ORIGINAL ARTICLE

David A. Medler · Michael R. W. Dawson

Training redundant artificial neural networks: Imposing biology on technology

Received: 14 September 1993 / Accepted: 22 December 1993

Abstract One biological principle that is often overlooked in the design of artificial neural networks (ANNs) is redundancy. Redundancy is the replication of processes within the brain. This paper examines the effects of redundancy on learning in ANNs when given either a function-approximation task or a pattern-classification task. The function-approximation task simulated a robotic arm reaching toward an object in two-dimensional space, and the pattern-classification task was detecting parity. Results indicated that redundant ANNs learned the pattern-classification problem much faster, and converge 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 in 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.

Introduction

Recently a considerable amount of debate has been generated as to 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 often is 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 counterintuitive 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 the recovery from behavioral 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, which was the reason for recovery of function following unilateral brain lesions (Gall & Spurzheim, 1810–1819; cited in Almli & Finger, 1992). Although alternative explanations of functional recovery exist (e.g., alternate strategies, vicarious functioning, diaschisis), redundancy is still held as a viable theory today (for a review of theories of recovery following brain trauma see Almli & Finger, 1992, and Marshall, 1984).

Further neurophysiological evidence for redundancy comes from studies of patients with hydrocephalus. A review of 279 adult patients 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; as cited in Glassman, 1987). This suggests 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. In fact, Glassman's (1987) calculations of the brain's safety factor, using reliability theory, indicate that the most conservative estimate of brain size is at least twice the minimum size required for short-term survival.

Other neurophysiological evidence for redundancy in the brain comes from recent studies of animal physiology. Kovac, Davis, Matera, and Croll (1983) studied the nervous system of *Pleurobranchaea californica* extensively, and found several physiological systems that produced essentially the same behavior; however, when combined, these systems greatly enhanced the precision of simple and

D. A. Medler (✉) · M. R. W. Dawson
Biological Computation Project, Department of Psychology,
University of Alberta, Edmonton, Alberta T6G 2E9, Canada,
E-mail: medler@psych.ualberta.ca

complex movements. Strehler and Lestienne (1986) analyzed the firing patterns of neurons in the monkey's visual cortex and found redundant doubling of coded information in the regularity of triplets of impulses triggered by specific stimuli. Furthermore, 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 layers 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 thought previously held, 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. Complementing the physiological evidence for the relevance of redundancy, there is a sizable theoretical literature on the relevance of redundancy.

Most theoretical work on the relevance of biological redundancy has centered on the factors in the evolution of redundancy. One common assumption, as was 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 an event rarely survived, it is unlikely to exert any natural-selection pressure for neural spare capacity in anticipation of brain damage (cf. Glassman 1987). But this argument 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 alternative evolutional theories.

Calvin (1983) considered the problem early hominids must have faced when trying to knock their prey down by throwing an object. The timing precision required to strike a target increases eightfold with a mere doubling of throwing distance; however, the precision of a single timing neuron is too crude to allow effective strikes at any significant distance. So Calvin proposed that the brain evolved redundant timing neurons to increase the timing accuracy above the known accuracy of any single neuron. This increase in the number of timing neurons to compensate for an otherwise deficient system is related to Swindale's (1986) hypothesis that it is easier to evolve several crude mechanisms that work in parallel to perform a function than one especially effective neural mechanism. Consequently, 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. Leon also noted that the neonate brain is far less developed than the adult brain, despite the fact that most of survival learning must occur within the first few months of life; he suggests that redundant systems exist to ensure learning, even with a degraded nervous system. On the other hand, Jacobson (1976) considered the connections between neurons involved in a memory trace to be 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.

We have seen that redundancy is a viable biological property, but can it be effectively implemented in ANNs? Redundancy has mostly been ignored in the design of ANNs. Recently, though, there has been a flurry of connectionist research on the use of 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 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 adapt itself successfully to the terrain as it moves across it. Both Baxt's (1992) network and Tabary and Salaün's (1992) network are not truly redundant in the way we 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 combined successfully to solve a larger problem.

Another form of computational redundancy widely studied today centers on 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 the machines are 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, calculating the mean output, or 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.

