UNIVERSITY OF ALBERTA

Training Redundant Artificial Neural Networks: Imposing Biology on Technology

BY

David A. Medler ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Science.

DEPARTMENT OF PSYCHOLOGY

Edmonton, Alberta
Spring 1994

ISBN 0-612-11295-0

UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR: David A. Medler

TITLE OF THESIS: *Training Redundant Artificial Neural Networks: Imposing Biology on Technology*

DEGREE: Master of Science

YEAR THIS DEGREE GRANTED: 1994

[illegible]

David A. Medler

9736 - 84 Avenue
Edmonton, Alberta
CANADA T6E 2E9

April 14, 1994

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled *Training Redundant Artificial Neural Networks: Imposing Biology on Tecnology* submitted by David A. Medler in partial fulfillment of the requirements for the degree of Master of Science.

[illegible]

Michael R. W. Dawson

[illegible]

Don Heth

[illegible]

Terry Nearey

March 25, 1994

Abstr[illegible]

One biological principle that is often over[illegible] in the [illegible] of artificial neural networks (ANNs) is redundancy: Redundancy is [illegible] p[illegible]esses within the brain. This paper examines the effects of redunda[illegible] AN[illegible]s when given either a pattern classification task or a function appr[illegible]mati[illegible] [illegible] different pattern classification tasks were used: parity and encoder. [illegible] [illegible]n appr[illegible]oximation task simulated a robotic arm trained to reach towards an obje[illegible] [illegible]nsional space. Initial results indicate that there is an optimal level of redundancy [illegible] of probability of convergence, convergence speed, and convergence efficiency. When this level of redundancy is used, redundant ANNs learned the pattern classification problem much faster, and converged on a solution 100% of the time whereas standard ANNs sometimes failed to learn the problem. Furthermore, when overall network error is considered, redundant ANNs were significantly more accurate than standard ANNs at performing the function approximation task. These results are discussed in terms of the relevance of redundancy to the performance of ANNs in general, and the relevance of redundancy in biological systems in particular.

Table of Contents

List of Tables

List of Figures

Training Redundant Artificial Neural Networks: Imposing Biology on Technology

Connectionist networks first appeared in psychological research with James' 1890 distributed memory model. Since then, other models based on connectionist theory have prospered over the years, most notably Hebb's (1949) theory of brain functioning, Selfridge's (1959) Pandemonium, and McClelland's (1981) Sharks and Jets network; It has not been until recently, however, that connectionist theories have had the opportunity to be tested. With the advent of the modern computer, Artificial Neural Networks (ANNs) have become an alternative modelling tool of brain function.

Recently there has been a considerable amount of debate generated over the relevance of connectionist networks to cognitive science and cognitive neuroscience (e.g. Lewandowsky, 1993; McCloskey, 1991; Seidenberg, 1993). Part of the debate centers on the fact that many ANN design decisions are based on engineering principles and not on biological principles. Consequently, there is often a trade-off between theoretical and technological advances. To be effective cognitive models, however, ANNs should draw on the characteristics of the brain, even though such design decisions may be counter-intuitive from an engineering viewpoint (Dawson & Shamanski, 1993; Dawson, Shamanski, & Medler, 1993). One biological characteristic that has often been overlooked in the design of ANNs is redundancy: Redundancy is the replication of processes within the brain.

The question of redundancy in biological systems has been debated since the nineteenth century when it was proposed that recovery of behavioural impairment resulting from brain injury was facilitated by the replication of processes within the brain. Initial theories held that the two hemispheres of the brain duplicated each other, and this was the reason for recovery of function following unilateral brain lesions (Gall & Spurzheim, 1810-1819; cited in Almli & Finger, 1992). We now know that the two hemispheres of the brain perform vastly different functions (e.g. Graham, 1990; Kandel, Schwartz, & Jessell, 1991; Nicholls, Martin, & Wallace, 1992) and, therefore, that initial theories of redundancy are incorrect. Today, instead of lateralization, redundancy in the brain is viewed at two different levels: The brain's lower functions are normally replicated within a concentrated area of the brain while the higher functions are represented throughout many areas of the brain (Kandel, Schwartz, & Jessell 1991). Therefore, although alternative explanations of functional recovery exist (e.g. alternate strategies, vicarious functioning, diaschisis;for a review of theories of recovery following brain trauma see Almli & Finger, 1992, and Marshall, 1984), redundancy is still held as a viable theory of functional recovery.

Further neurophysiological evidence for redundancy comes from studies of patients with hydrocephalus: A review of 279 patients aged 5 to 25 years who suffered hydrocephalus onset within the first year of life showed normal psychological functioning even though some patients had less than half the normal brain tissue mass (Berker, Lorber, & Smith, 1983; cited in Smith, 1984). In fact, Kolb and Whishaw (1990) state that as long as the cortex maintains its integrity and connections, intelligence can remain unimpaired even though the expansion of the ventricles may leave the cortex less than a centimeter thick. Unfortunately, however, Smith (1984) reports that the onset of age-related mental deterioration in such patients is more rapid and debilitating than in the normal population. These results suggest that the normal brain is at least twice as large as it needs to be for immediate survival, and that the extra baggage of the normal brain only replicates functions it already possesses. Furthermore, without these redundant processes, deterioration is rapid and terminal. This notion is echoed by Glassman's (1987) calculations of the brain's safety factor using reliability theory which indicate that the most conservative estimate of brain size is at least twice the minimum size required for short-term survival.

Recent analyses of animal physiology have also contributed neurophysiological evidence for redundancy in the brain. Kovac, Davis, Matera, and Croll (1983) extensively studied the nervous system of *Pleurobranchaea californica*, and found several physiological systems that produced essentially the same behaviour; however, when combined, these systems greatly enhanced the precision of simple and complex movements. Kovac et al. speculated that each of these systems has an internal specialization based on both the output connections from the command neurons to other elements of the command system and the organization of synaptic inputs to different elements of the command system. Consequently, redundancy within the command systems may bestow an increased input/output gradability yielding greater accuracy in movement.

In a slightly different vein, Strehler and Lestienne (1986) analyzed the intracellular recordings of 55 different single complex cells in area 18 of the visual cortex of a curarized rhesus monkey presented with lines of varying intensities, lengths, widths, orientations, and rate of travel. It was calculated that the probability of finding the number of redundant doublings of coded information in the regularity of triplets of impulses triggered by specific stimuli by chance was less than 1 in 2.5 billion. Therefore, Strehler and Lestienne concluded that information about a stimulus is represented redundantly within the visual cortex.

Similarly, Swindale (1986) noted that orientation selectivity in the visual cortex is produced by more than one mechanism, and in more than one location. Citing evidence from several experiments, Swindale explains that inactivation of the A layers of the lateral geniculate nucleus removes all responding from the middle laye.s of the visual cortex, yet leaves orientation selectivity in the upper cortical layers intact. Conversely, inactivating the upper layers of the visual cortex via cooling leaves the orientation selectivity of the middle layers unimpaired. These results contradict the previously held thought that orientation selectivity in the upper layers was simply a passive reflection of the responses of the middle layers. Instead, these results suggest that there are redundant systems for orientation selectivity. The above presented neurophysiological evidence for redundancy in biological systems is complemented by a sizable theoretical literature on the relevance of redundancy.

Most theoretical work on the relevance of biological redundancy has centered on the factors surrounding the evolution of redundancy. One common assumption, as described earlier, is that redundancy allows for recovery of function following brain trauma; however, some theorists and neurophysiologists use this assumption as an argument against biological redundancy. The argument follows the line that since brain damage is a rarely survived event, it is unlikely to exert any natural selection pressure for neural spare capacity in anticipation of brain damage (c.f. Glassman, 1987). This argument, however, assumes that recovery of function is the main reason for redundancy, as opposed to being a side effect of redundancy. If we assume for a moment that recovery of function is just a convenient side effect of redundancy, then we can consider alternate evolutionary theories.

Calvin (1983) considered the problem early hominids must have faced when trying to knock prey down via a thrown object: The timing precision required to strike a target increases eight-fold with a mere doubling of throwing distance. Consequently, the precision of a single timing neuron becomes too crude to allow effective strikes at any significant distance. In order to alleviate this problem, the brain may have evolved redundant timing neurons to increase the timing accuracy above the known accuracy of any single neuron (Calvin, 1983). This increase in the number of timing neurons to compensate for an otherwise deficient system is related to Swindale's (1986) hypothesis: It is easier to evolve several crude mechanisms that work in parallel to perform a function than one especially effective neural mechanism. Therefore, redundancy may have evolved not because brain damage was anticipated, but because it was easier to replicate, and thus improve, what was already present than to develop a single system beyond reproach.

A slightly different theoretical approach to redundancy comes from Leon (1992) and Jacobson (1976). Leon reviewed the literature on filial learning in both animal and human infants, and proposed that redundant structures within the brain allow the neonate to learn about its environment despite the degraded stimuli that it often encounters. Furthermore, Leon noted that the neonate brain is far less developed than the adult brain, despite the fact that the majority of survival learning must occur within the first few months of life; Leon suggests that redundant systems exist to assure learning even with a degraded nervous system. On the other hand, Jacobson (1976) considered the connections between neurons involved in a memory trace based on Hebb's model of the cortex, and defined redundancy as "to mean the condition that pairs of cells joined along one effective pathway are joined again along another" (p. 150). Using mathematical calculations and assuming initial random connections between neurons, Jacobson showed that redundancy is an inevitable consequence of the connections within the cortex.

Redundancy is a viable biological property, but can it be effectively implemented in ANNs? Redundancy has been mostly ignored in the design of ANNs. Recently, though, there has been a flurry of connectionist research on using multiple nets to solve problems. For example, Baxt's (1992) medical diagnosis network is based on two networks working in parallel: one network is trained to classify positive examples of myocardial infarction, and the other network is trained to classify negative examples. By combining their outputs, Baxt has produced a network that has a hit rate of 97.50% and a false alarm rate of 1.63%. Using similar principles, Tabary and Salaün (1992) trained a neural network to keep the upper bar of a simulated robotic bicycle horizontal while it moved over uneven terrain. To accomplish this, they trained a "static" network to control the angles of the bicycle's forks and a "velocity" network to control the speed of the bicycle. The combined networks allow the bike to successfully adapt itself to the terrain as it moves across it. Both Baxt's (1992) network and Tabary and Salaün's (1992) network are not truly redundant as defined earlier, but are more akin to different aspects of the same system working together, much like episodic and semantic memory systems (see Tulving, 1972). Nevertheless, their multiple nets suggest that smaller networks can be successfully combined to solve a larger problem.

