest* MLP ($h = 5$) appears to be the most fault tolerant. Again, this result is unexpected if one assumes that nets with more adjustable weights ought to have greater redundancy, and hence better resistance to damage. Indeed, excess units (and thereby adjustable weights) seem to be a liability—a finding which obviously warrants further investigation.

IV. Fault Tolerance of Augmented Nets

A strong implication of the above findings is that the back-propagation training fails to exploit redundancy (in terms of additional hidden units) in a way which promotes performance under fault conditions. Intuitively, we *should* be able to achieve better fault tolerance with a net having 10 or 20 hidden units than with one having 5. Yet, using back-propagation training, this seems not to be the case. In this section, we consider a technique (called *augmentation*) for translating extra hidden units into improved fault tolerance.

A. Augmenting the Standard Net

Given a trained MLP with h hidden units, it is straightforward to construct a second, *augmented* MLP with $2h$ hidden units, but that performs exactly the same function. This is shown conceptually in Fig. 4 for the simple case of a single hidden unit. This unit is replicated, together with its connections and input weights, in the augmented net. Since each output node now has twice as many inputs as in the standard net, the weights connecting the augmented net's hidden layer to the output layer must be half those in the original net to maintain the same input-output mapping. In principle, this procedure generalizes easily to the production of augmented nets with nh hidden units, where n is any integer greater than one. Our clear expectation is that the augmented network ought to be more fault tolerant than the standard net having the same number of nodes/connections, because of its "redundant-by-design" property.

We denote the augmented net with 10 hidden nodes produced by replication from the standard, trained 259-5-6 net as 5→10. (Note that this is essentially the same notation as that employed by Mozer and Smolensky [10] except that they are concerned with *removing* nodes from the standard net.) Two augmented nets were studied: 5→10, and 5→20.

[4] Of the remaining 36.8% of the 500 damage trials for the 259-5-6 net on which classification errors were made, the majority of these (103 out of 184) gave one classification error only.

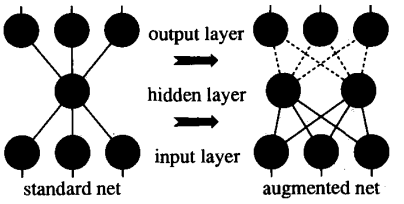


Fig. 4. Doubling the hidden-layer size of a simple MLP with a single hidden unit by replicating the hidden node and its input weights. Since each output node in the augmented net has twice as many inputs as in the standard net, the weights connecting the augmented net's hidden layer to its output layer (shown dotted) must be half those in the original net to maintain the same input–output mapping.

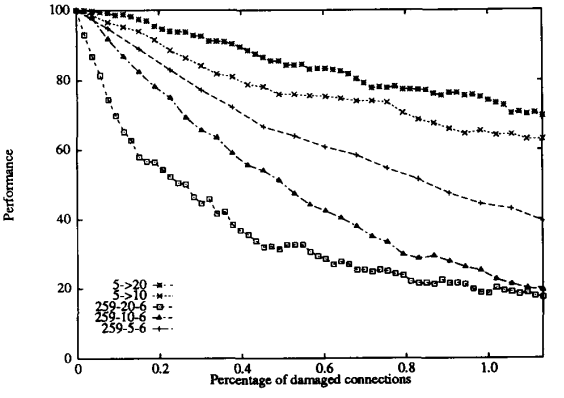


Fig. 5. Performance of standard and augmented MLP's plotted against percentage of cut connections, with the SLP omitted for clarity. The performance increase resulting from augmentation is clearly seen.

B. Results for Augmented Nets

The augmented networks were subjected to the same damage procedures as described for the standard networks. Fig. 5 shows a composite of performance versus percentage of damaged connections for all the MLP's considered so far. This demonstrates a very clear improvement resulting from augmentation to give the 5→10 net. Further, in the case of the 5→20 net, an additional improvement is gained over the 5→10 net.

V. Assessing Redundancy by SVD Analysis

The work so far reported has an obvious relation to that aimed at determining the *relevance* or *contribution* of hidden units to the function of back-propagation networks [10]–[13]. Generally, these techniques attempt both to give insight into the internal operation of networks and to identify excess units which are then removed to produce a *trimmed* or *skeletonized* network. Pruning methodologies offer particular promise of more effective and speedier training, as well as improved generalization performance. By contrast, this paper's concern is the relation between any excess units and fault tolerance.