These research examples have nevertheless centered on improving the performance of ANNs solely from an engineering perspective. 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 is 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 (1990) 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 to study the effects of redundancy on ANN learning and performance on two different types of task. The first task is a function approximation (FA) problem of controlling a simulated robotic arm trained to reach toward an object in two-dimensional space. The second task consists of different versions of a difficult pattern-classification (PC) problem: 3-, 5-, and 7-parity. It is hypothesized that, when compared to standard networks, the redundant networks will be more accurate (i. e., have less variability) on the FA problem, and will converge faster on the PC problem.

Experiment 1: Function approximation

One task that seems particularly well suited to neural networks is the control of robotic limbs (cf. 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. Using this approach, Churchland (1992) has designed a crablike robot that can reach successfully toward an object that has been placed in front of it. Although this traditional method of robotic-limb manipulation effectively mimics sensorimotor behavior, there is no indication that the nervous system carries forth such complex trigonometric functions in the step-by-step fashion 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 several ANNs (e. g., Arteaga-Bravo, 1990; Nguyen, Patel & Khorasani, 1990; both as cited in Zurada, 1992) that have successfully used the standard back-propagation algorithm to learn the forward- and inverse-kinematics problems required for robotic-limb manipulation.

It has been shown that ANNs learn to control robotic limbs successfully when biological constraints are imposed on the learning algorithm used for training (e. g., Kuperstein, 1988; Eckmiller, 1989; Walter & Schulten, 1993). In Experiment 1 we 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 by 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; the networks will therefore 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[1] with two input units, two hidden units, and two output units. Connection weights were randomly assigned from a rectangular distribution over the range (–5, +5). 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. A decision

[1] There is little agreement as to the way of counting the number of layers in an ANN. In this paper, only the nodes affecting the computational capabilities of the network are counted; so a network consisting of an input layer, a hidden-unit layer, and an output layer is considered to be a two-layer network (cf. Wasserman, 1989)

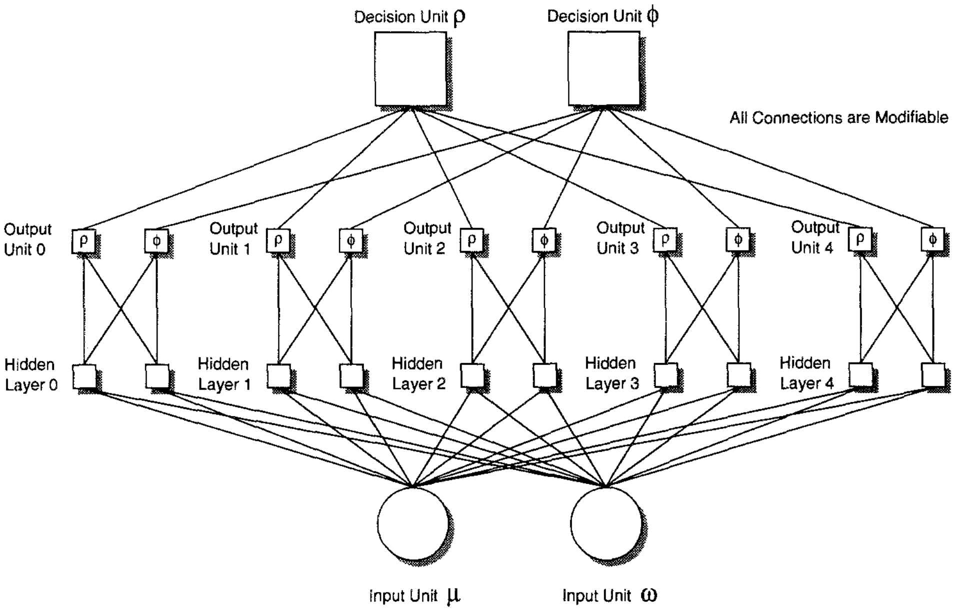

Fig. 1 Redundant network architecture for the simulated robotic arm

unit acts as an output unit for the redundant network. All connection weights within the redundant network were randomly assigned from the range (–5, +5), and all biases were initialized to zero: Each replicated network's initial state was randomized independently of the others. As can be seen in Figure 1, which illustrates the redundant-network architecture for the simulated robotic arm, there are no direct connections between the replicated networks or between output units and decision units of different function.