Another form of computational redundancy widely studied today centers around committee machines. Committee machines are based on the principle of using several computers (or networks) at once to solve the same problem. The training algorithm for such machines is rather unique (see Schapire, 1990): Briefly, the first machine is trained on one pattern set, and then subsequent machines are trained on new pattern sets composed of equal amounts of correctly and incorrectly classified patterns that have been passed through previous machines. Once trained, however, there is little agreement as to the best way of combining the outputs of the different committee machines. Several alternatives have been suggested, from a simple "winner-take-all" or "voting" strategy, to summing the outputs, to calculating the mean output, to implementing a separate network to choose which machine's output is the most appropriate. Regardless of the combining strategy used, the committee machines invariably perform better than single networks alone.

The above research examples have centered on improving the performance of ANNs from an engineering perspective solely. For example, it is not clear that any of the output strategies listed above, or even the training algorithms used for committee machines, are biologically plausible. Furthermore, Baxt's (1992) network and Tabary and Salaün's (1992) network necessarily have no basis in biological networks.

Constraints borrowed from biological networks, however, may have positive effects on the performance of ANNs as illustrated by Izui and Pentland's (1990) research on redundant networks. Using biological redundancy as a model, they mathematically analyzed the functional effects of one of the simplest forms of redundancy-- neuronal duplication. Their mathematical calculations predict that redundant networks are more accurate, faster, and more stable than standard networks. These predictions were confirmed by both a feedforward neural network trained on the XOR problem, and a feedback neural network trained on the travelling salesman problem. From these results, Izui and Pentland claim that the "highly redundant nature of biological systems is *computationally* important and not merely a side-effect of limited neuronal transmission speed and lifetime" (p. 237). Although Izui and Pentland's research has laid the mathematical foundations of network redundancy, their practical work requires expansion before redundancy is accepted as a useful addition in ANN design. For example, larger problem sets should be considered as well as the applicability of redundancy to different artificial neural network architectures.

The purpose of this current research is threefold: (1) To experimentally determine if there is an optimal level of redundancy in terms of network performance versus network training; (2) To study the effects of redundancy on ANN learning and performance on "toy" pattern classification tasks; and (3) To study the effects of redundancy on a function approximation task. The first experiment will monitor the performance of redundant ANNs trained on several sizes of a difficult pattern classification task (i.e parity) while varying the levels of redundancy from two to eight. The second experiment will compare the performance of standard networks and redundant networks-- with levels of redundancy held constant-- trained on two different types of pattern classification (PC) tasks (i.e. parity and encoder). The third experiment will compare the performance of the two different network architectures trained on a function approximation (FA) task: The task will be based on the inverse kinematics function required to train a simulated robotic arm to reach towards a point in two-dimensional space. It is hypothesized that the redundant networks will perform better than the standard networks in terms of problem solving ability for the PC problems, and in terms of overall network accuracy for the FA problem.

Experiment 1: Levels of Redundancy

There is no clear definition of exactly what is meant by redundancy. Jacobson (1976) defines redundancy as being the case that pairs of cells are connected through more than one pathway, whereas Izui and Pentland (1990) describe redundancy as the simple duplication of neurons. Many more authors (e.g. Calvin, 1983; Swindale, 1986; Glassman, 1987) are inclined to say that redundancy is the duplication of complete subsystems within the brain. The current experiments will adopt this last definition of redundancy. Redundancy will be created by duplicating a standard network architecture several times, and then combining the weighted output of these subnetworks to produce an overall response. The weighted output response of the redundant ANNs serves two purposes. First, weighting the response is more akin to biological learning as opposed to the more engineering inspired combination techniques such as taking the mean or median response. Second, although no direct connections exist between each of the subnetworks, the procedure of weighting the outputs may allow one subnetwork to influence the learning of the other subnetworks. In other words, it may be possible that each of the subnetworks may become specialized in solving a certain aspect of the problem posed (see Kovac et al., 1983). Therefore, the redundant networks may be viewed as a collection of standard networks working in parallel to solve a particular problem.

Using this definition of redundancy, we stumble upon an interesting dilemma: What is the appropriate network architecture to compare the redundant networks to? If we create the redundant network by replicating *N* times a standard network with *M* hidden units, should the redundant network be compared to the standard network with which it was created, or to a massively parallel network with *M x N* hidden units to equate the number of processing nodes? We can consider the connections between the input layer and hidden layer of a network as defining a hyperplane that bisects the problem space: the connections between the hidden layer and the output layer serve to combine the different regions defined by each hyperplane. Therefore, adding hidden units to a network is equivalent to placing more hyperplanes within the problem space, which increases the chance of the output units combining the correct hyperplane partitionings. Because the redundant networks are created by duplicating a standard network, then each subnetwork is limited to the same number of hyperplanes as the original standard network. This is equivalent to placing *M* hyperplanes into each of *N* different copies of the original problem space. The massively parallel network, however, places *M x N* hyperplanes within the original problem space. Therefore, the redundant networks will be compared to the original standard networks to equate the number of hyperplanes placed within the problem space. With this view of redundancy, it is theorized that the redundant networks will learn by driving each subnetwork into a local minima, and then combine the outputs of each subnetwork to produce the best response.

Adding redundancy to a network creates an interesting question from an engineering viewpoint: Are the added hardware requirements of the extra processing nodes and connections more than compensated for by an increase in performance? In other words, can we trade simplicity for efficiency? From a biological viewpoint, the question becomes one of maximizing functional performance (e.g. precision of timing neurons; Calvin, 1983) while keeping physical properties (e.g. head size; Glassman, 1987) to a minimum. Both of these approaches seek to find the optimal combination of performance and efficiency.

Consequently, the problem that now exists is to find the optimal level of redundancy where the increase in hardware requirements is offset by an equal or greater increase in performance. The performance of ANNs can be measured with three different metrics: probability of convergence, processing sweeps to convergence, and processing steps to convergence. The first metric measures the probability of the network to converge on a solution within a finite number of stimuli presentations. The second metric measures the number of times a training set must be presented to the ANN before successful learning occurs; this metric can be viewed as an indication of the theoretical speed of performance. Direct comparisons between the standard and redundant ANNs can be made on both of these metrics as the networks are theoretically working in parallel and, therefore, adding extra subunits in the case of the redundant ANNs will not affect computational speed. Finally, the third metric is a measure of how many computations the network must make in order to solve the problem presented-- in other words, processing efficiency. If sweeps to convergence is used as a base rate of efficiency for the standard networks, then the comparable efficiency of the redundant networks is calculated by multiplying the number of sweeps by *N*, where *N* is the level of redundancy being used.

It has been estimated from reliability theory that the brain has at least two, and as many as seven, different levels of redundancy (Glassman, 1987). Therefore, to assess the optimal level of redundancy for an ANN, the performance of a standard ANN trained on varying levels of a difficult pattern classification task (i.e. 2- to 8-parity) will be compared to the performance of ANNs with two to eight levels of redundancy.

Method

Network Architecture. The *standard* network architecture consisted of an input layer, a hidden unit layer, and an output layer: The number of input units and hidden units was equivalent to the size of the parity problem (e.g. ANNs trained on 3-parity had 3 input units, 3 hidden units, and 1 output unit). Connection weights were randomly assigned from a rectangular distribution over the range [-1, +1], and processing unit biases were initialized to 0. All biases and connections within the network were modifiable.

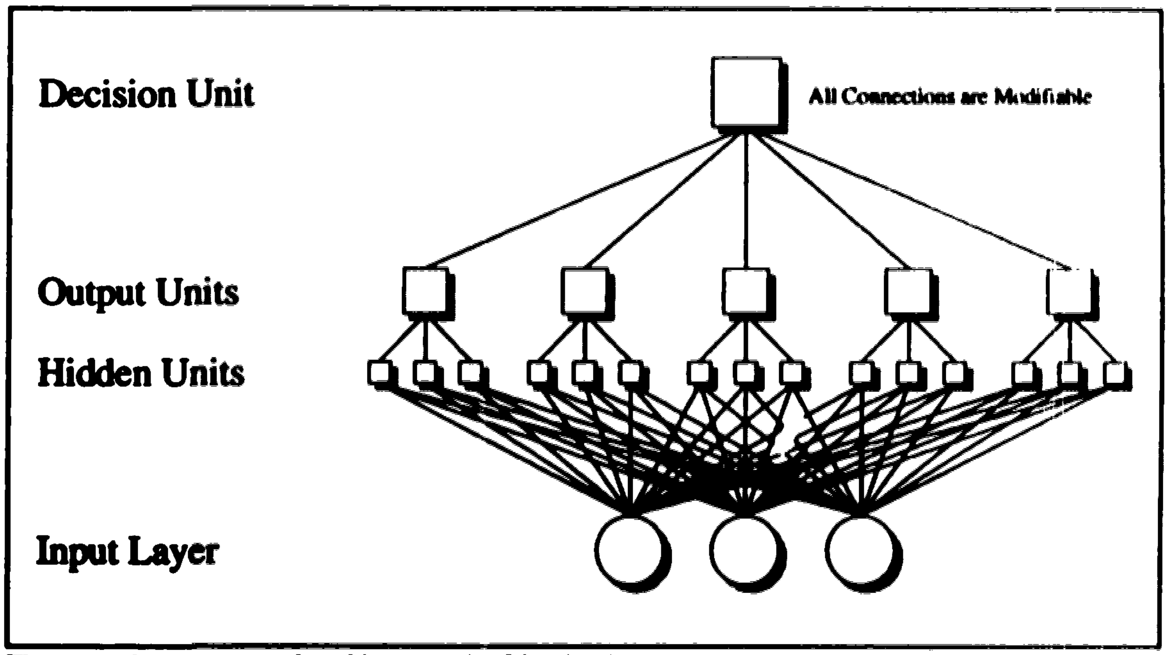

Figure 1. 3-parity network architecture with 5 levels of redundancy

The *redundant* network architecture was created by replicating the hidden unit layer and the output unit layer a set number of times. Each of the replicated output units was then connected to a *Decision Unit*, which acts as the redundant network's output unit. All connections leading into the Decision Unit are modifiable; therefore, the Decision Unit's response is a weighted sum of the replicated output units. Figure 1 shows the redundant network structure for an ANN with five levels of redundancy trained on a 3-parity problem.

It should be noted that, as opposed to a three-layer[1] network, no connections exist directly between each of replicated networks. Furthermore, each of the replicated networks was initialized independent of the others. Connection weights were randomly distributed over a larger range [-5, +5] to facilitate the theorized learning strategy, and all unit biases were set to 0. Seven different levels of redundancy were tested: 2, 3, 4, 5, 6, 7, and 8.