TABLE I
RANKED VALUES OF THE MAJOR DIAGONAL ELEMENTS OF THE SINGULAR-VALUE MATRIX AS FOUND FROM SINGULAR-VALUE DECOMPOSITION ANALYSIS OF THE NETWORK WEIGHT MATRICES. "REDUNDANT" NODES SHOULD BE REFLECTED IN LINEAR DEPENDENCIES LEADING TO SINGULAR VALUES CLOSE TO ZERO. WHILE THE STANDARD NETS (FIRST, THIRD, AND, FIFTH COLUMNS) DISPLAY NO SUCH LINEAR DEPENDENCIES, THE REDUNDANT HIDDEN NODES OF THE AUGMENTED NETS (SECOND AND FOURTH COLUMNS) ARE CLEARLY IDENTIFIED

259-5-6	5→10	259-10-6	5→20	259-20-6
11.936895	16.881317	11.271613	23.873783	12.228203
11.105055	15.704917	10.862765	22.210104	11.801762
10.699990	15.132061	10.611346	21.399969	11.369482
9.563616	13.524993	10.453448	19.127228	11.336587
8.857811	12.526835	10.080420	17.715624	10.705888
	0.000004	9.534395	0.000005	10.378176
	0.000004	9.278065	0.000005	10.212004
	0.000003	8.914895	0.000003	10.111387
	0.000002	8.525339	0.000002	9.815872
	0.000001	7.648053	0.000001	9.611293
			0.000000	9.296209
			0.000000	8.999689
			0.000000	8.885826
			0.000000	8.707872
			0.000000	8.583886
			0.000000	8.201498
			0.000000	8.081456
			0.000000	7.854286
			0.000000	7.626204
			0.000000	7.105098

For instance, Xue *et al.* [12] have employed the singular value decomposition (SVD) method of Klema and Laub [14] to analyze the hidden nodes of an MLP used for a nonlinear mapping task. The SVD analysis was applied to either the weight matrix or the output covariance matrix of the hidden units to check their corresponding ranks, allowing the detection of linear dependencies among the internal representations stored by the net. These authors reported that matrix rank was closely related to the required number of decision regions in pattern space.

In an attempt to understand our findings, we applied SVD analysis to the weight matrices of our MLP's.[5] The analysis was performed using NAG routine F02WCF [15] on a DEC VAX/780 computer. This delivered the ranked values of the major diagonal elements of the singular-value matrix; linear dependencies result in values close to zero. For this coin-classification application, however, no such linear dependencies were found for any of the standard MLP's, as can easily be seen from the first, third, and fifth columns of Table I. Although $h = 5$ hidden nodes are sufficient for solving this problem, networks with h equal to 10 or 20 have no "redundant" hidden nodes according to the SVD analysis. This result is entirely consistent with the notion that, in this coin-recognition application at least, the error back-propagation training distributes the internal representation relatively evenly across the available hidden units. As such, it does not recognize the existence of excess hidden nodes.

Table I shows the singular values found for all nets—standard and augmented. As expected, the excess nodes of the augmented nets, which are the basis of their enhanced fault tolerance, are clearly identified by the SVD analysis as redundant.

[5] In a subsequent personal communication, Yu Hen Hu (University of Wisconsin-Madison) has told us that linear dependencies are more likely to be found in the output covariance matrix than in the weight matrix.

VI. DISCUSSION AND CONCLUSIONS

We have investigated the fault tolerance of a variety of perceptron networks trained by back-propagation to classify coins from their acoustic emissions when striking a hard surface. The single-layer network had the lowest fault tolerance of all the networks studied; the addition of a hidden layer has a significant impact on performance under damage conditions which cannot be explained simply in terms of increased network size. However, error back-propagation training of the *whole* net did not make best use of any redundancy in the network in terms of promoting fault tolerance, no doubt because the algorithm is searching for a minimum in an error-cost function under fault-free conditions. Surprisingly in view of our initial assumption that excess or redundant hidden nodes would naturally produce a more damage-resistant net, increasing the size of the hidden layer did not improve performance under fault conditions at all. On the contrary, the best performance was achieved from the MLP with the smallest number of hidden units ($h = 5$). We were, however, successful in increasing appreciably the fault tolerance of MLP's with h hidden units in this coin-classification application by training a reduced version of the net with $h/2$ or $h/4$ nodes and deploying the trained weights appropriately in an augmented net.

A number of techniques has been suggested for assessing the contribution made by particular hidden nodes to network function. We have used one of these, singular-value decomposition (SVD), to gain insight into these results. Increasing hidden-layer size failed to improve fault tolerance of the standard nets, implying that the back-propagation training distributed the internal representation relatively evenly across the available hidden nodes. In particular, there was no suggestion that each hidden unit was coding one of a set of discrete "features." If this were the case, then a net with more hidden units than required to represent the essential features of the input data would almost certainly be more fault tolerant than its smaller counterparts.

This is at variance with the finding of Mozer and Smolensky [10] who state that "in our simulations, the network does not seem to be distributing the solution across all hidden units." However, as they themselves point out, they have studied small nets designed to solve theoretically interesting but artificial (in the sense of having exact solutions) problems. Thus, the discrepancy may well result from differences in the nature of the problems studied. To resolve this, we intend in the near future to apply Mozer and Smolensky's relevance analysis to our particular networks.

Clearly, further study is needed to see if the behavior reported here is typical for other applications and, indeed, other ways of assessing network performance. It is always possible that the present results are particular to the data being studied rather than informing us of general properties of neural networks. However, the coin-classification problem was deliberately chosen as a practical, representative example of a recognition task. Accordingly, we feel that the behavior observed is unlikely to be unique to just this problem.