Training stimuli. The simulated robotic arm was modeled after Churchland's (1992) crablike schematic creature with two rotatable eyes and an extendable arm that can reach toward an object placed in front of it (see Figure 2). An object was placed randomly in front of the simulated robot; if the object fell within an unreachable area (i.e., the gray area in Figure 2) then a new position was randomly chosen. 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) made by the shoulder and elbow joints for the arm to contact the object. All angles were normalized 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 between the two state spaces (see also Zipser & Andersen, 1988). As the mapping of the two spaces is continuous, there are an infinite number of input and output pairs; however, practicality limited the training set to 50 randomly chosen pairs.

Training procedure. The network was trained with the back-propagation algorithm by use of 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. The rate-of-learning parameter is used to dictate how large a step to make when traversing the weight space. 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 the desired output was less than 0.001 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 or until each pattern in a sweep produced a hit.

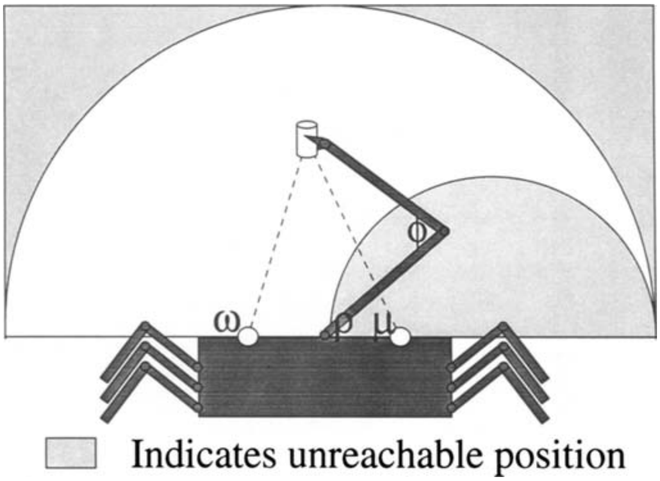

Fig. 2 Problem space definition for the simulated robotic arm

To assess the network's ability to learn the function-approximation problem, maximum network sweeps were increased from 100 to 50 000 in $\log_{10}$ steps. The total network sum of squared errors (SSE) – as measured by the difference between desired and actual network response – as well as the SSE for each individual output (ρ, ϕ), were recorded at each maximum sweep step. As the initial randomness of connection-weight assignment produces great variability in network learning, five different networks were trained for both the standard-network and the redundant-network architectures.

Results

The total SSE range and the median for the simulated robotic arm for both the standard and the redundant networks are shown in Figure 3. As can be seen, median SSE decreases faster for the redundant networks than for the standard networks. In fact, the average median SSE for the redundant network is significantly less than the average median SSE for

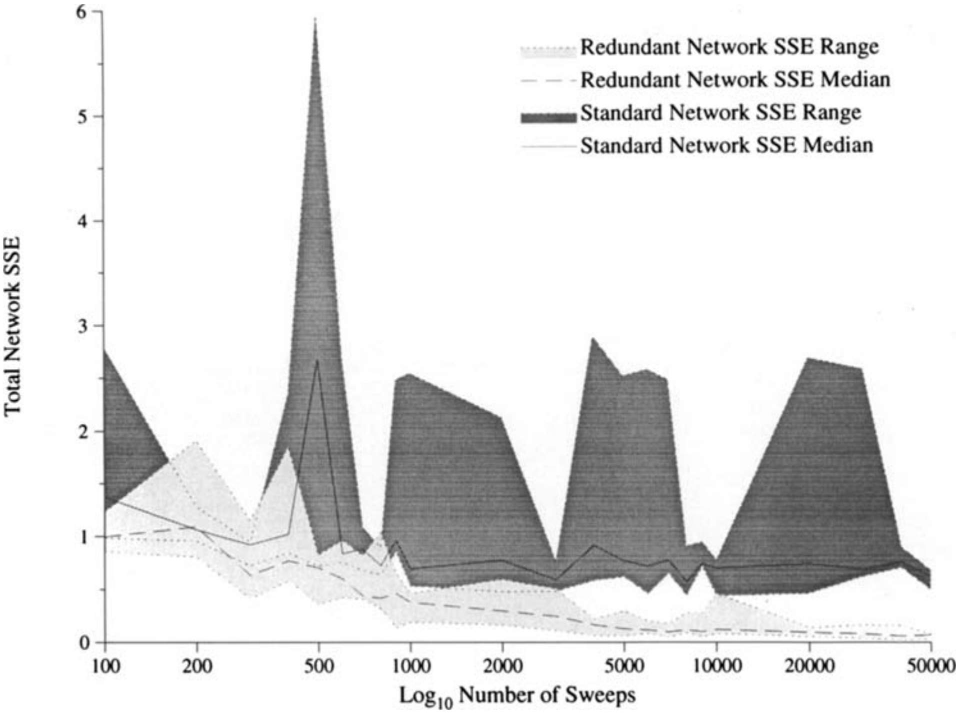

Fig. 3 Overall network SSE range and median for the simulated robotic arm: Standard vs. redundant network architecture