Training Stimuli. Parity is a linearly inseparable pattern classification task defined by the number of active input units: If the number of 1's in the input pattern is odd, then the output is 1, otherwise it is 0 (Minsky & Papert, 1969). A training set consists of 2^n distinct patterns (*n* is equal to the number of input units) comprised of all possible combinations of 0's and 1's; therefore, each training set has equal numbers of positive and negative examples of parity. ANNs were trained on 2-, 3-, 4-, 5-, 6-, 7-, and 8-parity problems which had training set sizes of 2, 8, 16, 32, 64, 128, and 256 respectively.

Training Procedure. The network was trained with the backpropagation algorithm using the *generalized delta rule* (GDR) (see Rumelhart, Hinton, & Williams, 1986). Backpropagation is described as a steepest descent optimization algorithm for traversing the surface of a weight space whose height measures error. Descent through the weight space is aided by two parameters: momentum (α) and rate-of-learning (η). Momentum is a technique for escaping local minima within the weight space by averaging the weight change for one pattern with the weight change for the previous pattern: it essentially filters out high-frequency variations of the error surface in the weight space . The rate-of-learning

[1] There is little agreement on how to count the layers within an ANN. In this paper, a layer consists of the processing node and the connections leading into it; consequently, the input nodes are *not* counted as a layer (after Wasserman, 1989).

parameter is used to dictate how large a "step" to make when traversing the weight space. For all parity problems, $\alpha = 0.9$, and $\eta = 0.1$[2]

To train the network, a pattern was randomly sampled-- without replacement-- from the pattern set and presented to the network. The network's actual output was then compared to the desired output, and connection weights and unit biases were modified according to the GDR algorithm. If the absolute difference between the actual output and desired output was less than 0.05 then a "hit" was recorded. One *sweep* of the network was completed once all patterns were presented to the network. Training of the network continued either until the maximum number of sweeps was completed (30,000) or until each pattern in a sweep produced a hit. As the initial randomness of connection weight assignment introduces considerable variability into network learning, 10 networks-- each with a different random start-- were trained for both the standard network and the redundant network.

Results and Discussion

The performance of standard ANNs versus redundant ANNs was compared using three measures: probability of convergence, sweeps to convergence, and total processing steps to convergence. Because networks will occasionally take an inordinate amount of time to converge on a solution, median scores are reported for both sweeps and steps to convergence.

[2]Although previous research (Tesauro & Janssens, 1988) has shown that learning may be facilitated for more difficult problems by adjusting the parameters α and η, these parameters were held constant for all problems as there is no indication that biological neurons manipulate their parameters with problem difficulty.

As can be seen in Table 1, the standard networks-- indicated by zero levels of redundancy-- classified the 4-parity problem successfully only 40% of the time, and failed completely to classify 5-parity and above within 30,000 processing sweeps. Averaged over the seven different parity problems, this translates into a 66% failure rate. In comparison, the average failure rate for the redundant networks was only 7%. Furthermore, it appears that as the level of redundancy increases, so does the probability of convergence (e.g. from 30% convergence with two levels of redundancy to 100% convergence with five levels of redundancy for the 7-parity problem). It should be noted, however, that the only networks to converge on a solution 100% of the time, regardless of parity size, were networks with five levels of redundancy.

Table 1. Median processing sweeps and steps to convergence as a function of parity and redundancy

Problem Size		Level of Redundancy							
		0	2	3	4	5	6	7	8
2-Parity									
	Sweeps	2551	932	841	819	663	641	660	491
	Steps	2551	1864	2522	3276	3315	3846	4620	3928
	n	10	10	10	10	10	10	10	10
3-Parity									
	Sweeps	1214	2235	973	630	587	688	447	526
	Steps	1214	4470	2919	2520	2935	4128	3129	4208
	n	10	9	10	10	10	9	10	10
4-Parity									
	Sweeps	14126	1732	1051	709	608	680	550	543
	Steps	14126	3464	3153	2836	3040	4080	3850	4344
	n	4	9	10	10	10	10	10	10
5-Parity									
	Sweeps	----	1548	997	831	634	748	747	693
	Steps	----	3096	2991	3324	3170	4488	5229	5544
	n	0	6	10	10	10	10	10	10
6-Parity									
	Sweeps	----	1846	1492	932	738	821	720	470
	Steps	----	3692	4476	3728	3690	4926	5040	3760
	n	0	8	10	10	10	10	10	10
7-Parity									
	Sweeps	----	1826	1663	874	724	579	716	664
	Steps	----	3652	4989	3496	3620	3474	5012	5312
	n	0	3	7	9	10	9	9	10
8-Parity									
	Sweeps	----	1811	1835	884	758	579	692	566
	Steps	----	3622	5505	3536	3790	3474	4844	4528
	n	0	4	9	8	10	7	10	9

Note. Maximum number of sweeps = 30000; n = number of converged networks out of 10.

Similarly, when sweeps to convergence are considered, there is a general decrease in sweeps with an increase in redundancy. Figure 2, which plots amount of processing over levels of redundancy collapsed across parity size, shows that this decrease begins to asymptote around five levels of redundancy, which suggests a floor effect. A slightly different function appears with the total processing steps to convergence, as calculated by multiplying the number of sweeps by the level of redundancy. This time, there is a slight cubic function (see Figure 2) with its lowest points being around four and five levels of redundancy depending on the problem difficulty. When the number of processing steps is averaged across all parity problems, networks with four levels of redundancy perform best, followed by networks with five levels of redundancy.

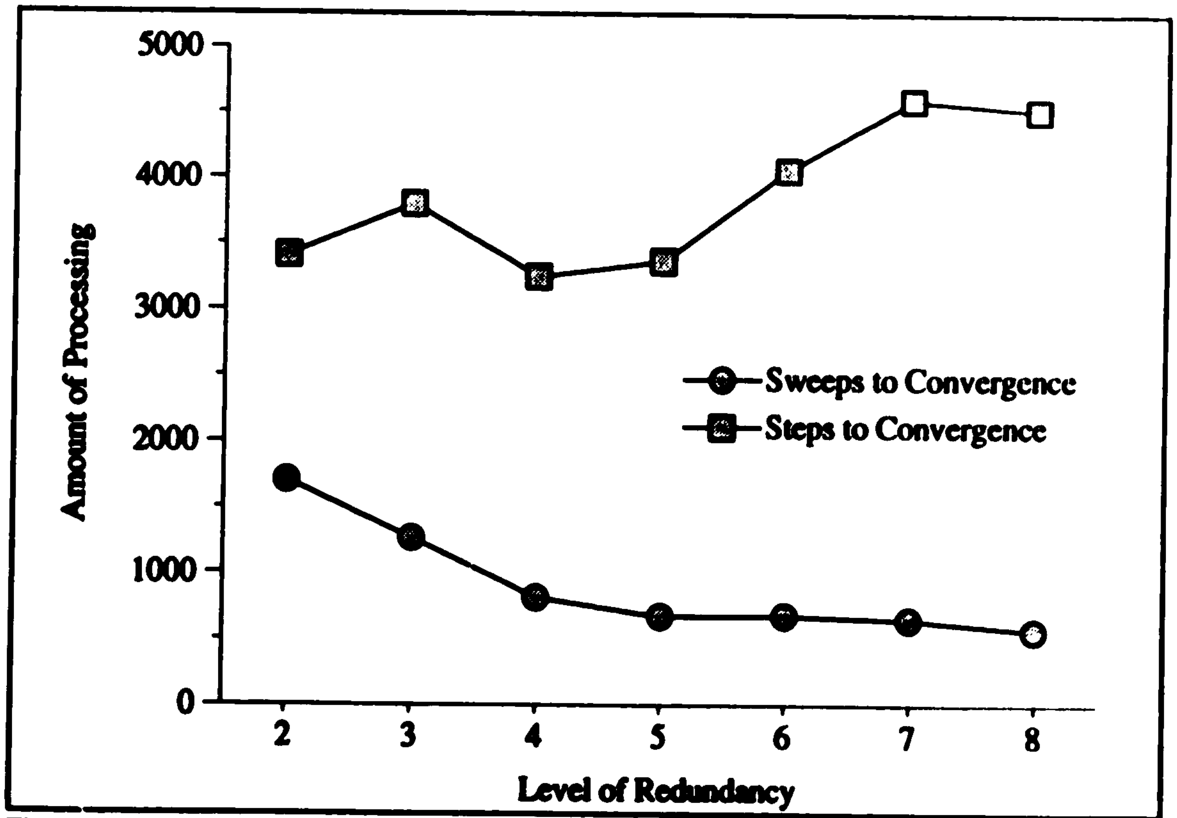

Figure 2. Network performance in terms of processing speed and efficiency for each level of redundancy averaged across parity problem size

For difficult pattern classification tasks, redundant ANNs perform better than non-redundant ANNs in terms of probability of convergence, processing speed, and processing efficiency. For the easier problems such as 2- and 3-parity, however, redundancy improves processing speed but hinders processing efficiency. In terms of the finding the level of redundancy where performance is balanced with efficiency, results suggest that ANNs with five levels of redundancy are optimal. If fewer than five levels are used, convergence cannot be guaranteed, while adding more than five levels does not increase processing speed but does decrease processing efficiency. Therefore, all further experiments within this paper will use ANNs with five levels of redundancy.

One final note on the performance difference between the standard ANNs and the redundant ANNs. As results indicate, standard ANNs failed completely to learn the larger problem sets, whereas ANNs with five levels of redundancy always converged on a solution regardless of problem set size. It is known that networks trained with the GDR have difficulties with larger problem sets, but that learning can be facilitated by manipulating the starting parameters (e.g. Tesauro & Janssens, 1988). Therefore, it is possible that the differences in performance may be related to the differences in initial network states: specifically, the starting weight parameters. The larger range of starting weights for the redundant networks was chosen to facilitate the theorized learning procedure. Although the difference in starting states may account for some of the differences in performance, pilot studies showed that the standard networks were actually hindered by such a large starting weight range. Nevertheless, future experiments will use the same starting parameters for both redundant and standard networks.

Experiment 2: Pattern Classification

Experiment 1 shows that redundancy improves the performance of ANNs trained on one type of pattern classification task (i.e. parity); however, it is not known if these results will generalize to other types of pattern classification problems, or if redundancy will improve the performance of networks with different architectures. Experiment 2 will look at the effects of redundancy on the performance of ANNs trained on two different types of difficult toy pattern classification tasks: the parity problem as in Experiment 1, and the n-$\log_2 n$-n encoder problem. Furthermore, the effect of redundancy on the "traditional" ANN architecture (e.g. Rumelhart, Hinton, & Williams, 1986) will be compared to the effect of redundancy on a different ANN architecture (Dawson & Schopflocher, 1992).