The issue of fault tolerance is of particular importance for the implementation of neural networks using VLSI techniques,

but has not been seriously studied in the case of the practically-important back-propagation networks. We have shown how, for a given network size (and, therefore, with no increase in required silicon area) fault tolerance might be increased in one typical and commercially significant application. If our results are shown to be generally applicable to other problem domains, and even to nets other than MLP's, then this added fault tolerance could lead to higher yields of working devices from the silicon-fabrication process, since any defects introduced are less likely to lead to the failure of the whole network.

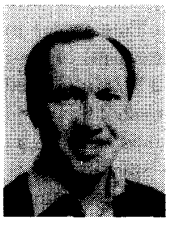
ACKNOWLEDGMENT

The authors thank Dr. C. Upstill of Roke Manor Research and Prof. A. Hey of Southampton University for their support and encouragement during the course of this work.

REFERENCES

[1] C. Mead, *Analog VLSI and Neural Systems.* Wokingham, U.K.: Addison-Wesley, 1989.
[2] J. G. Delgado-Frias and W. R. Moore, Eds., *VLSI for Artificial Intelligence and Neural Networks.* New York: Plenum, 1991.
[3] R. Beale and T. Jackson, *Neural Computing: An Introduction.* Bristol, U.K.: Adam Hilger, 1990.
[4] T. J. Sejnowski and C. R. Rosenberg, "NETtalk: A parallel network that learns to read aloud," Tech. Rep. JHU/EECS-86/01, Dep. Elec. Eng. Comput. Sci., Johns Hopkins University, Baltimore, MD, 1986.
[5] T. J. Sejnowski and C. R. Rosenberg, "Parallel networks that learn to pronounce English text," *Complex Systems*, vol. 1, pp. 145–168, 1987.
[6] J. L. McClelland, "Resource requirements of standard and programmable nets," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition,* vol. 1: *Foundations*, D. E. Rumelhart and J.L. McClelland, Eds. Cambridge, MA: Bradford Books/M.I.T., 1986, pp. 460–487.
[7] M. Stevenson, R. Winter, and B. Widrow, "Sensitivity of feedforward neural networks to weight errors," *IEEE Trans. Neural Networks*, vol. 1, pp. 25–36, 1990.
[8] P. K. Lala, *Fault Tolerant and Fault Testable Hardware Design.* Englewood Cliffs, NJ: Prentice-Hall, 1985.
[9] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition,* vol. 1, *Foundations*, D. E. Rumelhart and J. L. McClelland, Eds. Cambridge, MA: Bradford Books/M.I.T., 1986, pp. 318–362.
[10] M. C. Mozer and P. Smolensky, "Using relevance to reduce network size automatically," *Connection Science*, vol. 1, pp. 3–16, 1989.
[11] D. Sanger, "Contribution analysis: A technique for assigning responsibilities to hidden units in connectionist networks," *Connection Science*, vol. 1, pp. 115–138, 1989.
[12] Q. Xue, Y. H. Hu, and P. Milenkovic, "Analyses of the hidden units of the multilayer perceptron and its application in acoustic-to-articulatory mapping," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP '90)*, Albuquerque, NM, vol. 2, 1990, pp. 869–872.
[13] Y. H. Hu, Q. Xue and W. J. Tompkins, "Structural simplification of a feedforward, multilayer perceptron artificial neural network," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP '91)*, Toronto, Canada, vol. 2, 1991, pp. 1061–1064.
[14] V. Klema and A. Laub, "The singular value decomposition: Its computation and some applications," *IEEE Trans. Automat. Contr.*, vol. 25, pp. 164–176, 1980.
[15] Numerical Algorithm Group, *NAG Fortran Library Manual, Mark 12*, Numerical Algorithm Group, Oxford, U.K., 1987.

Martin D. Emmerson (S'91) graduated with a degree in microelectronics and microprocessor applications from Newcastle University, England, in 1985, and received the Ph.D. degree from the University of Southampton, U.K., in 1992. His area of research interest is in fault tolerance of neural networks.



Robert I. Damper (M'87–SM'89) received the M.Sc. degree in biophysics in 1973 and the Ph.D. degree in electrical engineering in 1979, both from the University of London. He also holds the Diploma degree from Imperial College, London, in electrical engineering.

He was appointed Lecturer in electronics at the University of Southampton in 1980, and Senior Lecturer in electronics and computer science in 1989. His research and teaching interests include signal processing, speech recognition and synthesis, and neural computing. He has published approximately 90 research articles and reports.

Dr. Damper is currently Chairman of the Signal Processing Chapter of the United Kingdom and Republic of Ireland Section of the IEEE. In addition, he is a member of the Institute of Physics, the Institution of Electrical Engineers, the Acoustical Society of America, and the European Speech Communication Association.