the standard network, (0.355 and 0.892 respectively), $F(1{,}44) = 23.899, p < .001$. Furthermore, the range of the total SSE is significantly less for the redundant network than for the standard network, $F(1{,}44) = 23.90, p < .001$. This holds true for both the elbow joint ϕ, $F(1{,}44) = 15.95, p < .001$, and the shoulder joint ρ, $F(1{,}44) = 6.91, p < .05$.

All calculations so far have centered on the total network-processing time, which is roughly equal for the two networks, as the redundant network theoretically works in parallel. We can also make comparisons between the total number of processing steps taken by each network (i. e., compare the performance of the standard network after *X* sweeps to the performance of the redundant network after *X/N* sweeps, where *N* is the degree of redundancy). Taking the total number of network-processing steps into consideration, there is little difference between the redundant ANN and the standard ANN when the number of sweeps is relatively small. In fact, when in Figure 3 we compare 1 000 sweeps of the standard ANN with 200 sweeps of the redundant ANN, the standard ANN actually has a lower median SSE than the redundant ANN (approximately 0.70 vs. 1.09). The difference between the two ANNs, however, increases dramatically in favor of the redundant network when the number of sweeps increases. Figure 3 shows that the median-network SSE for the standard network after 50 000 sweeps (approximately 0.63) is about five times greater than the median-network SSE for the redundant network after 10 000 sweeps (approximately 0.12).

Discussion

As was hypothesized, redundant networks are significantly more accurate than standard networks on function-approximation problems. Not only is the median SSE less for the redundant ANNs than for the standard ANNs, but the SSE range is significantly less for redundant networks as well. This means that the responding of the redundant ANNs is much less variable – or more stable – than the responding of the standard ANNs. Also, the advantage for the redundant ANNs increases with the number of sweeps completed when the total network-processing time is considered. This advantage holds even at the higher end of the sweep scale when the redundant and standard networks are equalized for the total number of processing steps.

One criticism that can be raised at this point is that the redundant network has five times as many hidden units as the standard network; so it is not surprising that redundancy provides a better performance. Earlier work by Nguyen et al. (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. (1990) trained two different networks, one a fully connected three-layer network (BP), and the other a modification of the BP network that divided both layers of hidden units symmetrically so that the output nodes received activation from only 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 is possible to conclude that redundant networks should perform better than standard networks with equivalent numbers of processing units.

Experiment 2: Pattern Classification

Experiment 1 has shown that redundancy improves the performance of ANNs trained on a function-approximation task in which performance is measured by overall network error. But, how does redundancy affect ANNs trained on tasks in which performance is measured by the amount of network time required to converge on a precise solution? Experiment 2 looked at the effects of redundancy on the number of network sweeps required for ANNs to learn a difficult pattern-classification task. Furthermore, the effect of redundancy on the standard ANN architecture (e.g., Rumelhart et al., 1986) was compared to the effect of redundancy on a different architecture (Dawson & Schopflocher, 1992).

Originally, the back-propagation algorithm was developed on the assumption that the activation function for processing units had to be differentiable and monotonic (Rumelhart et al., 1986). Such processing units are termed *integration devices* by Ballard (1986). Recently, however, Dawson and Schopflocher (1992) have shown that processing units with a nonmonotonic activation function – called *value units* (Ballard, 1986) – can learn linearly inseparable pattern-classification problems much faster than integration devices can. 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 in the pattern-classification problems were two-layer networks with one output unit. The number of input units and hidden units, however, were equivalent to the size of the parity problem being solved: namely 3, 5, or 7 units for the respective parity problem. Connection weights were randomized from a rectangular distribution over the range (–5, +5) for integration-device networks with a sigmoidal activation function, or (–1, +1) for value-unit networks with a Gaussian activation function. Processing-unit biases, regardless of activation function, were initialized to zero. The redundant networks were created as was described in Experiment 1, except that there was only one decision unit for each network.