The backpropagation algorithm using the GDR requires that processing units have an activation function that is differentiable and monotonic, otherwise networks tend to fall into a local minima where they assert that some property of the pattern space *p* is not true, but fail to assert that some property of *p* is true (Rumelhart, Hinton, & Williams, 1986); Such processing units are termed *integration devices* by Ballard (1986). Recently, Dawson and Schopflocher (1992) have modified the GDR to allow training of processing units with a non-monotonic activation function, called *value units* (Ballard, 1986). Value unit ANNs learn linearly inseparable pattern classification problems much faster than integration device ANNs (Dawson & Schopflocher, 1992; Dawson, Schopflocher, Kidd & Shamanski, 1992). Consequently, it is hypothesized that redundant ANNs will converge faster than standard ANNs, and that value unit ANNs will perform better then integration device ANNs. Therefore, the best performance is expected from the redundant value unit network.

Method

Network Architecture. The standard networks used for the parity problem were equivalent to those in Experiment 1. Connection weights, however, were randomized from a rectangular distribution over the range [-2.5, +2.5] for integration device networks using a sigmoidal activation function, or [-1, +1] for value unit networks using a Gaussian activation function. Processing unit biases, regardless of activation function, were initialized to zero. The standard networks for the encoder problems consisted of *n* input units, $\log_2 n$ hidden units, and *n* output units, where *n* is equal to the size of the encoder problem. Connection weights for both integration device networks and value unit networks were randomized over the range [-1, +1] and all biases were set to zero.

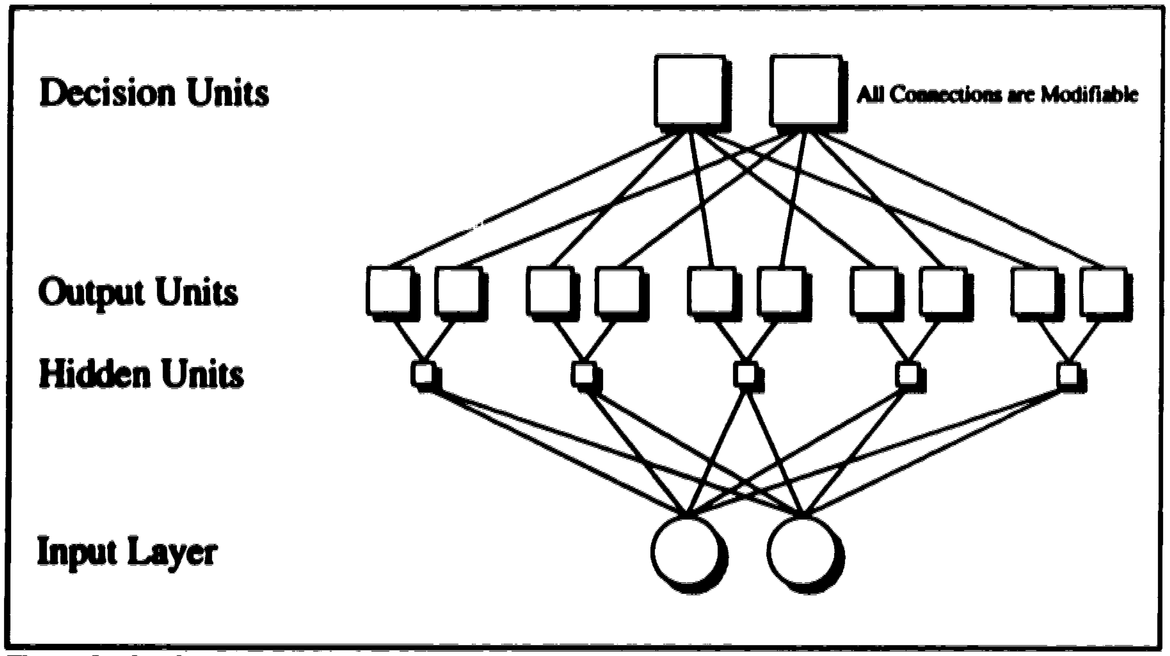

Figure 3. 2-1-2 encoder network architecture with 5 levels of redundancy

Redundant networks with five levels of redundancy were created as described in Experiment 1 with the exception that initial weight parameters were randomized over the same distribution as the standard networks: biases were set to zero. Furthermore, it should be noted that for the encoder networks, the number of Decision Units was equal to *n*, the size of the problem. As shown in Figure 3, which illustrates the redundant network architecture for the 2-1-2 encoder problem, output units are only connected to their corresponding Decision Unit. Therefore, as opposed to typical three-layer networks, connections between output units and Decision Units are not massively parallel.

Training Stimuli. The 2-, 3-, 4-, 5-, 6-, 7-, and 8-parity training sets used in this experiment were the same as those used in Experiment 1. The encoder problem, as defined by Ackley, Hinton, and Sejnowski (1985), is one in which a set of *n* orthogonal input patterns are passed through $\log_2 n$ hidden units and mapped onto a set of *n* orthogonal output patterns. For this experiment, the encoder training sets consisted of 2, 4, 8, 16, 32, or 64 input patterns composed of a single 1 and filler 0's (e.g. 1 0 0 0, 0 1 0 0, 0 0 1 0, 0 0 0 1). The output patterns and input patterns were equivalent.

Training Procedure. The networks were trained with the backpropagation algorithm using either the GDR for processing units with a sigmoidal activation function (Rumelhart, Hinton, & Williams, 1986), or a modification of the GDR for processing units with a Gaussian activation function (Dawson & Schopflocher, 1992). For the integration device networks, the parameters for both the parity problems and the encoder problems were set at $\alpha = 0.9$ and $\eta = 0.1$. Parameters for the value unit networks were $\alpha = 0$ and $\eta = 0.025$ for the parity problem[3], and $\alpha = 0$ and $\eta = 0.05$ for the encoder problems.

[3]Pilot studies showed that the optimal learning for value units occurred when η decreased as parity size increased. Although the current value of η is not the optimal learning rate, it produces consistent learning for both the small and large parity problems. Integration device learning was not affected by manipulation of η.

Training of the ANNs proceeded as described in Experiment 1. A hit was recorded if the actual output was 0.95 or higher when a 1 was desired, or 0.05 or lower when a 0 was desired, and the maximum number of sweeps allowed was held constant at 30000 for all networks. Training continued until all patterns in the set were learned or until the maximum number of sweeps was reached. As the initial random assignment of connection weights introduces variability in learning, each of the four different networks (i.e. standard integration, standard value unit, redundant integration, redundant value unit) was trained with different initial settings a total of 10 times. The minimum, median, and maximum number of sweeps to convergence, and the number of ANNs reaching convergence were recorded for each type of network.

Results and Discussion

Parity. Table 2 shows the minimum, median, and maximum number of sweeps required to reach convergence for the seven different parity problems: The number of networks to successfully converge on a solution, if less than 10, is shown in parentheses next to the median. As can be seen, the redundant networks converged on a solution 100% of the time while the standard networks often failed to converge on a solution even after 30000 sweeps. Furthermore, the redundant networks solved the problems faster (i.e. in fewer sweeps) than the standard networks for both the integration device architecture and the value unit architecture for every size of parity problem. In fact, redundancy increased the average speed of processing eight-fold, with a maximum increase of 25 times faster for the integration devices and 21 times faster for the value units: This advantage is most prominent for the larger problem sets. In terms of processing efficiency (i.e. equating the networks by multiplying the standard network sweeps by *1/N*, where *N* = 5), the redundant networks only outperform the standard networks as the problem difficulty increases.

When the different network architectures are compared, value unit networks clearly outperform integration device networks. First, the standard value unit networks converged on a solution 89% of the time whereas the standard integration device networks only found a solution 49% of the time and failed completely on the 8-parity task. Second, both the standard and redundant value unit networks had faster processing times than both types of integration device networks. In fact, for the smaller problems (i.e. 5-parity and less), the standard value unit networks actually outperformed the redundant integration device network. Finally, it should be

Table 2. Parity problem: Sweeps to convergence as a function of problem size, network architecture, and redundancy

		Network Architecture			
		Integration Device		Value Unit	
Problem Size	Sweeps to Convergence	Standard	Redundant	Standard	Redundant
2-Parity					
	Minimum	945	639	67	32
	Median	1239 (9)	844	154	151
	Maximum	2466	1838	351	200
3-Parity					
	Minimum	1028	1127	172	67
	Median	1883	1321	430	94
	Maximum	2207	2073	738	117
4-Parity					
	Minimum	2239	639	134	49
	Median	5377 (5)	807	394	108
	Maximum	26089	1324	1018	242
5-Parity					
	Minimum	2598	526	400	86
	Median	17108 (4)	675	578	102
	Maximum	28795	3412	1124	415
6-Parity					
	Minimum	2034	427	563	115
	Median	6216 (3)	738	928 (9)	208
	Maximum	12454	1653	1018	469
7-Parity					
	Minimum	2163	558	1242	84
	Median	2460 (3)	972	2142 (6)	129
	Maximum	5167	3918	11631	247
8-Parity					
	Minimum	----	565	856	84
	Median	---- (0)	1354	3553 (7)	166
	Maximum	----	3010	11437	257

Note. Numbers in parentheses indicate number of converged networks out of 10.

noted that the redundant value unit networks converged on a solution for the 8-parity problem faster than the standard integration device network trained on the 2-parity problem.

To assess the hypothesis that the individual networks within a redundant network are being trained towards local minima as assumed in Experiment 1, the underlying structure of a redundant integration device network trained on the 3-parity problem was analyzed. This analysis indicates that the entire network is not being trained towards a global minima, but that individual networks within the redundant network are being trained towards local minima. For example, Figure 4 shows the desired and actual output patterns of each individual network for a redundant integration device network that had converged after 579 sweeps. Connection weights from each individual network to

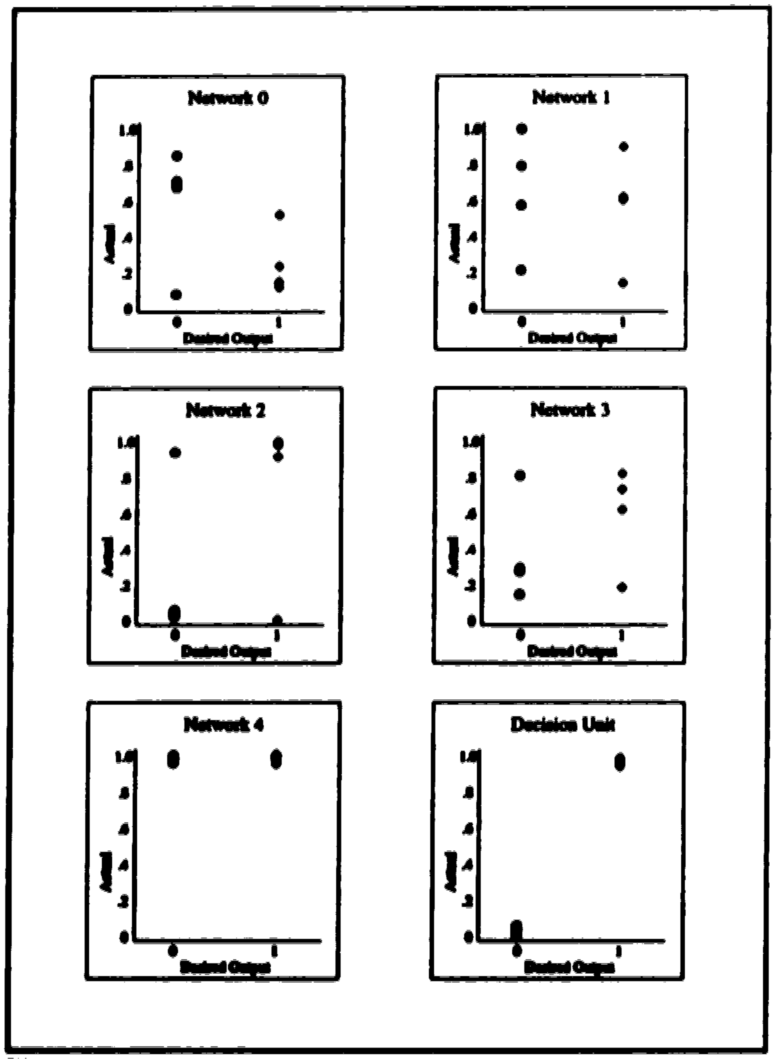

Figure 4. Actual versus desired output responses of the decision unit and its individual networks for a redundant integration device network trained on 3-parity

the decision unit are -7.42, -5.99, 6.36, 4.10, and 1.49 for Networks 0 to 4 respectively. Network 2 shows the most learning with 5 of the 8 patterns actually falling within hit parameters, and it contributes the greatest amount of excitatory activation to the Decision Unit. Network 3 also tends to classify the patterns correctly; however, it is not as accurate as Network 2 and this is reflected in its lower weighting. As a side note, the two patterns that Network 2 fails to classify are correctly classified by Network 3, and vice versa. Network 0 actually has learned to classify the opposite parity problem (i.e. responding 1 for an even number of 1's in the input pattern); it contributes a strong inhibitory response to the Decision Unit. Network 4 can be interpreted as a network that always has a disposition to respond positively, although this response is slight as indicated by the low weighting it has with the Decision Unit. Network 1 is difficult to interpret, except as a possible agent against Network 4, as most of the network's responses are above 0.5 and there is a relatively strong inhibitory weighting between it and the Decision Unit.

The above analysis supports the initial theory that redundant networks learn by driving the individual subnetworks into local minima, and then combining their weighted outputs to produce the best response. Therefore, the design decision in Experiment 1 to give the redundant networks a larger starting range for the weight parameters is validated. Furthermore, comparison of results between Experiment 1 and 2 show that the greater starting range for the weights did facilitate learning in standard integration device networks; however, redundant integration device networks initialized with the same range of weights as the standard networks still performed better. It appears that starting parameters do not overly influence the performance of both standard and redundant ANNs.

<u>Encoder</u>. Similar results are reported for the encoder problem. Table 3 shows that redundant networks, regardless of architecture, converge on a solution faster than standard networks. When the number of processing steps are taken into consideration, however, redundancy only aids the integration device networks for the larger problem sets; On the other hand, value unit networks profit greatly from redundancy. In fact, the worst performance of the redundant value unit networks (154 sweeps for the 8-3-8 problem) is better than the best performance of the redundant integration device (512 sweeps for the 4-2-4 problem). Furthermore, it appears that the redundant value unit networks take approximately the same number of sweeps to solve the larger problems as they do on the

Table 3. Encoder problem: Sweeps to convergence as a function of problems size, network architecture, and redundancy

		Network Architecture			
		Integration Device		Value Unit	
Problem Size	Sweeps to Convergence	Standard	Redundant	Standard	Redundant
2-1-2					
	Minimum	744	540	271	39
	Median	767	614	327	54
	Maximum	813	760	368	85
4-2-4					
	Minimum	1510	512	434	52
	Median	1675	626	514	76
	Maximum	2372	679	1126	145
8-3-8					
	Minimum	3086	639	605	63
	Median	3594	724	804	96
	Maximum	5350	829	1376	154
16-4-16					
	Minimum	4618	771	959	74
	Median	5360	931	1264	95
	Maximum	5780	993	1565	127
32-5-32					
	Minimum	5937	1073	950	43
	Median	6466	1175	1262	56
	Maximum	6980	1425	1694	66
64-6-64					
	Minimum	6715	1489	835	40
	Median	7301	1760	1162	47
	Maximum	7620	2184	1569	85

smaller problems while the standard value units appear to plateau for the 16-4-16 encoder problems and larger. Conversely, the number of sweeps to reach convergence for both the standard and redundant integration device networks steadily increases with problem size. When the number of pattern presentations are considered (i.e. multiplying the number of sweeps by the number of input patterns), all networks show a steady increase with problem size; however, the increase is less for the redundant networks than for the standard networks (e.g. 27 and 91 times for the value unit and integration device redundant networks versus 114 and 304 times for the respective standard networks). This suggests that redundant networks may be more resistant to the scaling problem often reported in the literature (e.g. Feldman-Stewart & Mewhort, 1994).

In conclusion, convergence on linearly inseparable pattern classification problems is much faster with redundant ANNs than with standard ANNs. Furthermore, redundant ANNs converge on a solution 100% of the time regardless of problem size, whereas the standard ANNs often fail to reach convergence. When the networks are equalized for total number of processing steps, as opposed to total network processing sweeps, the redundant networks show the most advantage on the more difficult problems. Finally, in support of Dawson and Schopflocher (1992), and Dawson, Schopflocher, Kidd, and Shamanski (1992), the value unit networks outperform the integration device networks on both the parity problem and the encoder problem in every aspect.

Experiment 3: Function Approximation

Experiment 1 and Experiment 2 have conclusively shown that redundancy can improve the performance of ANNs trained on difficult pattern classification tasks. The last question to be addressed is whether or not redundancy will improve the performance of ANNs trained on a function approximation task. With function approximation (FA), however, the number of sweeps to reach convergence is no longer an appropriate measure of network performance; therefore, performance will be evaluated via overall network error. Furthermore, as universal function approximators require an invertible activation function, value units (because of their non-monotonicity) are inappropriate for FA tasks (Dawson & Schopflocher, 1992). Consequently, only integration device networks will be evaluated.

One FA task that seems particularly well suited to neural networks is the control of robotic limbs (c.f. McClelland, Rumelhart, & Hinton, 1986; Eckmiller, 1989; Walter & Schulten, 1993; Zurada, 1992). Traditional robotic limb manipulation is achieved through a series of programmed end effector movements based on either forward or inverse kinematics (Craig, 1986). Using this approach, Churchland (1992) designed a crablike robot that can successfully reach towards an object that has been placed in front of it. Although this traditional method of robotic limb manipulation effectively mimics sensori-motor behaviour, there is no indication that the nervous system carries forth such complex trigonometric functions in the step-by-step fashion that is required. Employing ANNs to approximate the inverse kinematics function circumvents the computational complexity of the numerical solution while providing a learning mechanism for adaptation to environmental changes such as obstacles, loads, and friction (Eckmiller, 1989; Kuperstein, 1988).

Most research on robotic limb control via ANNs has been motivated by the superior performance of biological systems over the traditional robotic control algorithms (Walter & Schulten, 1993). This advantage derives from the organization of topographic maps, such as sensory and motor maps, within the brain (Churchland, 1992). Consequently, a common approach to robotic limb manipulation is to use self-organizing neural networks (e.g. Kuperstein, 1988; Eckmiller, 1989; Walter & Schulten, 1993). These ANNs normally use a variation of Kohonen's algorithm for self-organizing maps (see Wasserman, 1989). Zurada (1992), however, reports of several ANNs (e.g. Arteaga-Bravo, 1990; Nguyen, Patel, & Khorasani, 1990) that have successfully used the standard back-propagation algorithm to learn the forward and inverse kinematics problem required for robotic limb manipulation.

It has been shown that ANNs successfully learn to control robotic limbs when biological constraints are imposed on the learning algorithm used for training (e.g. Kuperstein, 1988; Eckmiller, 1989; Walter & Schulten, 1993). Experiment 3 will consider the performance of ANNs when the biological characteristic of redundancy is imposed on the network architecture. The ANNs will be trained on the inverse kinematics problem using the back-propagation algorithm. The problem space is based on Churchland's (1992) crablike robot which effectively maps the inputs from the robot's two eyes onto the required angular positions of the shoulder and elbow joints; therefore, the networks will be trained to approximate the function that maps one state space to another. It is hypothesized that ANNs with a redundant network architecture will be more accurate than ANNs with a standard network architecture.

Method

Network Architecture. The standard network architecture was a two-layer network with two input units, five hidden units, and two output units. Connection weights were randomly assigned from a rectangular distribution over the range [-1,+1]. Processing unit biases were initialized to zero. All connection weights and unit biases were modifiable.

The redundant network architecture was created by replicating the hidden unit layer and the output unit layer of the standard network five times. Each of the replicated output units was then connected to the corresponding decision unit via modifiable connections. All connection weights within the redundant network were randomly assigned from the range [-1,+1] and biases were set to zero. Each replicated network's initial state was randomized independently. As can be seen in Figure 5, which illustrates the redundant network architecture for the simulated robotic arm, no direct connections exist between the replicated networks nor between output units and decision units of different function.