Training stimuli. Parity is a rather difficult classification task defined by the number of active input units. It the number of 1s in the input pattern is odd, then the output is 1, otherwise it is 0 (Minsky & Papert, 1969). ANNs were presented with 3-, 5-, or 7-parity problems. Training set sizes were 8, 32, or 128 orthogonal input patterns for the 3-, 5-, and 7-parity problems respectively; consequently, there were equal numbers of positive and negative examples of parity in each training set.

Training procedure. The networks were trained with the back-propagation algorithm by either the GDR for processing units with a sigmoidal activation function (Rumelhart et al., 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 were set at $\alpha = 0.9$ and $\eta = 0.1$. Parameters for the value unit networks were $\alpha = 0$ and $\eta = 0.05$.

Training of the ANNs proceeded as described in Experiment 1 with some exceptions. First, 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. Second, 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. Again, 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

Table 1 shows the minimum, median, and maximum number of sweeps required to reach convergence and the total number of networks out of 10 to reach convergence for the 3-, 5-, and 7-parity problems. The redundant networks solved the problems in fewer sweeps than the standard networks. Furthermore, the redundant networks converged

Table 1 Sweeps to convergence as a function of network architecture and processing unit type

Number of Sweeps	Network Architecture					
	Standard Network			Redundant Network		
	3-Parity	5-Parity	7-Parity[a]	3-Parity	5-Parity	7-Parity
	Integration Device ANNs					
Minimum	661	5817	–	261	576	353
Median	2599	6943	–	623	717	1237
Maximum	24850	8068	–	1017	1047	2853
n	8	3	0	10	10	10
	Value Unit ANNs					
Minimum	49	213	1042	37	34	71
Median	81	258	2744	67	84	97
Maximum	200	1015	28322	156	130	302
n	10	9	8	10	10	10

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

[a] Because of the difficulty of the 7-parity problem, different values of η were tried. Value units learned best with $\eta = 0.01$, whereas integration devices failed to learn at all values of η.

on a solution 100% of the time while the standard networks often failed to converge on a solution even after 30000 sweeps. When equalized for the total number of network-processing steps (i.e., when the standard network sweeps were multiplied by $1/N$, where $N = 5$), the redundant networks outperform the standard networks only as the problem difficulty increases. And finally, the value-unit networks converged much faster than the integration-device networks for both the standard-network and the redundant-network architectures. The standard value-unit networks also converged on a solution more often than the standard integration-device networks, particularly with the more difficult problems.

To assess whether either the individual networks within a redundant network are being trained toward local minima, or the entire network is being trained toward a global optimum, we analyzed the underlying structure of a redundant integration-device network trained on the 3-parity problem. Analysis indicates that the individual networks within the redundant network are being trained toward local minima. For example, Figure 4 shows the desired and the 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 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 five of the eight 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, which 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 has actually learned to classify the opposite parity problem (i.e., responding 1 for an even number of 1s 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 is 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 Network 1's responses are above 0.5 and there is a relatively strong inhibitory weighting between it and the Decision Unit.

Discussion

Convergence on 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 failed to reach convergence. In fact, the standard integration-device networks only reached convergence 80% of the time on the 3-parity problem, and completely failed to converge on a solution for the 7-parity problem, even after 30000 sweeps. The standard value-unit networks, on the other hand, converged on a solution at least 80% of the time, even on the most difficult problem. When the networks are equalized for the total number of processing steps, as opposed to the total network-processing time, the redundant networks outperform the standard networks only on the more difficult problems. Finally, in support of Dawson and Schopflocher (1992), the value-unit networks outperform the integration-device networks on the PC problem in every aspect.

General discussion

The results from both the PC and the FA problems 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 outweighed by the savings in training, the accuracy in responding, and the network stability produced by redundant processes. This improvement in performance may be due to individual networks training toward not a global optimum, but toward 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 find local complimentary minima in the problem space; so each individual network is being trained toward a local optimum. Instead of developing one perfect algorithm or network, we should consider combining smaller and simpler networks that have their own specialization (cf. Ballard, 1986).

Our 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 some semblance of biological systems is preserved.