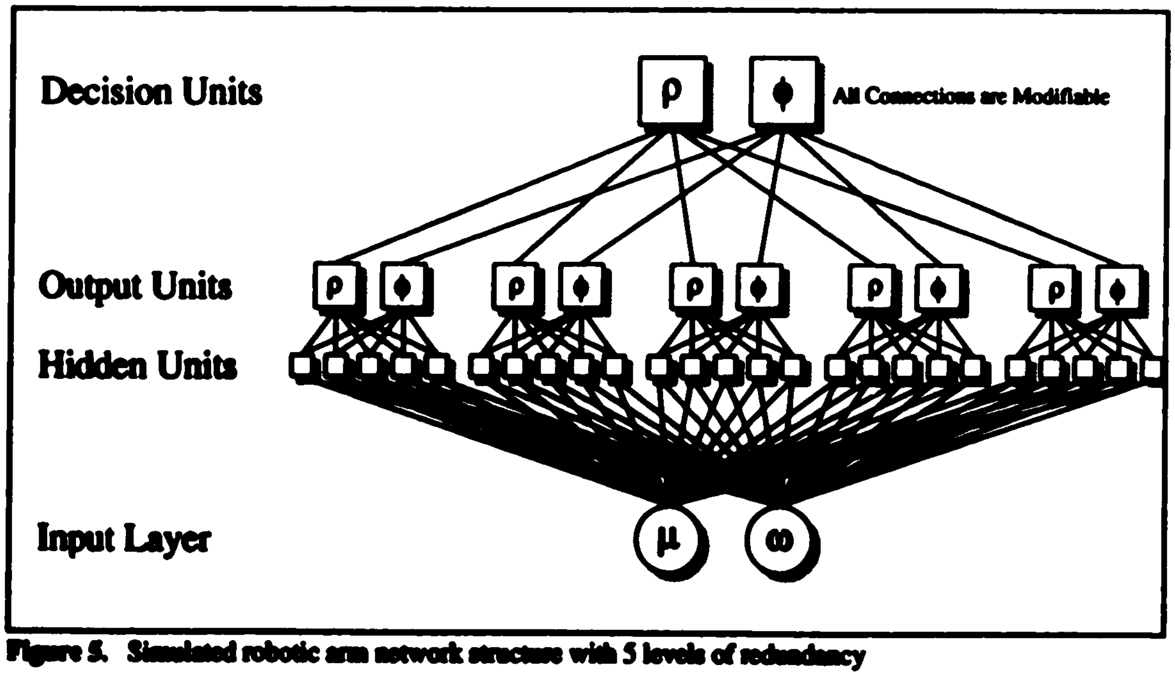

Figure 5. Simulated robotic arm network structure with 5 levels of redundancy

Training Stimuli. The simulated robotic arm was modelled after Churchland's (1992) crablike schematic creature with two rotatable eyes and an extendable arm that can reach towards an object placed in front of it (see Figure 6). The eyes were placed 6 units apart, and each arm segment was 7 units in length. Therefore, the robot's world was 14 units in depth and 28 units in width. To create the training set, an object was placed randomly in front of the simulated robot: If the object fell within an unreachable area (i.e.grey area in Figure 6) then a new position was randomly chosen. The grey semi-circle exists within the problem space because the robotic arm joints were limited to a range of 0° to 180°. Inputs to the network were the two angles (μ, ω) that the eyes subtended when converged on the object, while the desired network outputs were the angles (ρ, ϕ) that the shoulder joint and elbow joint made in order for the arm to contact the object. All angles were divided by 180 to fall within the range of 0 to 1. The inputs could be considered two-dimensional sensory-state space coordinates, and the outputs would then be considered as separate two-dimensional motor-state space coordinates. The network, therefore, learns the appropriate mapping

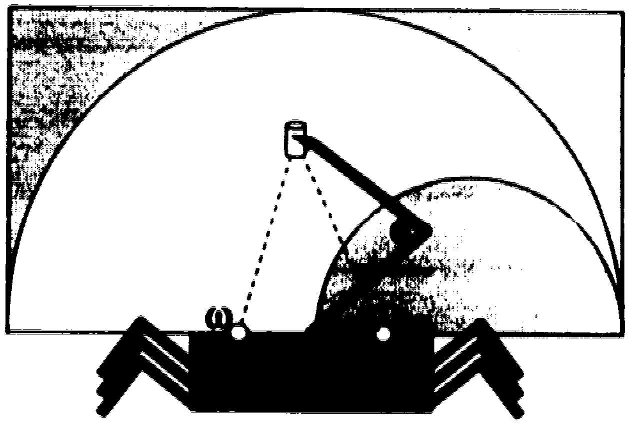

Figure 6. Problem space definition for the simulated robotic arm

between the two state spaces (see also Zipser & Andersen, 1988). As the mapping of the two state spaces forms a continuous function, there are an infinite number of input/output pairs; however, practicality limited the training set to 50 randomly chosen pairs.

Training Procedure. The network was trained with the backpropagation algorithm using the GDR for processing units with a sigmoidal activation function (Rumelhart, Hinton, & Williams, 1986). For the simulated robotic arm network, $\alpha = 0.9$, and $\eta = 0.1$.

To train the network, a pattern was randomly sampled-- without replacement-- from the pattern set and presented to the network. The network's actual output was then compared to the desired output, and connection weights and unit biases were modified according to the above algorithm. If the absolute difference between the actual output and desired output was less than 0.001-- approximately 0.2°-- then a "hit" was recorded. One sweep of the network was completed once all patterns were presented to the network. Training of the network continued until 50,000 sweeps were completed.

To assess the network's ability to learn the function approximation problem, total network SSE was recorded at $\log_{10}$ intervals beginning with 100 sweeps and ending with 50,000 sweeps; therefore, 23 scores were recorded for each network. To assess the network's accuracy, the final network responses for the shoulder (ρ) and elbow (ϕ) angle for each pattern were recorded at the completion of training. Again, as the initial randomness of connection weight assignment produces great variability in network learning, 10 different networks were trained for both the standard network architecture and the redundant network architecture.

<u>Results and Discussion</u>

The means and standard deviations of the total network SSE for both the standard and redundant networks trained on the simulated robotic arm are shown in Figure 7. As can be seen, standard networks perform better than redundant networks in terms of mean SSE when the number of processing sweeps is relatively small (e.g. < 1000). When the number of sweeps is increased, however, the redundant networks clearly performs better than the standard networks. Furthermore, variability in network responding decreases faster for the

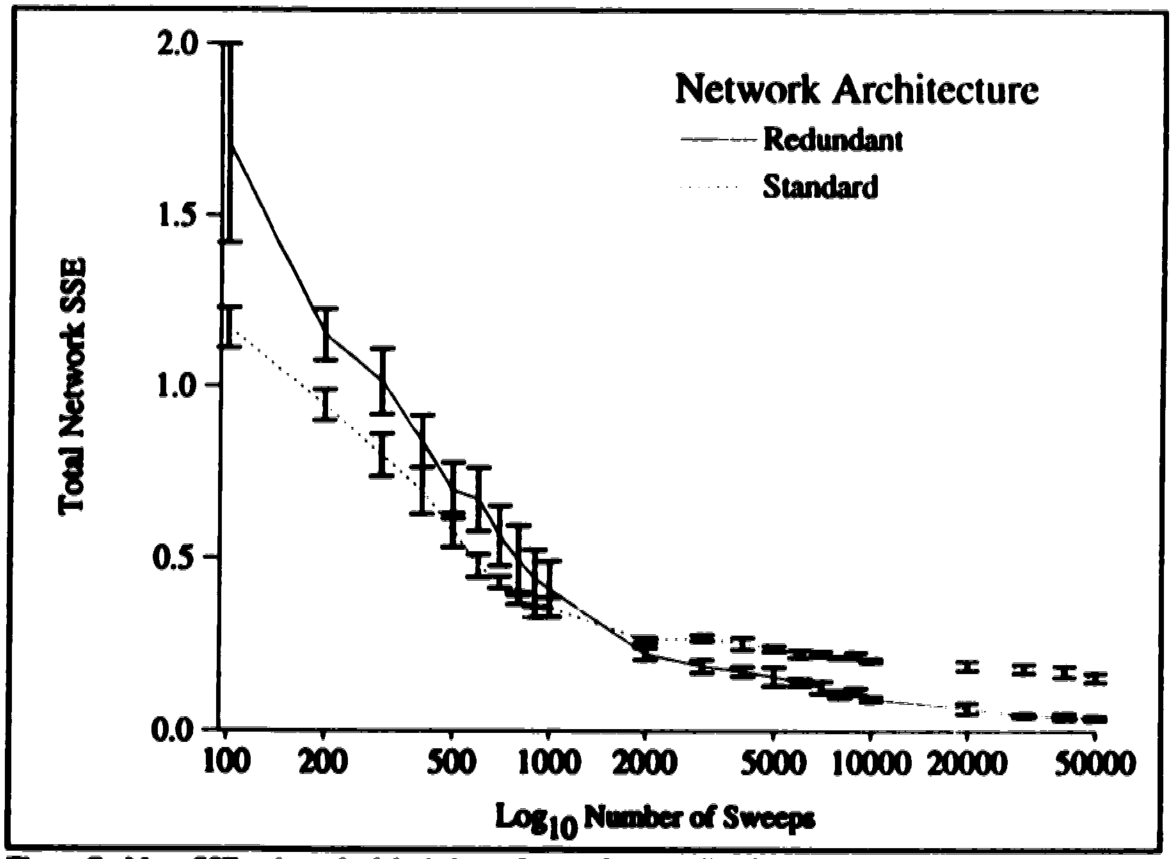

Figure 7. Mean SSE and standard deviations of network responding for the redundant network and standard network trained on the simulated robotic arm problem

redundant networks than for the standard networks. In terms of equating the number of processing steps taken (i.e. network efficiency), the performance of the standard network after *X* sweeps should be compared to the performance of the redundant network after *X/N* sweeps, where *N* is the degree of redundancy (this is in contrast to the PC tasks as convergence was not a criterion in the FA task). As Figure 7 shows, the redundant ANNs produce poorer performance than the standard ANNs when the number of sweeps is relatively small. When we compare 5,000 sweeps of the standard ANN with 1,000 sweeps of the redundant ANN, the standard ANN has a SSE of approximately 0.24 whereas the redundant ANN has a SSE of approximately 0.41. As the number of sweeps increases, the redundant networks begin to perform as well as the standard networks (e.g. SSE of 0.21 at 10,000 sweeps and 0.22 at 2,000 for the standard and redundant ANNs respectively). The difference between the two ANNs, however, increases dramatically in favor of the redundant network when the number of sweeps is large. Figure 7 shows that the mean network SSE for the standard network after 50,000 sweeps (approximately 0.16) is about four times greater than the mean network SSE for the redundant network after 10,000 sweeps (approximately 0.04). Furthermore, the smaller standard deviations of the redundant networks show there is less variability in network responding.

To assess how accurate the networks were in reaching towards the target, two different error measurements were calculated: angular error of the shoulder and elbow joints, and positional error (X,Y coordinate) of the robot's hand. Figure 8 shows the two different error measurements for the 10 standard ANNs after 50,000 sweeps, the errors for the 10 redundant ANNs after 10,000 sweeps (to equate processing steps) and after 50,000 sweeps (to equate processing sweeps). The angular errors for the all three ANNs show a relatively positive correlation-- if the ρ is less than it should be, then ϕ will also be less, if

ρ is overestimated, then ϕ is overestimated. Although all networks had low mean errors (e.g. standard network: 0.9° and -0.05° for ρ and ϕ; redundant network [10,000]: 0.46° and

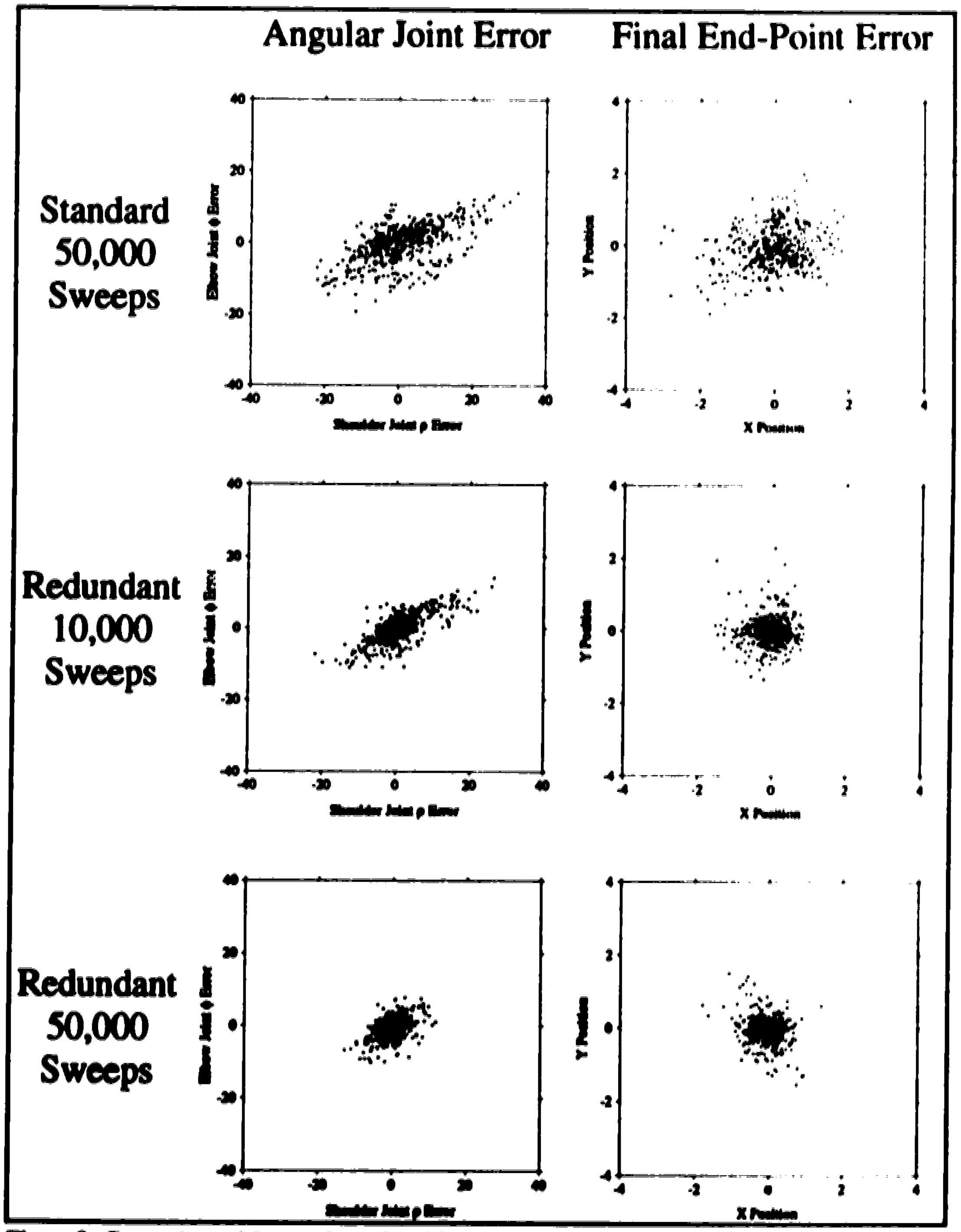

Figure 8. Comparison of the angular and positional errors for the standard ANNs after 50,000 sweeps, and the redundant ANNs after 10,000 and 50,000 sweeps.

0.83° for ρ and ϕ; redundant network [50,000]: -0.06° and -0.6° for ρ and ϕ), the amount of variability in responding was much greater for the standard network. As can be seen in Figure 8, the redundant networks had less variability in their responding than the standard networks as indicated by the tighter clustering of error points. Another way of viewing this result is that the redundant networks are more stable in their responding than standard networks.

A close examination of the angular errors show that some of the angles were out as much as 30° or more for the standard networks. This would seem to be an extreme failure of the system to correctly reach the object. When these angles are used to compute the final cartesian plane coordinates of the robot's hand, however, the errors are not so extreme. In fact, the maximal positional errors were limited to just under 4 units for the standard networks, and under 2 units for the redundant networks. Again, both redundant ANNs were closer to the desired target location than the standard ANNs. Finally, Figure 8 shows that contrary to the angular errors, there were no real correlations between positional errors in the X direction with positional errors in the Y direction. This would suggest that the network was not learning the trigonometric functions required to reach an object (as would be indicated by small angular errors), but was learning to map visuo-spatial coordinates to motor-spatial coordinates (as indicated by the small positional errors).

As hypothesized, redundant networks are more accurate than standard networks on function approximation problems. Not only is the final mean SSE less for the redundant ANNs than for the standard ANNs, but the variability in responding is less for the redundant networks as well. In terms of efficiency, however, the redundant ANNs only outperform standard ANNs at the higher end of the sweep scale.

One criticism that can be re-raised at this point is that the redundant network has five times as many hidden units as the standard network; therefore, it is not surprising that redundancy provides better performance. Earlier work by Nguyen, Patel, and Khorasani (1990; as cited in Zurada, 1992) suggests, however, that the performance of redundant networks may be better than standard networks with equivalent numbers of processing units. Nguyen et al. trained two different networks, one a fully connected three layer network (BP), and the other a modification of the BP network that symmetrically divided both layers of hidden units so that the output nodes only received activation from half the total number of hidden units (BPOS). Although the BPOS network required slightly more training sweeps than the BP network to reach the same accuracy, the BPOS network had an overall shorter training time due to the smaller number of weight changes required. As the BPOS network is similar to the redundant network in that both architectures limit the number of connections, it may be possible to hypothesize that redundant networks should perform better than standard networks with equivalent numbers of processing units.

General Discussion

The results from both the PC problem and the FA problem confirm Izui and Pentland's (1990) mathematical analysis of redundant networks: Redundancy produces faster convergence, more accurate results, and more stable networks than comparable standard networks. In terms of the relevance of redundancy to the performance of ANNs in general, redundant networks should be considered as a viable alternative to standard networks. The initial cost of the extra hardware associated with redundancy is far out-weighed by the savings in training, accuracy in responding, and network stability produced by redundant processes. This improvement in performance may be due to individual networks training towards not a global optimum, but an orthogonal local optimum, much as Schapire's (1990) learning algorithm encourages. Indeed, analysis of the individual networks within the redundant network showed that the networks are finding local complimentary minima in the problem space; therefore, each individual network is being trained towards a local optimum. Instead of developing one perfect algorithm or network, we should consider combining smaller and simpler networks that have their own specialization (see also Ballard, 1986).

These results have shown that there is another alternative to the combining algorithms used by committee machines (e.g. mean response, winner-take-all, median response, etc.). The modifiable connections from the individual output units to the decision unit allows the network to train itself. As opposed to taking the mean output response of individual networks, which gives equal weighting to all networks, the amount of contribution is weighted according to how well the individual networks classify the problem.

Furthermore, all individual networks contribute to the final result, unlike winner-take-all or median response methods. Consequently, the modifiable connections of the decision unit have proven to be a functional alternative to those methods conventionally used while preserving some semblance of biological systems.

Ironically, it is the modifiable connections of the decision unit that provide the strongest line of argument against the redundant network. Critics may claim that our redundant network is nothing more than a three-layer network, with the output units simply making up the second layer of hidden units. If this were the case, then it would not be surprising that the redundant networks were able to converge on all PC problems, as a three-layer network is capable of carving a problem space into an arbitrary number of distinct regions (Lippman, 1987). The response to this criticism centers on the architecture of the redundant network. Each of the subnetworks is an isolated unit that is capable, in theory, of solving the problem on its own. In fact, Figure 4 shows that is indeed possible. This architectural constraint is clearly different from the massive parallelism common to networks with two layers of hidden units. Nevertheless, future research will concentrate on the differences between redundant networks and standard networks with equivalent numbers of processing units. It is postulated that redundant networks will be more resistant to damage and will generalize better than standard networks as suggested by biological research (e.g. Glassman, 1987; Leon, 1992).

The biological plausibility of the decision unit allows us to speculate on the relevance of redundancy in biological systems. By modelling redundancy with an ANN, we can begin to confirm or deny some of the theories and findings introduced earlier. First, the apparent specialization of each individual network within the redundant network is in line with biological evidence from certain crustaceans whose movement is regulated via a set of redundant command neurons, each specialized for a specific range of motion. "Specialization, which is made possible by redundancy, presumably increases the integrative flexibility and effector repertoire of the command system" (Kovac, Davis, Matera, & Croll, 1983; p.1535). Moreover, the specialization of one network for detecting the opposite parity can be related to the parallel ON and OFF channels leading from the retina to the visual cortex. Normal vision requires the push-pull action of both channels, although Swindale (1986) reports that a visual system with a blocked ON channel can still detect the onset of a dark spot, and therefore the absence of a light spot. Similarly, the opposite parity detector signifies the absence of odd parity by detecting even parity. Redundant information is carried by the ON and OFF channels of the visual system, and by the odd and even parity detectors of the ANN.