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

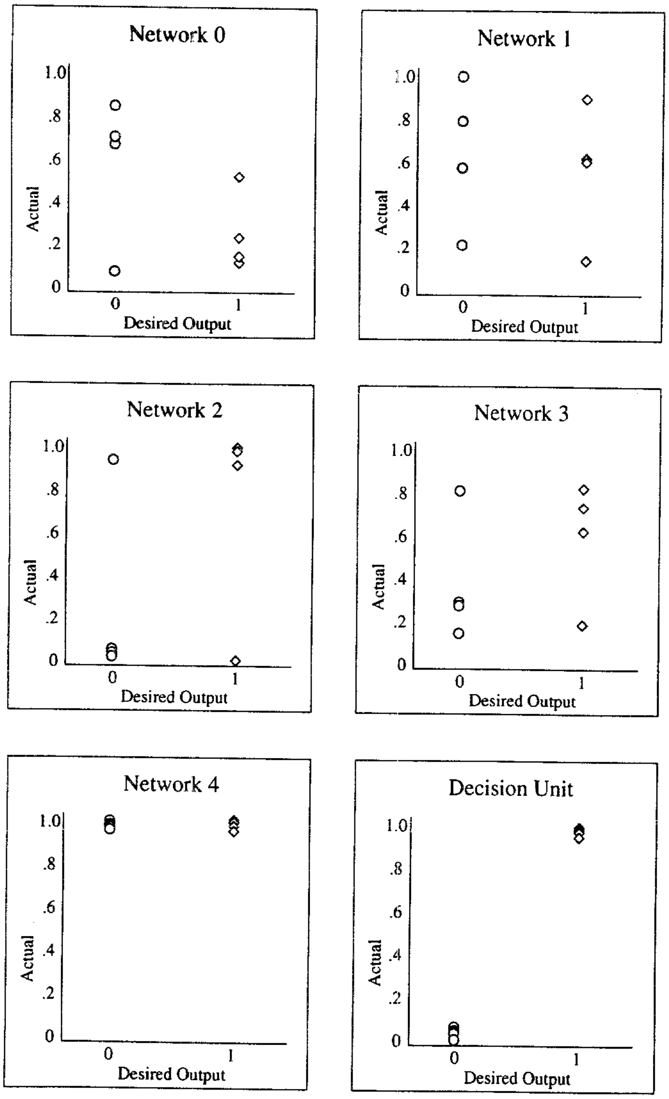

Fig. 4 Actual vs. desired output responses of the decision unit and its individual networks for a redundant integration-device network trained on 3-parity

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. Figure 4 shows that this in fact is possible. This architectural constraint is clearly different from the massive parallelism common to networks that have two layers of hidden units. Nevertheless, future research will concentrate on the differences between redundant and standard networks that have equal numbers of processing units. It is postulated that redundant networks will be more resistant to damage and will generalize better than standard networks, as is 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 modeling 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 particular range of motion (Kovac et al., 1983). 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 evolutional 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 needed to train both the FA and the PC networks suggests that it is easier to evolve several crude mechanisms working in parallel than one extremely effective mechanism. Analysis of Figure 4, however, suggests 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, and therefore may be detrimental to network performance.

Further research will consider the possibility of loss of redundancy accounting for loss of functioning in patients suffering from debilitating diseases. As was 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. Modeling 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). One can therefore 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 who are recuperating from brain trauma-functional recovery is not immediate, but gradually improves (Marshall, 1984). Eventually, after enough redundant connections are cut, recovery of function for the ANN should be impossible.

Furthermore, the results of Experiment 1 show that the variability in making a response is much greater for a nonredundant network than for a fully redundant network; therefore, as redundancy decreases, variability in responding should increase. Monitoring the variability changes should be an effective way of 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 (1980), who conclude that for the assessment of damage caused by multiple sclerosis, "variability is a more sensitive indicator of visual pathway damage than the usual measure of mean" (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 Alzheimer's disease.

Acknowledgements This research was supported by Natural Science and Engineering Research Council of Canada (NSERC) Research Grant 2038 and NSERC Equipment Grant 138704, both awarded to M. R. W. Dawson. We would like to thank S. Lewandowsky and an anonymous reviewer for valuable comments on the original draft.

References

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. *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.

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. W. R., & Shamanski, K. S. (1994). 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.

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

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.

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.

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

McClelland, J. L., Rumelhart, D. E., & Hinton, G. E. (1986). The appeal of parallel distributed processing. In D. E. Rumelhart, J. L. McClelland, & 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.

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

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, 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, & 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.

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.

Tulving, E. (1972). Episodic and semantic memory. In E. Tulving & 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.