Other evolutionary theories are supported by the performance of the redundant ANN. For example, the increased precision of the redundant network over the standard network on the FA problem lends credence to Calvin's (1983) hypothesis about redundancy evolving to increase the precision of a system. In fact, as the upper limit of network sweeps increases, the worst redundant network is more precise than the best standard network. Also, the number of sweeps to train both the FA network and the PC network suggests that it is easier to evolve several crude mechanisms working in parallel than one extremely effective mechanism. Analysis of Figure 4, however, suggest that there may be an upper limit to the amount of redundancy required for optimal performance. Three of the subnetworks clearly show learning of the problem, whereas the responses of the other two subnetworks are difficult to interpret. Too much redundancy may actually cause overlearning to occur and, therefore, may be detrimental to network performance. This is also shown in Experiment 1 where learning efficiency was hampered by too much redundancy for the easier problems. The problem of what level of redundancy to use may be analogous to the "Three Bears" principle described by Seidenberg and McClelland (1989): too much or too little redundancy, and the problem will not be solved efficiently.

Further research will consider the possibility of loss of redundancy accounting for loss of functioning in patients with debilitating diseases. As stated earlier, it is widely held that redundancy in the brain allows for functional recovery after brain damage (Almli & Finger, 1992). It follows that loss of redundancy may cause loss of functioning. Modelling redundancy via computer simulation has a distinct advantage over biological models, in that precise ablations can be performed on artificial neural networks (see Hinton & Shallice, 1991; Farah, O'Reilly, & Vecera, 1993). Therefore, one can monitor the performance of the ANN when specific connections are cut. After each connection is cut, it is expected that there will be a slight decline in performance until the ANN has compensated for the missing link. This type of recovery is typical of patients recuperating from brain trauma-- functional recovery is not immediate, but gradually improves (Marshall, 1984). Eventually, after enough redundant connections are cut, then recovery of function for the ANN should be impossible.

Furthermore, the results of Experiment 3 show that the variability in making a response is much greater for a non-redundant network than a fully redundant network;

therefore, as redundancy decreases, variability in responding should increase. Monitoring the variability changes should be an effective tool for estimating how much damage the system has suffered, and should even predict when terminal drop will occur. A practical application of this theory has already been hinted at by Patterson, Foster, and Heron, who conclude that for assessing damage by Multiple Sclerosis, "variability is a more sensitive indicator of visual pathway damage than the usual measure of mean" (1980, p.143). By attempting to model this increase in variability, we may be in a better position to understand the underlying damage associated with such diseases as Multiple Sclerosis and Alzheimers.

References

Ackley, D. H., Hinton, G. E., & Sejnowski, T. J. (1985). A learning algorithm for Boltzmann machines. *Cognitive Science, 9,* 147-169.

Almli, C. R., & Finger, S. (1992). Brain injury and recovery of function: Theories and mechanisms of functional reorganization. *Journal of Head Trauma Rehabilitation, 7,* 70-77.

Ballard, D. (1986). Cortical structures and parallel processing: Structure and function. *The Behavioral and Brain Sciences, 9,* 67-120.

Baxt, W. G. (1992). Improving the accuracy of an artificial neural network using multiple differently trained networks. *Neural Computation, 4,* 772-780.

Calvin, W. H. (1983). A stone's throw and its launch window: Timing precision and its implications for language and hominid brains. *Journal of Theoretical Biology, 104,* 121-135.

Churchland, P. M. (1992). *A neurocomputational perspective: The nature of mind and the structure of science.* Cambridge, MA: MIT Press.

Craig, J. J. (1986). *Introduction to robotics mechanics & control.* Reading, MA: Addison-Wesley.

Dawson, M. R. W., & Schopflocher, D. P. (1992). Modifying the Generalized Delta Rule to train networks of non-monotonic processors for pattern classification. *Connection Science, 4,* 19-31.

Dawson, M. R. W., Schopflocher, D. P., Kidd, J., & Shamanski, K. S. (1992). Training networks of value units. *Proceedings of the Ninth Canadian Conference on Artificial Intelligence*. 244-250.

Dawson, M. W. R., & Shamanski, K. S. (1993). Connectionism, confusion, and cognitive science. *Journal of Intelligent Systems*, In press.

Dawson, M. W. R., Shamanski, K. S., & Medler, D. A. (1993). From connectionism to cognitive science. In L. Goldfarb (Ed.) *Proceedings of the Fifth University of New Brunswick Artificial Intelligence Symposium*. Fredericton, NB: UNB Press.

Eckmiller, R. (1989). Generation of movement trajectories in primates and robots. In I. Aleksander (Ed.). *Neural computing architectures: The design of brain-like machines*. Cambridge, MA: MIT Press.

Farah, M. J., O'Reilly, R. C., & Vecera, S. P. (1993). Dissociated overt and covert recognition as an emergent property of a lesioned neural network. *Psychological Review, 100*, 571-588.

Feldman-Stewart, D., & Mewhort, D. J. K. (1994). Learning in small connectionist networks does not generalize to large networks. *Psychological Research, in press.*

Glassman, R. B. (1987). An hypothesis about redundancy and reliability in the brains of higher species: Analogies with genes, internal organs, and engineering systems. *Neuroscience & Biobehavioral Reviews, 11*, 275-285.

Graham, R. B. (1990). *Physiological psychology*. Belmont, CA: Wadsworth.

Hebb, D. O. (1949). *The organization of behavior: A neuropsychological theory*. New York: Wiley.

Hinton, G. E., & Shallice, T. (1991). Lesioning an attractor network: Investigations of acquired dyslexia. *Psychological Review, 98*, 74-95.

Izui, Y., & Pentland, A. (1990). Analysis of neural networks with redundancy. *Neural Computation, 2*, 226-238.

Jacobson, J. Z. (1976). Relative possibilities of loops and redundant connections in neural nets. *Journal of Mathematical Psychology, 13*, 148-162.

James, W. (1972/1890). *Psychology: Briefer course*. London: Collier-MacMillan.

Kandel, E. R., Schwartz, J. H., & Jessell, T. M. (1991). *Principles of neural science* (3rd ed.). New York: Elsevier.

Kolb, B., & Whishaw, I. Q. (1990). *Fundamentals of human neuropsychology* (3rd ed.). New York: W. H. Freeman.

Kovac, M. P., Davis, W. J., Matera, E. M., & Croll, R. P. (1983). Organization of synaptic inputs to paracerebral feeding command interneurons of *Pleurobranchaea californica*. I. Excitatory inputs. *Journal of Neurophysiology, 49*, 1517-1538.

Kuperstein, M. (1988). Neural model of adaptive hand-eye coordination for single postures. *Science, 239*, 1308-1311.

Leon, M. (1992). The neurobiology of filial learning. *Annual Review of Psychology, 43*, 377-398.

Lewandowsky, S. (1993). The rewards and hazards of computer simulations. *Psychological Science, 4*, 236-243.

Lippman, R. P. (1987). An introduction to computing with neural nets. *IEEE ASSP Magazine, April*, 4-22.

McClelland, J. L. (1981). Retrieving general and specific knowledge from stored knowledge of specifics. *Proceedings of the Third Annual Conference of the Cognitive Science Society* (pp. 170-172). Berkeley, CA.

McClelland, J. L., Rumelhart, D. E., & Hinton, G. E. (1986). The appeal of parallel distributed processing. In D. E. Rumelhart, J. L. McClelland, and the PDP Group (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition. Vol 1.* (pp 3-44). Cambridge, MA.: MIT Press.

McCloskey, M. (1991). Networks and theories: The place of connectionism in cognitive science. *Psychological Science, 2,* 387-395.

Marshall, J. F. (1984). Brain function: neural adaptations and recovery from injury. *Annual Review of Psychology, 35,* 277-308.

Minsky, M. L., & Papert, S. A. (1969). *Perceptrons.* Cambridge, MA: MIT Press.

Nicholls, J. G., Martin, A. R., & Wallace, B. G. (1992). *From neuron to brain* (3rd ed.). Sunderland, MA: Sinauer.

Patterson, V. H., Foster, D. H., & Heron, J. R. (1980). Variability of visual threshold in Multiple Sclerosis: Effect of background luminance on frequency of seeing. *Brain: A Journal of Neurology, 103,* 139-147.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In D. E. Rumelhart, J. L. McClelland, and the PDP Group (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition. Vol 1.* (pp 318-362). Cambridge, MA.: MIT Press.

Schapire, R. (1990). The strength of weak learnability. *Machine Learning, 5,* 197-227.

Seidenberg, M. S. (1993). Connectionist models and cognitive theory. *Psychological Science, 4,* 228-235.

Seidenberg, M. S., & McClelland, J. L. (1989). A distributed model of word recognition and naming. *Psychological Review, 96,* 523-568.

Selfridge, O. G. (1959). Pandemonium: A paradigm for learning. In D. V. Blake and A. M. Uttley (Eds.), *Proceedings of the Symposium on Mechanization of Thought Processes* (pp. 511-529). London: H. M. Stationary Office.

Smith, A. (1984). Early and long-term recovery from brain damage in children and adults: Evolution of concepts of localization, plasticity, and recovery. In C. R. Almli and S. Finger (Eds.), *Early brain damage: Research orientations and clinical observations, Vol I* (pp 299-324). New York: Academic.

Strehler, B. L., & Lestienne, R. (1986). Evidence on precise time-coded symbols and memory of patterns in monkey cortical neuronal spike trains. *Proceedings of the National Academy of Sciences of the United States of America, 83,* 9812-9816.

Swindale, N. V. (1986). Parallel channels and redundant mechanisms in visual cortex. *Nature, 322,* 775-776.

Tabary, G., & Salaün, I. (1992). Control of a redundant articulated system by neural networks. *Neural Networks, 5,* 305-311.

Tesauro, G., & Janssens, B. (1988). Scaling relationships in back-propagation learning. *Complex Systems, 2,* 39-44.

Tulving, E. (1972). Episodic and semantic memory. In E. Tulving and W. Donaldson (Eds.), *Organization of Memory.* (pp 385-397). New York: Academic.

Walter, J. A., & Schulten, K. J. (1993). Implementation of self-organizing neural networks for visuo-motor control of an industrial robot. *IEEE Transactions on Neural Networks, 4*, 86-95.

Wasserman, P. D. (1989). *Neural computing: Theory and practice.* New York: Van Norstrand Reinhold.

Zipser, D., & Andersen, R. A. (1988). A back-propagation programmed network that simulates response properties of a subset of posterior parietal neurons. *Nature, 331*, 679-684.

Zurada, J. M. (1992). *Introduction to artificial neural systems*. St. Paul, MN: West.